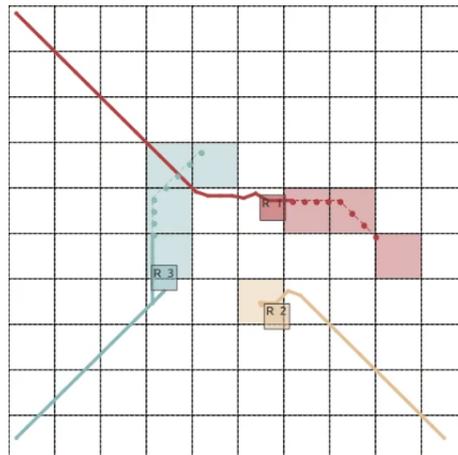


# Interconnection between Communication and Suboptimality for Distributed Control Systems

## Kumulative Dissertation

Tobias Sprodowski



vom Fachbereich 3 genehmigte Fassung (Mathematik und Informatik)  
der Universität Bremen  
zur Erlangung des Grades eines Doktors der Ingenieurwissenschaften – Dr.-Ing. –

23. März 2021

Datum des Promotionskolloquiums: 18.03.2021

*Gutachter*

Prof. Dr.-Ing. Udo Frese

Dr. rer. nat. Jürgen Pannek

# Eidesstattliche Erklärung

---

Hiermit versichere ich, dass ich

1. die Arbeit ohne unerlaubte fremde Hilfe angefertigt habe,
2. keine anderen als die von mir angegebenen Quellen und Hilfsmittel benutzt habe  
und
3. die den benutzten Werken wörtlich oder inhaltlich entnommenen Stellen als solche  
kenntlich gemacht habe.

Bremen, 23. März 2021

---

Tobias Sprodowski

# Abstract

---

Today, distributed systems are present in every day's life: For example, distributed systems occur in the form of the internet, system architectures utilising edge computing, robotic scenarios in production or manufacturing processes, or the upcoming spreading of autonomous, connected vehicles. On the one hand, the mentioned examples highlight that distributed systems may occur naturally due to their occurrence of independent entities. On the other hand, external reasons motivate to divide artificially a whole system into subsystems to escape from a single point of failure and lack of scalability. However, distributed systems require a coordination or communication mechanism and an appropriate choice of control methods, which are capable of handling such systems. Here, the Distributed Model Predictive Control scheme (DMPC) was successfully applied in various application fields such as process control, robotic scenarios and traffic scenarios. As these robotic or traffic scenarios as examples for distributed systems encounter a high changing dynamic, the chosen coordination and communication mechanism should be efficient due to necessary synchronisation between the subsystems. In this thesis, a non-cooperative setting is examined, where each subsystem follows an individual assigned target and has to exchange information steadily to ensure first and foremost collision avoidance. With utilisation of wireless communication resources and their finite bandwidth and channels, this work present methods to attenuate the communication effort between the subsystems. Based on a quantisation implemented on the communication exchange between the subsystems, it is shown that each subsystems achieves their assigned target, i.e. the overall system still converge. The improvements based on our quantisation method for the communication load is presented and for a robotic scenario convergence is shown utilising a weaker assumption than terminal constraints based on traffic rules. Additionally, to improve convergence of the overall system, dynamic priority rules are introduced, which calculate an dynamic optimisation order for the subsystems, where different schemes are utilised: While the first is based on a simple sort based on a priority criterion in every time instant, the latter prevents deadlocks and examine additionally concurrent execution of the subsystems. We close with an overview of possible applications and transformation to other scenarios and give an outlook on further developments.

# Kurzfassung

---

Verteilte Systeme sind heute überall im täglichen Leben vorhanden: Als Beispiele sind das Internet als globales verteiltes System, Edge-Computing, robotergestützte Szenarien in Produktions- oder Fertigungsprozessen oder die bevorstehende Verbreitung autonomer, vernetzter Fahrzeuge zu nennen. Während in den genannten Szenarien verteilte Systeme bereits aufgrund ihrer bestehenden Architektur aus unabhängigen Entitäten bestehen, motiviert es andererseits dazu, ein Gesamtsystem künstlich aufzuspalten, um Ausfallwahrscheinlichkeiten und Rechenaufwand zu minimieren und Skalierbarkeit zu verbessern. Dieser Vorteil erzwingt jedoch die Notwendigkeit der Koordination oder Kommunikation und eine geeignete Wahl von Steuerungsmethoden, die in der Lage sind, solche Systeme zu handhaben. Dazu wurde die verteilte modellprädiktive Regelung (DMPC) in vielen Anwendungsfeldern wie z.B. der Prozesssteuerung, Roboterszenarien oder der Verkehrssteuerung erfolgreich eingesetzt. Da diese verteilten Systeme oft mit einer hohen Änderungsdynamik insbesondere in Verkehrs- oder Roboterszenarien einhergehen, sollte der Koordinations- und Kommunikationsmechanismus zwischen den Teilsystemen effizient sein. In dieser Arbeit untersuchen wir eine nicht-kooperative Problemstellung, in dem jedes Teilsystem einem individuell zugewiesenen Ziel folgt und ständig Informationen austauschen muss, um in erster Linie Kollisionsvermeidung zu gewährleisten. Da die oft verwendete drahtlose Kommunikation beschränkte Kapazitäten hat, wollen wir Methoden zur Minimierung des Kommunikationsaufwands zwischen den Teilsystemen vorstellen. Wir zeigen anhand unserer implementierten Quantisierungsmethode, dass das Gesamtsystem mit reduzierter Kommunikation immer noch Konvergenz erreicht, d. h. jedes Teilsystem erreicht das ihm zugeordnete Ziel. Wir stellen unsere Quantifizierungsmethode mit Verbesserungen für den Kommunikationsaufwand in dieser Arbeit vor und zeigen darüber hinaus Konvergenz für ein Roboter-Szenario unter Anwendung einer Annahme basierend auf Verkehrsregeln. Um die feste Optimierungsabfolge der Teilsysteme aufzubrechen, untersuchen wir darüber hinaus Prioritätsregeln, bei denen unterschiedliche Schemata verwendet werden: Während das erste Schema auf einer einfachen Sortierung basiert, die auf einem Prioritätskriterium in jedem Zeitmoment basiert, verhindert das zweite Deadlocks und untersucht zusätzlich die Möglichkeit der gleichzeitigen Ausführung der Teilsysteme. Wir schließen mit einem Überblick über mögliche Anwendungen und Übertragungen auf andere Szenarien und geben einen Ausblick auf weitere Entwicklungen.

# Danksagung

---

Diese Arbeit entstand hauptsächlich im Rahmen meiner Tätigkeit an der Universität Bremen vom September 2014 bis Dezember 2019 in der Arbeitsgruppe *Dynamics in Logistics* von Herrn Dr. Jürgen Pannek. Für die sehr gute Betreuung und auch die Möglichkeit, mich über die Zeit weiterzuentwickeln und viele Dinge auszuprobieren, möchte ich sehr herzlich bei ihm bedanken. Ohne diese wissenschaftliche Stelle und Betreuung wäre eine solche Arbeit nicht möglich gewesen. Ganz herzlich möchte ich mich auch bei Herrn Prof. Dr.-Ing. Udo Frese bedanken, der die Doktorandenbetreuung auf den letzten Metern übernommen hat und mich sowohl inhaltlich als auch in den formalen Dingen sehr gut unterstützt hat. Ich danke auch Herrn Prof. Dr.-Ing. Michael Freitag, der mir im BIBA die Möglichkeit gab, an bestehenden und neuen Projekten weiter mitzuarbeiten und mich auch in der Prüfungskommission unterstützt hat. Bei der Prüfungskommission möchte ich mich auch bei Herrn Prof. Dr. Rolf Drechsler und Frau Prof. Dr. Kathrin Flaßkamp, Dirk Schweers und Lukas Rust bedanken, die ebenfalls unkompliziert und schnell sich bereit erklärt haben, mitzuwirken.

Ein großer Dank geht auch an meine ehemalige Arbeitsgruppe, insbesondere an Kishwer Abdul Khaliq, die langjährig meine Bürokollegin war und mich auch immer wieder motiviert hat. Darüber hinaus bedanke ich mich bei Haniyeh Dastyar, die nahtlos diesen Job übernommen hat. Bedanken möchte ich mich auch bei Elham Behmanesh, Ping Liu, Marcella Bernardo Papini, Gabriel Icarte und Qiang Zhang, die immer gutes Feedback in unseren wöchentlichen Kolloquien gegeben haben. Ein großer Dank gilt auch Thorben Funke, dessen detailliertes Feedback sehr hilfreich war. Sehr dankbar bin ich auch den BIBA-Kolleginnen und Kollegen, die mich auf diesem Weg in vielerlei Hinsicht unterstützt und begleitet haben, hier sind insbesondere Marit Hoffmeyer und Hendrik Stern hervorzuheben.

Zum Schluss möchte ich mich auch ganz herzlich bei meiner Familie bedanken, die mich immer ermutigt haben, unbekannte Wege zu beschreiten und vor allem ein großer Dank an Carolin, deren Maß an Verständnis und Unterstützung nie aufgebraucht war.

# Contents

---

<b>Abstract</b>	<b>iv</b>
<b>Kurzfassung</b>	<b>v</b>
<b>1 Motivation</b>	<b>1</b>
<b>2 State of the art</b>	<b>5</b>
2.1 Model predictive control . . . . .	5
2.2 Distributed model predictive control . . . . .	11
2.2.1 Control schemes . . . . .	11
2.2.2 Coordination . . . . .	12
2.2.3 Communication . . . . .	17
2.2.4 Quantisation of communication . . . . .	19
2.2.5 Stability . . . . .	21
2.2.6 Priority rules . . . . .	23
2.3 Summary of state of the art . . . . .	24
<b>3 Problem setup</b>	<b>26</b>
3.1 Centralised communication . . . . .	28
3.2 Distributed communication . . . . .	30
<b>4 Quantisation and stability</b>	<b>32</b>
4.1 Reduction of communication . . . . .	33
4.1.1 Occupancy grid . . . . .	33
4.1.2 Prediction coherence . . . . .	37
4.1.3 Differential communication . . . . .	41
4.1.4 Feasibility . . . . .	43
4.1.5 Interval communication . . . . .	44
4.2 Convergence of the distributed system . . . . .	52
4.2.1 Sufficient prediction horizon length . . . . .	56
4.3 Contributions of the corresponding publications . . . . .	62
<b>5 Priority rules</b>	<b>63</b>
5.1 Criteria for priority rules . . . . .	63

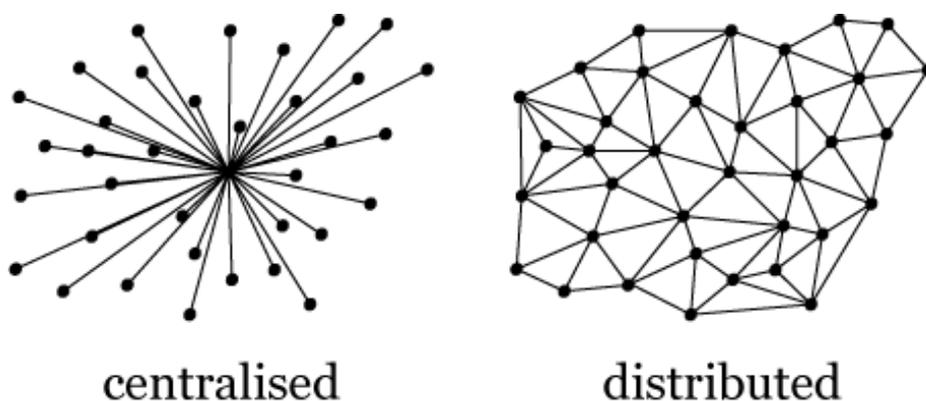
5.2	Priority schemes . . . . .	64
5.2.1	Simple priority assignment . . . . .	65
5.2.2	Priority scheme with memory function . . . . .	66
5.2.3	Dependencies and concurrent execution . . . . .	67
5.2.4	Impact of priority rules . . . . .	69
5.2.5	Contribution of the corresponding publication . . . . .	71
<b>6</b>	<b>Applications and outlook</b>	<b>73</b>
6.1	Applications for quantised distributed model predictive control . . . . .	73
6.2	Conclusions & outlook . . . . .	74
	<b>List of Tables</b>	<b>95</b>
	<b>List of Figures</b>	<b>97</b>
<b>A</b>	<b>Attached publications by the author</b>	<b>98</b>
A.1	Stability of distributed MPC in an intersection scenario . . . . .	99
A.2	Occupancy Grid based Distributed MPC for Mobile Robots . . . . .	111
A.3	Connection between Quantisation and Bandwidth Requirements of Distributed Model Predictive Control . . . . .	118
A.4	Differential Communication with Distributed MPC based on Occupancy Grid	125
A.5	Interval Superposition Arithmetic Inspired Communication for Distributed Model Predictive Control . . . . .	153
A.6	Relaxed collision constraints based on interval superposition principle in a DMPC scheme . . . . .	162
A.7	Collision avoidance for mobile robots based on an occupancy grid . . . . .	171
A.8	Dynamic-priority-based DMPC with an occupancy grid for mobile systems	198
A.9	Impact of Quantisation on Consistency of DMPC in Street Traffic with Dynamic Priority Rules . . . . .	228
A.10	Analytical Aspects of Distributed MPC Based on an Occupancy Grid for Mobile Robots . . . . .	231
A.11	Frequency based Model Predictive Control of a Manufacturing System . . . . .	258
A.12	A multi-product job shop scenario utilising Model Predictive Control . . . . .	265
<b>B</b>	<b>Software</b>	<b>300</b>

# 1

## Motivation

Nowadays, the production and logistic scenarios have to cope with the increased product complexity and the stricter performance requirements like production rate or delivery time constraints. Considering a scenario with several machine lines working simultaneously, varied production rates, capacity, work force and material constraints lead to a high-complex model. Using a central control for the whole system becomes computational intensive and scales badly (van Steen and Tanenbaum, 2017). Furthermore, this leads to the risk of a single point of failure (Deswarte et al., 1991). If additional requirements are imposed like short reaction time or robustness with respect to disturbances, the computation time and further more the reaction time have to satisfy these bounds.

To improve scalability, one way is to reduce the model to those details which might be needed for measuring a meaningful system state or retaining the possibility to react on disturbances. The question arises, how to reduce complexity while keeping a high level of details. This leads to the field of decentralised or distributed systems where the centralised problem is split into subsystems (see Fig. 1). In such a divided scenario, the subsystems are considered as multi-agent systems. Each agent follows a defined goal (objective) and is



**Figure 1:** Centralised (left) and distributed architecture (right) illustrating agents as nodes and communication links with edges between the agents from Dua (2018)

described by certain system properties. The properties or the position of an agent are denoted as system state and the limitations as constraints. In this setting, complexity is traded against communication and coordination. A decentralised setting without communication often reaches only a local optimum in comparison to a global optimal solution of a centralised system. The subsystems are not aware of the state and predictions of other subsystems or an overall system objective and therefore can only compute a local optimum based on their available local information. In many scenarios, a distributed setting, which incorporates communication among subsystems, leads to a better performance, where examples are given for the mentioned scenarios in the following:

In production scenarios, subsystems could be machines or production lines, which are handled by multi-agents implementing local controllers. To determine an efficient production or material flow, priority rules can help to distribute material or production flows in a manufacturing system to retain a steady-state production level (Lödding et al., 2003). Another goal might be a higher efficiency of machines avoiding a long blocking due to the lack of resources (Monteiro et al., 2007). On a more dynamic level, distributed control schemes are utilised also in robotic scenarios: Here, as the entities, i.e. the individual robots are naturally distributed, in the last years the embedded computational ability allows to perform control tasks on the robots itself. The chosen control schemes differ from cooperative schemes, where a common goal is defined (e.g. environment exploration, formation control). In these scenarios, mostly non-cooperative schemes are used, where only collision avoidance is ensured and each robots is following an individual assigned goal. Although cooperative schemes are superior as a global objective is incorporated in the subsystems, more realistic are non-cooperative or partially non-cooperative schemes as not all subsystems do not want to or are not able to follow a global objective due to lack of information or are not able to provide such information.

Considering the problem of individual goals in traffic scenarios, this also affects traffic flows on a macroscopic or traffic scenarios on a microscopic level, i.e. controlling each individual vehicle. The development of autonomous cars is today a widespread area and these systems are naturally decoupled. Hence, the necessity of a short reaction time to disturbances and the reduction of complexity is also one motivation using decentralised or distributed control methods. Here, autonomous connected cars are considered which form a distributed system to share information about a communication network. However, if congestion information can be provided, the connected cars can incorporate this information in their necessary planning calculations. Furthermore, the motivation in developing autonomous and connected cars is mainly driven by financial and safety reasons (Chen and Englund, 2016). For economic reasons, a platooning of trucks gain advantages, as shorter safety distances allows them to save fuel and road space. Again, communication plays an important role. Regarding safety reasons, an autonomous, connected car or truck may reduce failures and mistakes in traffic scenarios in comparison to human drivers (Wierwille et al.). As autonomous cars establish a valid vehicle-to-vehicle-communication (V2V), information can be used to avoid congestions or to reduce the required space (Khaliq et al., 2016).

What all scenarios (production, robotic, traffic) have in common, is the need of a steady communication link to ensure either constraints or update information continuously. As this broadcasted information might revealing an 1:n relation between sender and receiver, an efficient exchange and low additional overhead is mandatory to establish an efficient distributed control.

This thesis focus on the control of distributed systems considering communication effort and quality of solution. As our setting considers non-cooperative scenarios where individual goals are followed, a coupling between the participating subsystems has to be established, which requires a constant communication link due to high dynamical changes in the states of the subsystems as they occur in robotic or traffic scenarios. Since with the usage of wireless communication technology, bandwidth and number of channels are finite. Thus, a reduction of the communication effort is preferable, in particular if urban scenarios with a high density of participants, e.g. vehicles, are considered. Hence, the following research questions are formulated:

**Research question 1:** What is the necessary and sufficient communication overhead to guarantee stability and a certain degree of suboptimality in the distributed non-cooperative setting?

**Research question 2:** Which effects does a relaxation from a fixed optimization order to dynamic priority orders have on stability and suboptimality?

**Research question 3:** How can convergence be achieved in a setting using an occupancy grid as quantisation of states?

Different distributed control methods exists, where Distributed Model Predictive Control (DMPC) has the advantage to allow the direct incorporation of hard constraints in the underlying optimisation control problem (OCP) and is also able to handle non-linear constraints or dynamics as this might be necessary for vehicles or robots. The outline of the thesis is as follows: In Chapter 2 the state of the art of classic non-linear model predictive control (MPC) is introduced in Section 2.1 reflecting also the existing approaches to show stability. As we utilise distributed systems, in Section 2.2 the DMPC scheme is explored in detail with coordination possibilities between the subsystems and the communication methods so far proposed in literature in Section 2.2.2 and 2.2.3 respectively. As one key aspect is coordination and communication, we classify the coordination and communication schemes, which are proposed and present a comparison in terms of communication effort of the state-of-the art methods complemented with a discussion about the classified publications. To highlight another key aspect of the thesis, in Section 2.2.4 the different approaches of quantisation schemes are classified and presented. The chapter is then completed with

a short review about stability in DMPC in Section 2.2.5, which requires additionally assumptions to achieve stability considering the classic approaches, which were utilised for (centralised) MPC. Priority rules to break up a fixed optimisation order of subsystems are introduced in Section 2.2.6, where we focus on approaches, which were appropriate and applied in traffic or robotic scenarios.

In Chapter 3, the DMPC setting is defined for especially robotic and traffic scenario in a non-cooperative manner to outline our problem setting, where we distinguish between a centralised and distributed communication. The results regarding communication effort are presented in Chapter 4 considering methods for differential communication and quantisation reduction schemes. Therein, also a stability approach is presented with a weak assumption derived from traffic rules to achieve convergence for the distributed systems contemplated with a discussion about a sufficient prediction horizon length, which is crucial to reach an equilibrium for most MPC schemes.

To speed up convergence time, the DMPC scheme was extended by utilising priority rules in Chapter 5 with different schemes to illustrate the speed-up in numerical results. Conclusions with discussing advances and limits of the methods are presented in Chapter 6 with an outlook on further work considering other application scenarios, where such a DMPC could be utilised.

**Notation** For the sequence  $x \in \mathbb{N} : x = (1, \dots, n)$  the short notation  $x = [1 : n]$  is used. A sequence of tuples  $(z(1), z(2), \dots, z(N))$  is denoted as  $\mathbf{z} = (z(k))_{k=1}^N$ .  $\|\cdot\|_2$  is used for the euclidean distance in a metric space, while  $\|\cdot\|_\infty$  stands for the infinity norm  $\|x\|_\infty := \max_{i \in [1:n]} |x_i|$ .

# 2

## State of the art

Most control methods distinguish between linear and non-linear dynamic systems. Motivated in Chapter 1, non-linear systems are considered, which depend on the unknown input of other participants and are not directly connected to a certain output. Non-linear constraints result from the properties of the system itself or follow from safety reasons such as a collision avoidance constraint to prevent accidents in traffic scenarios. Evaluating an adaptive and distributed control scheme, the non-linear Distributed Model Predictive Control (DMPC) is a common method used in the control theory field for the last years. Nowadays, the theory of Model Predictive Control (MPC) is well understood, detailed presented in Rawlings et al. (2017); Grüne and Pannek (2017) and is used both in academia and industry (Qin and Badgwell, 2003). As the most important property of (D)MPC is a shifting horizon with a fixed time interval, MPC is also known as Receding Horizon Control (RHC).

We first introduce in Section 2.1 the centralised and classical approach and extend this in Section 2.2 to a distributed setting discussing the different approaches of control schemes in Section 2.2.1, coordination in Section 2.2.2 and information exchange mechanisms in Section 2.2.3, which were made in recent years. Then, after a short summarisation in Section 2.3 we will continue with the development of stability for (D)MPC in Section 2.2.5 and the incorporation of priority rules in Section 2.2.6 before the research questions are drawn.

### 2.1 Model predictive control

The MPC scheme measures the system state and obtains an optimal control to keep the system in a feasible state and drive it to a certain target state. We consider systems with discrete dynamics or continuous system discretised in the form

$$z^+ = f(z(n), u(n)) \quad (1)$$

where  $z(n) \in \mathbb{X}$  is the state,  $u(n) \in \mathbb{U}$  the applied control to reach the next state  $z(n+1)$ , and  $n$  is the time. The system state is measured in discrete time steps, which reveals discrete measuring points of the controlled system.  $\mathbb{X}$  and  $\mathbb{U}$  are called the state and control set, where  $\mathbb{X}, \mathbb{U}$  are measurable and usually subsets with  $\mathbb{X} \subset \mathbb{R}^i, \mathbb{U} \subset \mathbb{R}^j$  with  $i, j \in \mathbb{N}$  denoting the dimensions of the state and control, respectively. As an example, one might assume a 3-dimensional state for a robot (2D plane and orientation) and two dimensions for the control (speed and orientation). The target of the system is defined as  $z^* \in \mathbb{X}$ .

We define the cost function to be minimised as  $\ell(z, u)$  which we assume to be positive semi-definite and  $\ell(z^*, 0) = 0$ . MPC includes the prerequisite to establish the forward shifting optimization, that at any time step  $n$  for a feasible state, there exists a control  $u(n)$  such that the successor state is feasible, i.e.

$$\exists u: f(z(n), u(n)) \in \mathbb{X} \rightarrow z(n+1) \in \mathbb{X}. \quad (2)$$

This property is in literature known as (weak) control-forward-invariance. A control sequence  $\mathbf{u} := (u(0), \dots, u(N-1))$  is calculated to obtain for the system a discretised trajectory  $\mathbf{z} = z(n), \dots, z(n+N)$  over a given horizon  $N \in \mathbb{N}$  by minimising the cost objective at a time instant  $n \in \mathbb{N}_0$  for  $z^0 := z(n)$  with the formulation of the following optimal control problem (OCP)

$$\operatorname{argmin}_{\mathbf{u} \in \mathbb{U}} J(\mathbf{u}; z, z^0) := \sum_{k=0}^{N-1} \ell(z(k), u(k)) \quad (3)$$

w.r.t.

$$\begin{aligned} z(k) &\in \mathbb{X}, & k &\in [0: N] \\ u(k) &\in \mathbb{U}, & k &\in [0: N-1] \\ z(k+1) &= f(z(k), u(k)), & k &\in [0: N-1], \end{aligned}$$

which reveals an optimal control sequence  $\mathbf{u}^* := (u^*(0), \dots, u^*(N-1))$ . The first control step  $u^*(0)$  is taken and applied to the system. These steps are illustrated in Alg. 1, using the OCP from (3) incorporating the feedback measured state, i.e. the feedback, which closes the loop, to compute an optimal control for a trajectory over a finite time horizon:

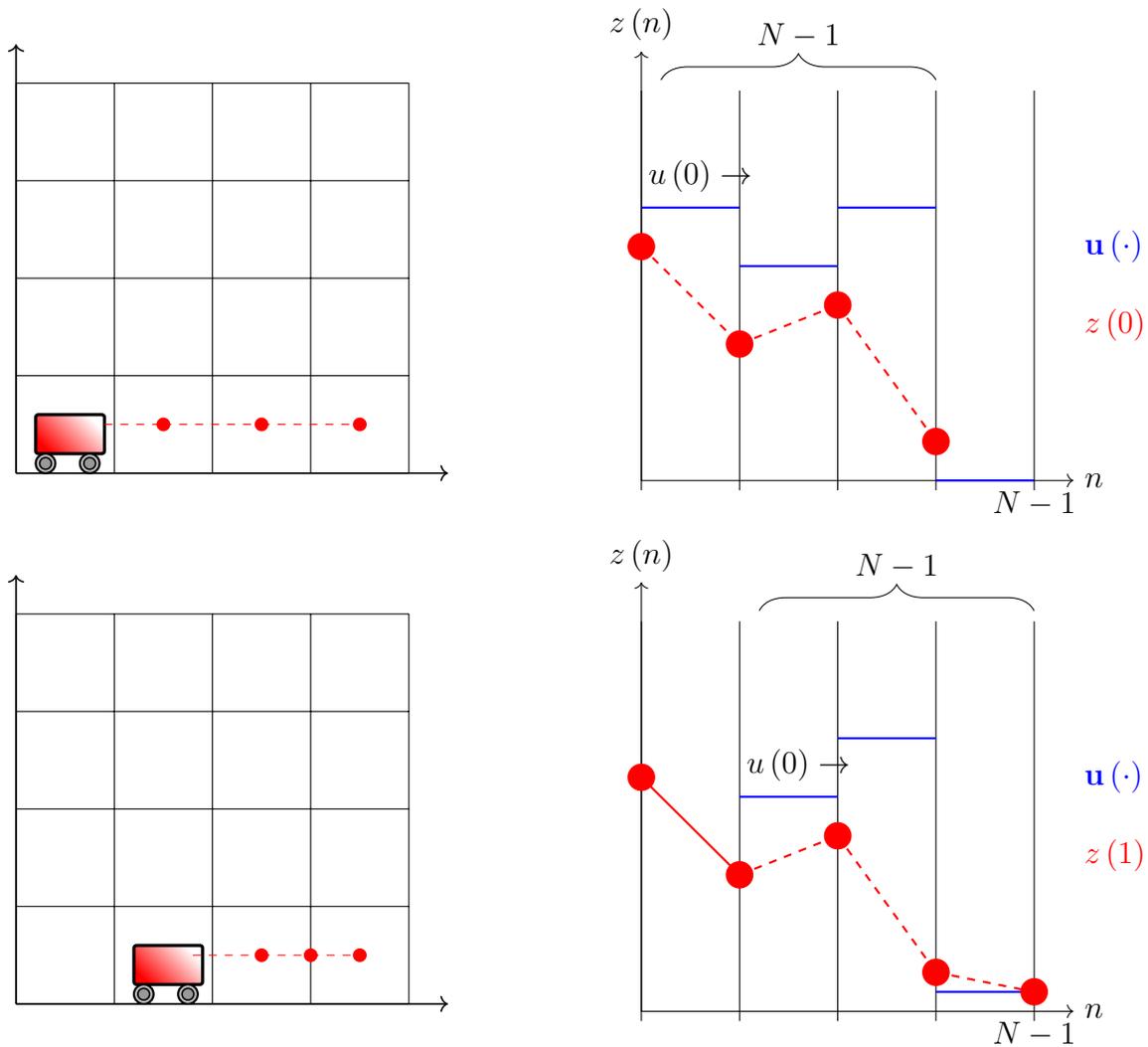
---

#### Algorithm 1 Classical MPC Algorithm

---

**While** target  $z^*$  not reached

1. **Measure** system state  $z(n)$
  2. **Solve** OCP (3) according to cost objective  $\ell$
  3. **Obtain** optimal control sequence  $\mathbf{u}^*$
  4. **Apply**  $u^*(0)$
  5. **Set**  $n := n + 1$
-



**Figure 2:** Illustration of a MPC cycle of two successive time instants ( $n, n + 1$ ) from top to bottom: The left figures illustrates the car with predicted discrete trajectory points with horizon length  $N = 4$ , the right side the predicted trajectory in dashed lines (red) with calculated optimal control, where  $u(0)$  (blue) is applied. The lower figures illustrates the applied control and next state starting with the next optimisation step.

An illustration of two successive steps of the MPC algorithm is also given in Fig. 2: In the left diagrams an example car predicts a discrete finite trajectory with prediction horizon  $N$  and on the right the current and predictive states  $z$  with the obtained control  $u$  are drawn. The measurement is taken at a fixed sampling rate, which executes Alg. 1 in discrete sampling intervals, respectively. In the next time step  $n + 1$  the procedure is repeated over the shifting horizon. The optimal control resembles a static state feedback and closes the control loop. We denote the feedback by  $\mu^N := \mathbf{u}_p^*(0)$  and the resulting closed loop by  $z(n + 1) = f(z(n), \mu^N(z(n)))$ .

To ensure that the system is stable and stays within in the defined feasible state set  $\mathbb{X}$ , we have to define two properties: stability and (sub)optimality. We define stability, if a sequence exists  $\mathbf{u}$  such that we are able to drive a system to a equilibrium state and keep it in that equilibrium region. In the literature, stability is classified in different ways, see

Mayne et al. (2000), Balluchi et al. (2005) or (Grüne and Pannek, 2017, Chapter 6.7). Here, one can distinguish between stability, asymptotic, strong, weak or practical stability. If there exists an equilibrium point  $z^* \in \mathbb{X}$  of a system (1) and a real number  $\epsilon \in \mathbb{R}_{\geq 0}$  exists with  $k > 0, k \in \mathbb{N}_0$  and for an  $N > 0, N \in \mathbb{N}_0$ , if

$$\|f(z(k), u(k)) - z^*\| \leq \epsilon \quad \forall k > N \quad (4)$$

is satisfied, then the system is strongly stable. If this holds for  $\lim_{k \rightarrow \infty} \|z(k) - z^*\| = 0$  for all  $\|z(0)\| < r$  if there exists a  $r \in \mathbb{R}_{\geq 0}$ , then the system is asymptotically stable. Furthermore, if (4) holds for all  $u \in \mathbb{U}$ , then the system is strongly stable. If such a control  $u \in \mathbb{U}$  exists, then the system is defined as weakly stable. For the latter it is challenging to prove stability as it has to be guaranteed that a (not necessarily optimal) control sequence is found, which is feasible.

Stability can be ensured through several methods, which we introduce briefly: For the classic MPC approach, either terminal constraint could be utilised fixing the end point of the trajectory to the equilibrium, bounding functions to limit the objective function or the approach of Lyapunov functions to ensure that the objective function decays in the ongoing time steps.

In Keerthi and Gilbert (1988), endpoint constraints are used to force the system to the target state  $z^*$  at the last value of the prediction horizon sequence. Chen and Allgöwer (1998) impose a terminal cost function to implement the MPC algorithm for non-linear systems, which adds a penalty to terminal states outside the equilibrium state. Additionally, to relax the terminal constraint as a fixed end point, a terminal set around the given equilibrium is defined to guarantee asymptotic stability and let switch the control to a Lyapunov-based controller, which keeps the system inside this region (Scokaert et al., 1999). The authors compare the methods of utilising terminal end points or proposing a terminal set. The latter has the advantage that the end point condition can be relaxed and it has to be shown that the local controller is capable to keep the system state inside this terminal set.

Combining both stability approaches, terminal sets and terminal costs are presented in Nicolao et al. (1998). There, a non-linear controller is implemented using a terminal set and terminal costs, in which the terminal costs are based on the effort of the linearised control law to keep the system inside the terminal set. Michalska and Mayne (1993) implement this approach as a dual-mode controller, where outside the terminal set a non-linear control law is applied to have an receding online feedback control, while on the inside region a linear feedback controller is applied. A dynamic terminal set to relax the feasibility condition, which adapts the region in every time instant, is presented in Mayne and Falugi (2016).

Another approach is the suboptimality: Grüne and Rantzer (2008) relaxes the necessity of terminal sets or costs by examining the suboptimality of the MPC controller. As we deal with finite horizon controllers, suboptimality provides a comparison between the performance of the finite optimal control problem with the applied controller and the original optimal control problem considering an infinite horizon objective function. Suboptimality is the indicator to compare the resulting feedback of an implemented controller regarding a finite cost

functional to an infinite horizon cost functional. Stability without terminal sets or costs are extensively discussed and reviewed in Grüne and Pannek (2017): To show stability, bounding functions in the form of decaying  $\beta$ -functions with  $\beta(r, n) \in \mathcal{KL}$  are used which provides an upper bound for the system state, see (Grüne and Pannek, 2017, Def. 2.14). The set  $\mathcal{KL}$  is a combination of two sets  $\mathcal{K}$  and  $\mathcal{L}$ , where  $\mathcal{K}$  describes the continuous functions  $\alpha: \mathbb{R}_0^+ \rightarrow \mathbb{R}_0^+$  which are strictly increasing with  $\alpha(0) = 0$ . The set  $\mathcal{L}$  describes the continuous functions  $\delta: \mathbb{R}_0^+ \rightarrow \mathbb{R}_0^+$  which are strictly decreasing with  $\lim_{n \rightarrow \infty} \delta(t) = 0$ .  $\alpha(\cdot)$  and  $\delta(\cdot)$  represents bounding functions, which set the boundary for our system function. Then  $\mathcal{KL}$  denotes the set of all  $\beta(r, n)$  and we define  $\beta(r, n)$  as continuous and  $\beta(r, \cdot) \in \mathcal{L}$  for all  $r \geq 0, n > 0$  and  $\beta(\cdot, n) \in \mathcal{K}$ . If there exists  $\beta \in \mathcal{KL}$ , such that the objective of the system is below  $\beta(r, n)$ , then the system is asymptotically stable.

Another approach utilises Lyapunov functions to ensure stability and allows to establish a bound on the degree of suboptimality for an OCP over a fixed horizon length as stated in (3) with  $J^N(z^0, u^*)$  for  $N$  to denote the finite horizon length  $N$ . A Lyapunov function  $V^N(z^0) := J^N(z^0, u^*)$  over a finite horizon  $N$  is defined as  $V^N: S \rightarrow \mathbb{R}, S \subset \mathbb{X}$  with  $z^0$  as initial prediction. The Lyapunov function can be described as the loss of energy or costs in each time step of the whole system which ensures stability and to be bounded by a set of function, which are defined by  $\mathcal{K}_\infty$ : The set  $\mathcal{K}_\infty$  describes the continuous functions  $\alpha: \mathbb{R}_0^+ \rightarrow \mathbb{R}_0^+$  which are strictly increasing, unbounded and satisfying  $\alpha(0) = 0$ . More formally, we require functions  $\alpha_1, \alpha_2, \alpha_3 \in \mathcal{K}_\infty$  such that

$$\begin{aligned} \alpha_1(\|z(n)\|) &\leq V^N(z(n)) \leq \alpha_2(\|z(n)\|), \\ \alpha_3(\|z(n)\|) &\leq \ell(z(n), \mu^N(z(n))). \end{aligned} \quad (5)$$

Additionally, we require the Lyapunov inequality

$$V^N(z(n)) \geq \alpha \ell(z(n), \mu^N(z(n))) + V^N(z(n+1)) \quad (6)$$

to hold for  $n \in \mathbb{N}$  and  $\alpha \in (0, 1]$ .

If these conditions (5), (6) hold, the closed loop solution  $z(n)$  behaves like a trajectory of an asymptotically stable system. Moreover, we obtain the performance bound in comparison to the finite controller and infinite horizon as

$$J^\infty(z(0), \mu^N(z(0))) \leq \frac{V^N(z(0))}{\alpha} \leq \frac{V^\infty(z(0))}{\alpha},$$

where  $\alpha \in (0, 1]$  represents a degree of suboptimality of the controller. In contrast to the usage of  $\beta$ -functions, the Lyapunov property does not include initialization of the states as only the loss of energy assures stability. As we are considering a system with feasible initial state  $z(0) = z^0 \in \mathbb{X}$ , we are allowed to utilise the definition of  $\beta$ -functions as upper boundary for stability. Furthermore, this approach using the property of stability and suboptimality allows for showing stability without imposing terminal costs or constraints.

Nevertheless, the Lyapunov theorem requires a monotonous decrease of the optimal value function, which might be infeasible for some system, especially if external disturbances have to be considered. Hence, further extensions are made: One approach is allow for a temporary increase of the optimal value function by the definition of a finite interval until the optimal value function again decreases monotonously. Beyond the end of the chosen interval, the value function will satisfy the Lyapunov condition again (Pannek and Worthmann, 2014, Section 3).

Another approach is to weaken the requirement of terminal costs or constraints: the (asymptotic) stability condition is relaxed to *practical stability* in Grimm et al. (2005), where practical stability are used in terms that not an equilibrium point was reached but a bounded region, towards the system is steered and kept. Asymptotic bounds on the value functions guarantee that stability can be achieved with finite prediction horizon length: Grüne and Rantzer (2008) abstract the results to achieve stability by explicit bounds for suboptimality. In Grüne and Pannek (2017) the computation of suboptimality is obtained by using information of the controllability and the bounds of the optimal value function, which requires that such a bound can be computed explicitly.

To tackle the problem of possibly (infinite-horizon) asymptotic stability, Anderson et al. (2018) utilises a finite-horizon convergence approach using a definition of extended controllable sets, which allows the system to steer to a terminal set in finite time. The number of taken steps here is bounded by the ratio of optimal value function and minimal distance to the target region. Utilising suboptimality linear parameter-varying (LPV) as a time-dependent approach offers an alternative via introducing a time-dependent scheduling signal, which connects the dependency between input and output and abstracts the non-linearity of the model to a linear approximation. The technique called *LPV embedding* utilises a scheduling map, which is state dependent and linearise the system depending on the current state and time. Nevertheless, the drawback comes from the lack of an abstract embedding technique, as this cannot be generalised.

Including system properties as dissipativity of the overall system allows to state the model as a Linear Matrix Inequality (LMI) (Cisneros and Werner, 2018). This can be solved offline to guarantee stability a priori before running the underlying system. Regarding the loss of energy from Lyapunov, Höger and Grüne (2019) illustrates for the properties detectability (see Grimm et al. (2005)) and strict dissipativity the connection between both properties and how strict dissipativity helps to derive conditions for detectability.

While the results here were given for non-linear MPC in a most central implementation, in the next section the distributed model predictive control scheme (DMPC) is considered, as in most cases a coordination mechanism is required unless the systems could be fully decoupled and controlled in a decentralised manner. Extending the current to a distributed setting, we present the DMPC scheme in the ensuing Section 2.2.

## 2.2 Distributed model predictive control

Solving a centralised MPC problem may require intensive computation. To reduce complexity, one approach is to split up the system into subsystems. In production scenarios we could split a central controller of a production system into controllers which handle one machine or one production line only. In contrast to the latter, a distributed setting may naturally occur if we consider autonomous cars as subsystems using a shared resource. These subsystems have the same properties but can have different initialisations or targets.

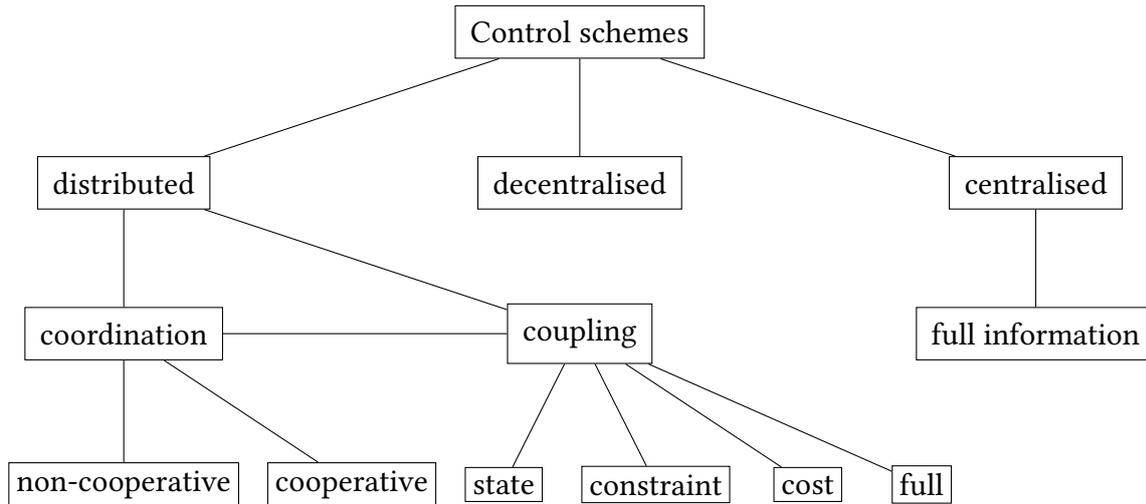
Concerning a distributed or decentralised system, several implementations can be found in the literature. First, the decentralised MPC splits up the central problem in subproblems with same properties (Venkat et al., 2005). Each subproblem performs its own local optimization without any connection or communication to other subsystems. The subsystems are coupled either by constraints or state variables. These couplings are treated as disturbances which the subsystems have to incorporate.

Communication is not considered in decentralised schemes (Venkat et al., 2005). As each controller is optimising a local optimal solution, it can be locked to a local optimum and the global optimum of the centralised setting is not reached. Due to the lack of communication, decentralised MPC is mostly inappropriate for subproblems with interaction between each other. Therefore, the setting is often extended to Distributed Model Predictive Control which incorporates communication enable to find a closer solution to the central optimal one.

In literature, DMPC was first proposed in Jia and Krogh (2001) and Camponogara et al. (2002) without any central controller unit for linear, time invariant systems. The overall centralised problem is decomposed into several small problems with the same properties. Each subsystem performs its own optimisation depending on its local state measurement and informations from other subsystems. This can be, e.g. active communicated predictions of the states or coupled constraints among the systems. Therefore, we can consider a multi-agent system where each agent represents a subsystem implementing a local controller.

### 2.2.1 Control schemes

We distinguish for control schemes between non-cooperative and cooperative schemes of DMPC. To give the reader an insight of the classification, this is depicted in Fig. 3. A comprehensive survey considering coordination and communication is presented in Maestre and Negenborn (2014), but due to the emphasis on communication and the effort that occurs, we modify the classification as we focus here on the reduction of communication. In the non-cooperative control scheme, the controllers of each subsystem optimise their local problem with respect to coupled dynamics or constraints with other subsystems. As those results are only locally optimal for each controller, the overall global optimal solution for the whole system may be not reached as the controller are unaware of the objectives of the other subsystems. Often, only a Nash equilibrium is reached (Venkat et al., 2005). To overcome this problem from a non-cooperative setting, cooperative schemes have to obtain information about the global system (Stewart et al., 2010). This could either be specific states of the subsystems or also the objective function. This global objective is known by



**Figure 3:** Classification of control schemes

each subsystem and has to be incorporated at least partially to penalise control actions, which hinder the system from achieving the global equilibrium. With large-scale systems the coordination effort might be time-consuming if each system broadcasts its prediction to all other subsystems. Hierarchical structures could help to restrict the communication and coordination effort for large scale systems. A broad review about such architectures is presented in Scattolini (2009).

Both cooperative and non-cooperative schemes need a mechanism for coordination. We first present an overview about coordination schemes, which are widely used in DMPC. At next, we focus on the communication effort, especially between agents as with high-dynamic systems as mobile robots a higher load of information has to be exchanged. Note that we use agents and subsystems as synonyms, hence each subsystem is also acting as an agent with its own state, rules and knowledge.

### 2.2.2 Coordination

Using a distributed setting, some coordination mechanism has to be established between the systems. The question arises, which mechanism of coupling between the subsystems is appropriate and how this should be established. In many scenarios, the subsystems are coupled either via constraints, states or objective function (costs).

As we focus on the communication part, we choose a categorization where the coupling is defined via the used knowledge of other agents, which induces a necessity of a form of communication. We divide the mentioned literature in their coupling methods which is either state-based, constrained-based or coupled via objective or cost function. We denote with  $|z_{1,\dots,Q}^N|$  that the full state, constraint or costs over the horizon  $N$  to a defined neighbored set  $Q$  is transmitted. For example, if an objective function  $J_p^N$  of a system  $p$  depends on state variables of a neighbored set of systems  $z_{1,\dots,Q}$ , we define this coupling as state- and

cost-based as first it has to collect the neighbour states  $z_{1,\dots,p-1,p+1,\dots,Q}$  and second, those are used in the objective function. Furthermore we classify the algorithms with regards to the number of communication steps taken in each iteration and what is communicated (x-step method for taking x communication iterations).

The categorization is motivated by the issue that spreading few intermediate states in several steps may induce lower communication costs than spreading the whole trajectory over the full horizon in one iteration step. Furthermore, one iteration step is defined as one time step between  $n$  and  $n + 1$ . We will separate the survey in state- and cost-coupled, state- and constraint-coupled, constraint- and cost-coupled and full-coupled.

### State- and cost-coupled

Considering at first state-and cost-coupled distributed systems, in Rawlings and Stewart (2008) a DMPC coupled by states and objective function in a chemical control chain is implemented. The system is modelled as a directed graph whereupon the states are coupled with the predecessor and the costs with the successor. Here the communication is bounded by the input and downstream flow of the depending neighbours in a chemical process, where the controller has to obey the state trajectories of the predecessor and communicate its input trajectory to its downstream. Therefore we categorize this as a two-step-method.

Stewart et al. (2010) uses a state exchange between the systems which combines the model states which are exchanged between two subsystem and a coupled cost function which includes the control values in the terminal costs. This approach is extended in Stewart et al. (2011) for non-linear distributed plants, where a non-linear distributed non-convex optimization is done via a gradient projection. As the agents exchange their next gradient step, the communication load is at the worst case m-times for m gradients as the exchange is repeated until the overall next step is smaller than the current one.

Liu et al. (2008) use a two-tier hierarchical DMPC controller system, where the lower tier stabilise the systems and monitors the measurements of the system. The upper tier is a network system, which coordinates the measurements from the other connected controllers. The model is also using differences between state measurements to retain stability of the overall network. A similar approach is used by Farina et al. (2015), who implement this two-tier-model for a cooperated control of robots. Each robot has the full trajectory available from its neighbours. The communication is bounded to direct neighbours and the group leader. The different scenarios (collision avoidance, formation control, coverage and containment control) use a two-step-method in communication by sending and receiving the predictions to its direct neighbours. Farina et al. (2014) implement a DMPC scheme for a continuous time system, which is coupled via states and costs of a fully connected system.

Jia and Krogh (2002) utilise the DMPC approach by exchanging coupled states which are bounded to handle uncertainties. Maestre et al. (2011) use a 7-step-method where at first negotiations are made between agents, which evaluate the influences of their neighbours. At a second step, the input trajectories are exchanged and last the differential costs (named as  $\Delta\ell$ ) are send to the affected neighbours, which also updated their input trajectories again. Scheu et al. (2010) use a partial goal-interaction method which uses states as interaction variables

for incorporation of other subsystems and interaction sensitivities. These sensitivities  $\phi_{1,\dots,P}$  are defined as a quotient of the derivatives of objective function and control to calculate a drift of a current subsystem. As for each iteration the interaction sensitivities are exchanged as the trajectories, we categorise this as a two-step method. Al-Gherwi et al. (2013) use a robust DMPC approach to divide a linear time-varying system into separated subproblems. They propose two schemes, which are either cooperative and obey to a global objective function or a Nash-based optimization, which is locally optimal between two neighbored agents. The functions estimation are separated in an online and offline computation. In every iteration all controllers exchange their states and initial estimated states in each optimization step. After that the control is applied, the states are exchanged and next iteration starts.

A discrete approach with a divided space of an intersection into a grid with separate cells is used by Von Eichhorn et al. (2013). In their model, the MPC algorithm is assisting the driver of a car and predicts the possible goals via a cost-to-go-gradient functions which measures the costs to the possible targets to weight the likelihood which target is preferred. The complexity is reduced by the existence of only two or three targets if they are considering a 4x4-intersection and hence, the count of possible targets is bounded naturally. This algorithm is further used for collision prediction which is implemented as a minimal threshold for a time to brake. As communication is not executed directly, only measurements of states are passively taken and incorporated into the cost function. Rochefort et al. (2014) describe the exchange of predicted states for a cooperative control approach of vehicles by including them directly in the cost function. However, the communication load is quite high as the full prediction has to be broadcast between the participating vehicles. The cost function is also combining various missions target which can be coupled and tuned via weight functions. Altmüller and Grüne (2012) use distributed control for partial differential equations using the heat equation as an example. The control is implemented as a directly to the derived state, while the dedicated cost function penalises the deviation from state and control.

Dunbar (2007a, 2006) proposes a distributed scheme coupled via the system dynamics modelled in a directed graph where each agent represents a node and communication is implemented via edges. They apply this on a Van der Pol oscillators setting. Furthermore, similar to Rawlings and Stewart (2008), the communication between neighbours is divided in an upstream and downstream channel which results in a two-step algorithm for each iteration. In Dunbar (2007b) this is quite opposite to Dunbar (2007a, 2006): Here, the system dynamics was decoupled and in contrast to the previous work, the cost function summarises the local states with the states of the neighbour agents. As the coupled systems should not deviate in their predicted trajectories too much from the system trajectory, only the affected systems are updated.

### **State- and constrained-coupled**

Jia and Krogh (2001) propose a DMPC-Stability constrained scheme with a one-step state prediction exchange of each subsystem, which are incorporated in the constraints. The communication method of Stewart et al. (2010) is also utilised by Camponogara et al. (2002). Here, the approach is called stability-constrained MPC according to Jia and Krogh (2001),

where the constraints are coupled via the prediction of the agents. The communication effort is not further examined, as only the system coupling is defined based on tight state constraints. Thus, the predicted states according to the MPC prediction scheme have to be communicated fully in each time instant.

In Stewart et al. (2010), this approach is used with optional coupled constraints  $C_{1,\dots,P}$  where the model states and control inputs are also coupled in the objective function. Savorgnan et al. (2011) couple the in- and outputs and evaluate them in each cost function of the coupled subsystems whereas the connections between the subsystems are dynamic. The number of exchanged variables is reduced due to linear combination of the state variables via basis functions. Camponogara et al. (2009) and Camponogara and Oliveira (2009) apply the DMPC scheme with coupling the control output in a network traffic scenario modelled as a linear-dynamic network where each junction is a DMPC node. Agents coupled via their constraints iterate serialised until convergence is reached. In Camponogara et al. (2002, 2009) and Camponogara and Scherer (2011) the distinction is made between local, neighbour and remote variables, which are set by the local agent or neighbored systems. Camponogara and de Lima (2012) illustrate a gradient descent algorithm as a distributed subgradient method where the agents coordination is done via states denoted as neighbour variables and control values as subgradient approximations to gain information about the neighbourhood.

Savorgnan et al. (2011) present a parallel Multiple-Shooting-Method, where the connection between subsystems are constant and the coupling is realised via directed input-output connections. Doan et al. (2011) utilise a communication approach which uses coupled constraints, where the communication radius is bounded to the direct neighbours. Their state are exchanged separately in two-x steps: Main variables about the state and intermediate variables  $\gamma_{1,\dots,Q}$  depending on the connections between the agents, which are taken iteratively while performing the optimization and thus, as the iterative optimisation might take more steps, at minimum two steps have to be considered. Furthermore, this classifies as a two-step algorithm as they are providing the states and intermediate variables, which describes the distance to the vehicles target used as constraints for the OCP. Trodden and Richards (2010) use a tube-based DMPC which follows a forth going tube modelling a range of time-dependant trajectories. The coupling is done via constraints where at first all agents start with a feasible solution and obtain their new plan in every iteration from the coupled subsystems if their plan was updated. Mercangöz and Doyle (2007) apply the DMPC approach as a fused state where each node collects the states from its neighbours. This fused state is used in the cost function and after receiving the neighbored control signals, each node optimises again until a certain threshold is reached. Therefore the communication rate depends on the amount of optimizations. Dunbar (2006) uses a directed graph with up- and downstreams like Rawlings and Stewart (2008). The states of the agents are coupled and incorporated via constraints which ensure, that the agents do not deviate from a feasible state and control threshold. The approach from Venkat et al. (2005) is based on Jia and Krogh (2002) and Camponogara et al. (2002) with the extension of approaches to a feasible-cooperation-based DMPC to tackle the lack of convergence in their plant model when a

decentralised control scheme is applied. Their approach uses an overall objective function with coupled states. In contrast to the aforementioned approaches the communication load rises up to  $n$ -to- $n$  in the worst-case with four times interchange of states. Nevertheless, they show that in one iteration step the system could be stabilised with higher communication load as all states of all subsystems  $|z_{1,\dots,P}|$  have to be available to all subsystems.

A very different approach is proposed by Heidarinejad et al. (2011a), where the sampling rate of the states can differ for each subsystem. Each Lyapunov-based controller submits its control trajectory to the first controller which checks the received solutions for feasibility. Non-feasible trajectories are rejected. Noise in the communication channel is recognised in this approach, hence, the first controller assumes a zero input for the affected controller. Pannek (2013) developed a parallel covering algorithm based on the approach from Richards and How (2004b). As the algorithm evaluate the dependencies between agents to maximise the parallelization rate between the agents. At first, a solution is computed, which is spreaded to all agents and in a second step a priority rule is obtained to get an execution order. Then, an updated control sequence is sent to the other agents with updated constraints imposed by the other agents. The priority rule gives an hierarchy which determines a dependency which agents can optimise in parallel. Grüne (2012) also uses the approach from Richards and How (2004b) but assume that at any optimising step the predicted states  $z_i(1), \dots, z_i(n)$  from each agent  $i \in P$  are transmitted to each other subsystems sequentially. In contrast to Pannek (2013), this approach is handling the optimization step of the agents sequentially and therefore only one communication step is needed at the expense of non-parallel execution.

### **Constraint- and cost-coupled**

Talukdar et al. (2005) implement the DMPC scheme for preventing energy grids from hierarchical failures. Here, the control inputs of all connected neighbours, which are distinguished in different regions, are incorporated in the objective function. Therefore, we can conclude, that this scheme is coupled via constraints and costs. Camponogara and Scherer (2011) use the approach from Camponogara and Oliveira (2009) illustrating a distributed gradient descent algorithm and a control barrier function. Gros (2014) proposes a distributed non-linear MPC (NMPC) algorithm based on real-time iterations as a wind farm control. Utilising a direct-multiple-shooting method, the constraints of each wind turbine is coupled and updated by a central controller as a central communication instance. This is also done in two steps as the subsystems sends their costs and current power rate to the central controller which updates then the controllers with a new gradient step. Richards and How (2004a) use for cooperated uninhabited vehicles a DMPC scheme based only on coupled constraints, which are broadcasted to each subsystem. Alessio and Bemporad (2007) uses a distributed DMPC set which is partially coupled via the process model. Stabilization is guaranteed via the degree of coupling the model states. This leads to higher communication loads as more states has to be exchanged.

### Full-coupled

Rantzer (2009) uses dual decomposition with decoupling the systems via Lagrange multipliers. Each subsystem incorporates in its cost function the prediction of the influence of other agents to itself and the received messages from the other agents about their states. These informations are aggregated in the local cost function. As each subsystem depends on the full state for solving the decomposed problem, full state communication is required in each time instant. In Maestre et al. (2010) this scheme was considered to be handled either with a market instance, which share only aggregated information or estimation based on an applied Kalman filter to attenuate the full state communication.

Müller et al. (2012) establish information exchange based on coupled constraints and objective function. The whole trajectory is transmitted once to all neighbours. Then, the next state after re-optimization to its neighbours is sent, which tests the result and return that back. After that, a flag is transmitted if the own optimization or the re-optimization is applied. This results in four steps for communication. Tightening the coupling to only affected neighbours reduces the communication steps to two steps.

### Summary state of the art of coordination

What is common for all approaches that the coupling is established via two properties as for the OCP the information has to be included either in costs or constraints. Most methods use either the advances of slow dynamics in the overall systems or slow-changing communication links to define a neighbourhood to avoid a broadcast to all subsystems for the whole system. In Table 1 the used methods are illustrated: The white bullets stands for an optional incorporation of this features as the author uses this in some purposes.

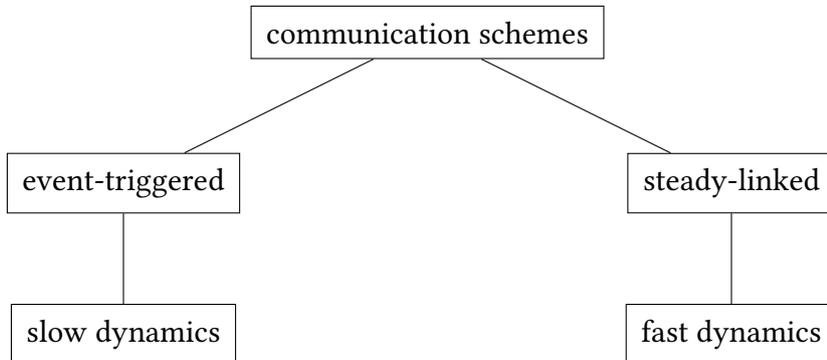
In the next section, we will examine communication reduction methods: The first are based on event-triggering approaches, i.e. that systems only transmit information if a certain condition is matched. The second ones are based on quantisation approaches, where steadily transmitted communication effort is reduced, which is also the main concern of our setting.

### 2.2.3 Communication

As for most distributed MPC scenarios with couplings a coordination between the subsystems is immanent to achieve convergence (Venkat et al., 2005), one focus of the thesis is subject to the reduction of communication, especially when wireless communication is considered. As the number of available communication channels is limited, especially in urban scenarios with a high density of nodes the packet delay is still a problem, most investigated in vehicular ad-hoc networks, which have to guarantee a finite delay of arrival time of packets (Karabulut et al., 2018; Khaliq et al., 2017). To reduce the communication effort on links between coupled or state-dependent nodes, event-based optimal control is utilised in Grüne and Müller (2009) for a non-linear model and applied for a tank by triggering the control if certain state regions by the partitioned state-space are crossed. For an overview of the choice of methods, see Fig. 4.

**Table 1:** Categorisation of coordination in DMPC. White bullets denote optional incorporation if not used over the full time evolution.  $\mathbf{u}_{1,\dots,Q}^1$  denotes that the first control value ( $\mathbf{u}(1)$ ) is transmitted from one agent  $z_p$  to neighbours  $z_{1,\dots,p-1,p+1,\dots,Q} \in Q$ .

Author	State	Constraint	Costs	Iter.	Comm. rate/iter.
Rawlings and Stewart (2008)	•	-	•	2	$ z_{i-1}^N + \mathbf{u}_{i+1}^N $
Dunbar (2007b)	•	-	•	2	$ z_{1,\dots,Q}^1 \cdot n_{\max} \mathbf{u}_{1,\dots,Q} $
Dunbar (2007a)	•	-	•	2	$ \mathbf{u}_{1,\dots,Q}^N $
Farina et al. (2014)	•	-	•	1	$ z_{1,\dots,Q}^N + \mathbf{u}_{1,\dots,Q}^N $
Farina et al. (2015)	•	-	•	2	$ z_{i,j}^{n+N+1} \cdot Q $
Liu et al. (2008)	•	-	•	1	$ z_{1,\dots,P}^1 $
Rantzer (2009)	•	-	•	1	$ z_{1,\dots,P} $
Rochefort et al. (2014)	•	-	•	1	$ \mathbf{u}_{1,\dots,P}^N $
Altmüller and Grüne (2012)	•	-	•	1	$ \mathbf{u}_{1,\dots,P}^N $
Scheu et al. (2010)	•	-	•	2	$ \phi_{1,\dots,P} + z_{1,\dots,P}^N $
Stewart et al. (2010)	•	○	•	1	$ z_{1,\dots,P}^N + \mathbf{u}_{1,\dots,P}^N + C_{1,\dots,P} $
Stewart et al. (2011)	•	-	•	$M$	$ z_{1,\dots,P}^N + \theta_{1,\dots,P} $
Maestre et al. (2011)	•	-	•	7	$ 2 \cdot \mathbf{u}^N \cdot Q + \Delta \ell^1 + \Delta \mathbf{u}^1 $
Al-Gherwi et al. (2013)	•	-	•	$ n_{\max} $	$ \mathbf{u}_{1,\dots,P}^1 \cdot n_{\max} + z_{1,\dots,P} $
Von Eichhorn et al. (2013)	•	-	•	1	$ z_{1,\dots,P}^N  +  $
Mercangöz and Doyle (2007)	•	-	•	$1 \geq \infty$	$ P \cdot \mathbf{u}_{1,\dots,Q}^1 $
Venkat et al. (2005)	•	-	•	1	$ 4P \cdot (z_{1,\dots,P}^N) $
Dunbar (2006)	•	•	-	1	$z^N f_{1,\dots,Q}$
Trodden and Richards (2010)	•	•	-	2	$ \mathbf{u}_{1,\dots,Q}^{N, \text{updated}} $
Camponogara et al. (2009)	•	•	-	1	$ \mathbf{u}_{1,\dots,Q}^N $
Camponogara and Oliveira (2009)	•	•	-	1	$ \mathbf{u}_{1,\dots,Q}^N $
Camponogara and de Lima (2012)	•	•	-	2	$ \mathbf{u}_{1,\dots,Q}^N + \xi_{1,\dots,Q} $
Doan et al. (2011)	•	•	-	2	$ \gamma_{1,\dots,Q} + \mathbf{u}_{1,\dots,Q} + z_{1,\dots,Q} $
Savorgnan et al. (2011)	•	•	-	1	$ C_{1,\dots,Q} + Z_{1,\dots,Q} $
Jia and Krogh (2001)	•	•	-	1	$ x_{1,\dots,P}^1 $
Jia and Krogh (2002)	•	•	-	1	$ z_{1,\dots,P} $
Camponogara et al. (2002)	•	•	-	1	$ x_{1,\dots,P}^1 $
Richards and How (2004a)	•	•	-	1	$ \mathbf{u}_{1,\dots,P}^N $
Alessio and Bemporad (2007)	•	•	-	1	$ z_{1,\dots,P}^1 \cdot Q $
Heidarinejad et al. (2011a)	•	•	-	1	$ \mathbf{u}_{1,\dots,P}^N $
Pannek (2013)	•	•	-	1	$ 2 \cdot \mathbf{u}_{1,\dots,P}^N $
Talukdar et al. (2005)	-	•	•	$\leq 3$	$ \mathbf{u}_{1,\dots,Q}^1 $
Camponogara and Scherer (2011)	-	•	•	1	$ \mathbf{u}_{1,\dots,Q}^N $
Gros (2014)	-	•	•	2	$ 2 \cdot z_{1,\dots,P}^1 $
Müller et al. (2012)	•	•	•	4	$ 2 \cdot z_{1,P}^N + \ell_{1,\dots,P} $



**Figure 4:** Classification of communication schemes

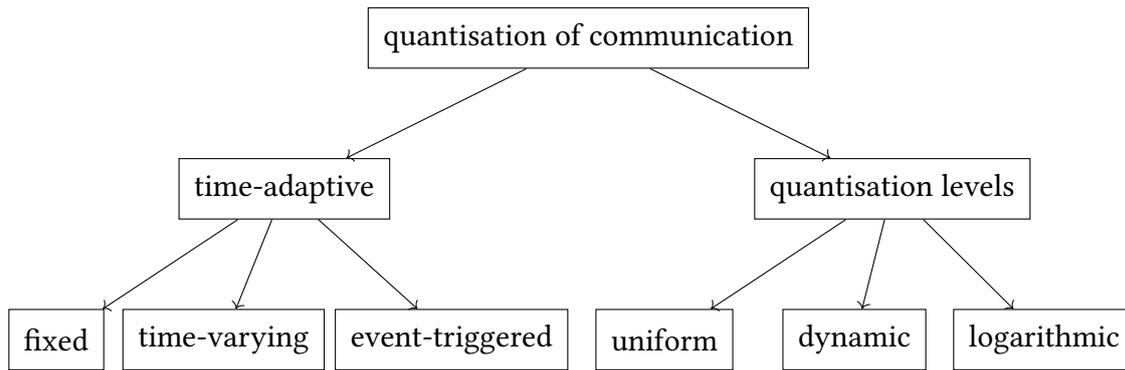
Other approaches use event-based mechanisms (Varutti et al., 2009), that use a variation of the sampled time interval, when a recalculation is necessary. Utilising a threshold defined on the system state ensures a reduced control effort when the system deviates from the defined state region (Varutti et al., 2010). Here, the intervals to recalculate the control are adapted to occurring events when the system leaves the defined stability region, while practical stability is guaranteed by a bounded cost functional, which restricts the control effort to converge to the stability region. Eqtami et al. (2011) provide a different event-triggering rule, where the open-loop prediction is used as long as the predicted states are fitting along with the measured states of the subsystems. Including packet losses that may occur by using wireless networks, Wang et al. (2015) derive conditions for maximum packet losses and a minimum number of triggering events with subject to energy consumption for linear networks.

Utilising a stochastic MPC setting, Quevedo et al. (2015) use a stochastic cost function, which incorporates the dropouts of the communication channel in the optimisation problem via a history-dependant feedback policy between a controller and an actuator, which impose the control signals to the system. Here, the sequence of communication dropouts is known to the controller and a buffer on the actuator side is used to store the input sequence to apply the next input values from the received sequence if a dropout occurs. The communication reduction is realised by using a larger recalculation interval when the optimisation restarts. Hence, the recalculation interval has to be smaller or equal than the prediction horizon length. While those described scenarios based on event-triggered processes mostly handle plants, chemical reactors or other systems with a slow varying dynamics, a large varying dynamic (e.g. mobile robots or autonomous vehicles) may induce many triggering events, which reduce the gain of communication reduction.

Therefore, as a steady communication load due to many state changes is often inevitable, our focus is on the reduction of the communication on the links among the subsystems itself. Furthermore, also approaches, which consider neighbourhoods as in Dunbar (2007a) or Farina et al. (2014) are not considered, as most dynamic vehicles have to use wireless resources to communicate, and hence, the number of usable channels is limited. As the vehicles are most in a dense setting, a unicast would not reduce the availability of these resources, as due to the nature of wireless communication protocols they are based on protocols to avoid collisions (CSMA/CA, (Colvin, 1983)) and therefore, each communication on the same channel has to be considered by the CSMA/CA protocol and hence, communication bandwidth is reduced. One efficient possibility with both low computational effort and easy to implement are quantisation methods, which are discussed in the next section.

### 2.2.4 Quantisation of communication

For our purpose, we classify quantisation methods into time-adaptive and quantisation-level based, i.e. for the first the adaptation of the quantisation over time and for the latter the quantisation methods for the values. For the first, we distinguish into three different types: Fixed, time-varying and event-triggered quantisation. For the latter we use the classification

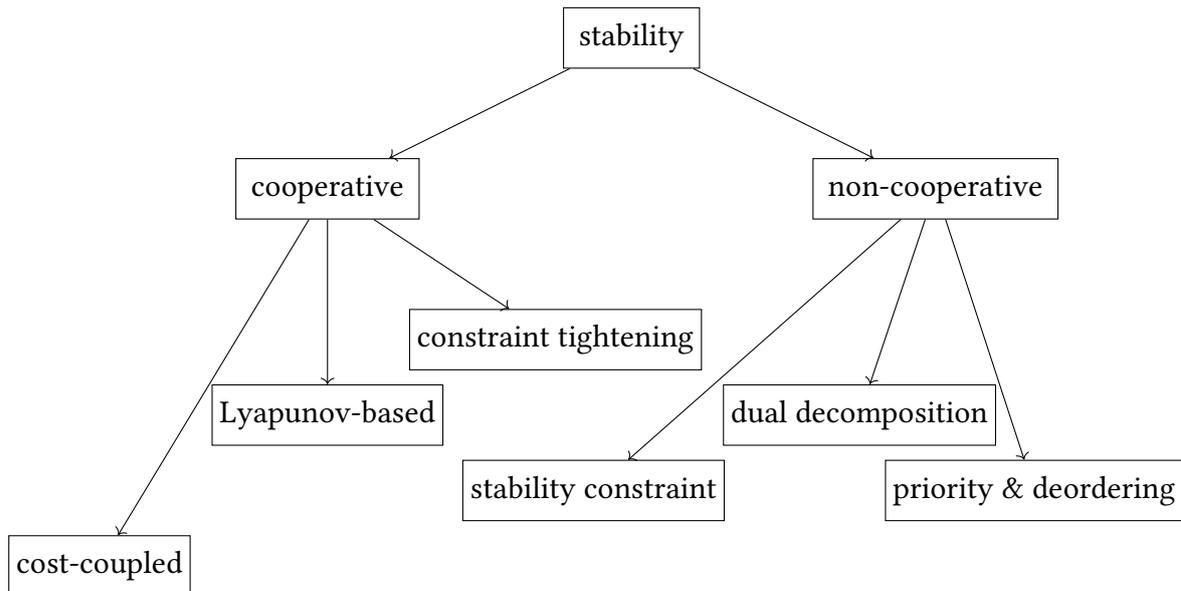


**Figure 5:** Classification of the quantisation approaches

of uniform, dynamic and logarithmic quantifiers. This classification scheme is illustrated in Fig. 5. Goodwin et al. (2004) utilise a fixed quantisation scheme with uniform quantisation levels between a centralised controller and several plants or agents. The authors are using an incremental update scheme and a fixed control set, which is a priori known to the actuator and reduces the communication effort further. Nair et al. (2006) use fixed quantisers with time-varying sampling time to derive the suboptimality performance regarding the available coding size. The scheme is bidirectional as the quantisation is implemented between plant and controller (measurement) and vice versa (control input). A distributed consensus approach considering a quantisation on fixed levels is illustrated in Kashyap et al. (2007). Here, the quantised consensus is to obtain the average over several multi-agents with assigned integer value without changing the sum of all values. Convergence conditions are derived for fully-connected networks.

Regarding time-varying quantisation, El Mongi Ben Gaid and Çela (2006) evaluate the relation between the quantisation precision of communicated data and sampling step size. In each sampling step, the quantisation precision is chosen based on a pre-calculated control set given by the designer. A quantisation considering uniform and logarithmic quantisers with a time-varying approach is considered in Guo and Dimarogonas (2013). Here, a quantisation function using uniform and logarithmic quantisation levels is used with fixed quantisation over the time evolution, while convergence can be guaranteed when logarithmic quantisers are applied. To reduce the communication effort regarding the quantisation levels, Yi and Hong (2014) develop an adopting quantisation scheme, which uses a zooming technique to adapt the quantisation levels based on the state with subject to the equilibrium state. Pu et al. (2015) develop a quantisation of states which is adopted in every iteration regarding the bit size of the communication channel. Therefore, the quantise operator and states are transmitted in every time instant.

Liu et al. (2016) use instead a flexible quantisation adoption to follow an event-triggered approach where changes are also transmitted in a quantised state when the system hit a certain threshold depending on the last state. Here, the final states of all agents are additionally regarding a common set constraint. Zhu et al. (2016) apply for an ADMM-scheme an a-priori chosen quantised setting for the exchange of the consensus constraints among



**Figure 6:** Classification of the stability approaches

the agents. They show, that the quantisation size is proportional to the final state region (or consensus level) of the subsystems which allows the system to reach their equilibrium, here consensus faster. In Chen et al. (2016), the authors evaluate a scheme where a sensor network communicating with a central instance is utilising an  $H_2/H_\infty$  estimator to incorporate unknown noise. To reduce the communication load further, they introduce a dual data compression strategy which is based on a weighted random choice of the sensors components for transmitting data. Li et al. (2018) combines the features of necessary quantisation levels and event-triggered communication in a cooperative scheme to derive conditions for convergence and optimality. Each agent holds a separate encoder-decoder pair to the other agents, which is driven by an event-triggering threshold based on the measurement and quantisation error, which is kept by each agent to calculate the next time instant when an update is necessary. Then, with mild assumptions and for a convex problem, where the existence of a solution is assumed, the convergence rate can be determined for a strongly connected system.

### 2.2.5 Stability

Based on the presented results in the introductory Section 2.1, the concepts from (centralised) MPC were also applied for the distributed variant with a variety of modifications. Achieving convergence or stability in DMPC rises more difficulties due to the coordination effort of distributed schemes, which has to be taken into account. Additionally, as stability comes in many cases with initial and recursive feasibility, many approaches depend stability conditions on that property. In the following section, the introduced approaches are discussed regarding these properties. The distinction of the methods is made into cooperative and non-cooperative schemes, which is also depicted in Fig. 6. With cooperative schemes, the overall costs are a promising candidate to be used to achieve stability. Nevertheless, the

drawbacks comes in most cases with an additional coordination effort due to the needed exchange of cost values (Grüne and Worthmann, 2012)[Chapter 1]. In the fundamental DMPC paper of Jia and Krogh (2001), the authors propose a stability constraint, which restricts the predicted states for each  $x_i$ , where  $i$  describes the  $i^{\text{th}}$  subsystem, by an upper constant bound  $\beta$ . Here, the drawback is that only a Nash equilibrium can be achieved if the setting is established in a non-cooperative manner.

On the one hand, for cooperative schemes, Venkat et al. (2005) propose stability to achieve optimality for LTI (linear time invariant) systems, i.e. the Pareto front, via coupled objectives. Here, the costs are made available in a composite model, which provides full state information and full costs for all subsystems. Dunbar (2007a) extends this for non-linear systems, where a consistency constraint is used between neighboured systems, which ensures a bounded deviation between calculated and assumed trajectory, which are estimated by the neighbours to reach a pre-defined terminal set. Heidarinejad et al. (2011b) use a multi-rate Lyapunov function for distributed controllers, which are not adjusted to the same sampling time, i.e. systems with slow or fast dynamics. An iterative scheme is proposed, which uses Lyapunov-based controllers in such a manner that each local controller finds a trajectory bounded by an input constraint. This restricts the differences of the iterations to coordinate the possible actions of the controller and constraint by a necessary decrease of the optimal value function as this serves the Lyapunov definition.

On the other hand, for non-cooperative schemes, Jia and Krogh (2001) impose a deviation on the states of the subsystems, where the controllers are bounded by a stability constraint to keep the overall system near to the equilibrium. The cost function is defined locally and optimised solely by each controller to obtain a local optimum. Unlifting the restrictions of terminal costs or terminal constraints for distributed non-cooperative schemes, Grüne and Worthmann (2012) use the controllability assumption from Grüne et al. (2010) for unconstrained (centralised) MPC schemes and apply this for the distributed scenario. To ensure feasibility, the scheme is initialised with fixed abstract feasible position. Stability is ensured based on the assumption that in any time instant a feasible control set for each subsystem exists, which is admissible, i.e. the conclusion that in every time instant the systems could solve an emerging conflict. Pannek (2013) extends this approach by the ability to use a parallel optimisation for the agents instead of the sequential scheme proposed by Richards and How (2004b). He uses a deordering rule to generate a hierarchy of dependencies with a memory rule to recognise possible periodic states, which might lead to deadlocks.

Another possibility to show stability for distributed (linear) systems is to derive properties from the central problem formulation to use either dual decomposition methods (Giselsson, 2010) or using a stopping criterion. The latter is based on the difference on the obtained dual costs to reduce the necessary number of necessary iterations as such methods normally converge slowly. This approach is extended in Giselsson and Rantzer (2014) with a adaptive constraint tightening on the subsystems to guarantee that the optimisation is already in a tightened solution space to allow for a smaller number of iterations. Lucia et al. (2015) use a contract-based setting, most similar to Jia and Krogh (2001) by applying contracting on

the coupling variables between subsystems. There, each controller estimates a sequence of contracts, wherein the predicted trajectories will lie in the future. This approach allows for the inclusion of disturbances or noised channels, which may occur in network control systems.

### 2.2.6 Priority rules

In this part, we discuss the utilisation of dynamic priority rules for traffic and robotic scenarios in recent decades. Originated in queuing theory, policies such as FIFO (First-in-first-out) or LIFO (Last-in-first-out) assign priorities to agents based on criterion values, e.g. in order of appearance or lexicographical order. This results in a fixed order, which is also used by Richards and How (2004b).

Considering traffic scenarios especially for unsignalised intersections, many approaches utilise a priority rule based on static infrastructure, e.g. the lanes of streets (Wu, 2001). She uses the queuing theory to measure the capacity of intersections with the arrival rate of vehicles. Troutbeck and Kako (1999) propose a limited priority scheme, which imposes a small fixed delay on the major lanes to allow vehicles to merge from the minor lane. This scheme requires a continuous and regular flow of vehicles in both major and minor lanes. Priority lanes considering different route option to go from a common source to a target are also used in Evers and Proost (2015), while the vehicles are assigned to their route based on the Stackelberg Game. In this setting, a central unit called a government instance assigns the availability of the routes to all vehicles. Each vehicle decides sequentially, which route should be taken depending on the decision made on the previous vehicles and the available routes. The results show that the lowest costs for all vehicles (in terms of waiting time, travel time and resource costs) are achieved by blocking one route or lane as the waiting times considering the congestions on the routes with less priorities were static. Furthermore, the results are compared to a classical traffic lights system, which also blocked the minor routes completely.

In the field of mobile robots, the first approaches to control and plan are implemented in a centralised manner: Erdmann and Lozano-Pérez (1987) propose a priority scheme for collaborating robots, which is assigned a-priori, based on tasks to be fulfilled. Here, the tasks are dependent on each other. The configuration space is defined as combined time and state space and additionally, covers the degrees of freedom of the robots. The complexity of planning is reduced as only the movement of previous, higher prioritised robots are regarded. For path-planning, an approach with a-priori assigned priority rules to decompose space and time dimensions to reduce complexity is presented in Kant and Zucker (1986). Here, in the state space, a trajectory is planned which avoids static obstacles. After that, the velocity on this path is calculated to avoid collisions with moving obstacles. Buckley (1989) proposes a priority graph approach to detect the dependencies of path-planning. He distinguishes between start and goal conflicts: The robots, which are moving in a straight-line motion, should either give way to the other robots when a conflict is recognised in their start region or otherwise would be highly prioritised if they are close to their target region.

Furthermore, intermediate occurring deadlocks, which appear as cycles in the graph, are solved by taking out the last detected robot in the cycle, which is then assigned the lowest priority. Last, Yongjie (2009) applies a potential field control approach in combination with simulated annealing to ensure real-time capabilities and a fixed priority scheme correlated to predefined tasks of the robots.

Considering a distributed and dynamic behaviour of mobile robots, Bourbakis (1997) develops a dynamic priority scheme with traffic rules for mobile robots without a central control instance. The complexity is reduced as the robots synchronise only with direct neighbours and follow a predefined traffic rule set, which utilises an analysis of free corridors by avoiding obstacles and other robots in the given operational space. In Bennewitz et al. (2002), the heuristic  $A^*$ -search path-planning algorithm is combined with a dynamic priority scheme. In this framework, an a-priori order is generated based on the minimum path length. The priority scheme evaluation is executed in a random search following the hill-climbing approach. In Fankhauser et al. (2011), a priority scheme for robots on predefined paths by inertia values is used. Here, robots with higher inertia gain a higher priority. Liu et al. (2014) adapt a collision avoidance scheme for mobile robots to a dynamic priority scheme. Using a dependency graph of collision avoidance constraints, the robot gains higher priority with a larger subtree, i.e. with a larger subtree they obey to more collision avoidance constraints of the other robots. In Luo et al. (2016), a dynamic priority scheme is utilised in which the robots bid for a future collision assessment to ensure that the robot with highest number of collision avoidance constraints gains highest priority. In a more general setting, priority rules in combination with non-linear DMPC (non-cooperative) are utilised in Pannek (2013) to allow parallel execution order of the distributed agents and to establish a time-dependent hierarchy whenever sequential execution is necessary.

## 2.3 Summary of state of the art

Considering the communication effort and quantisation scheme, in our setting, with a non-cooperative scheme and coupling constraints induced by the other agents or robots, we rely on fixed quantisation scheme, which is based on the available state space. Furthermore, a global quantisation scheme, which is equal among all agents, does not require an individual coding-/decoding scheme, which might add additional effort due to blockage of wireless communication channels and a dense setting with all agents/robots in a shared operational space.

From the discussed publications of priority calculations, we identify that priority rules are either time- or state-based. Time-based criteria utilise the time since the robot has entered the scenario or the required time to reach the target. State-based criteria, on the other hand, consider either the control effort or the minimal distance to target. While the first aims for energy preservation, the second prefers to retain the shared space mostly empty to avoid unnecessary detours for the others.

The presented overview considering coordination, communication, quantisation, stability and priority rules in distributed model predictive control leads to several research questions, which are formulated in the next section: In our case we use a non-cooperative scheme where each agent optimises using its own local OCP. We utilise a communication scheme, which will be implemented in several stages utilising different reduction levels of the communication. To illustrate our results, we implement a robot scenario to establish a distributed setting. Based on this scenario, we are using the following methods for the formulated research questions from Chapter 1:

## Outline for Contributions

We propose the methods, which are described in Chapter 4 and Chapter 5 to get answers for the formulated questions.

1. A quantisation method for the states is proposed in Chapter 4.
2. Furthermore, we discretise the state set by an *occupancy grid* in Section 4.1.1 to be able to reduce the communication effort between the controllers in one sampling step utilising *differential communication* in Section 4.1.3 based on *prediction coherence* in Section 4.1.2 and extends this reduction by *interval communication* in Section 4.1.5.
3. We introduce dynamic priority rules in Chapter 5 incorporating different criteria to improve the system performance in Section 5.1 and the adopted schemes in Section 5.2.
4. To achieve practical stability especially in a setting of distributed robots, we show in Section 4.2 with the utilisation of a switched control law, how a coordinated scheme can be established to ensure this property.

# 3

## Problem setup

We consider a distributed system with describing the state  $z_p$  for each subsystem  $p$ , where  $p \in [1: P]$  and  $P \in \mathbb{N}$ . The underlying model for each subsystem  $p$  is discretised via finite time instants and the following state is defined by  $z_p^+$  obtained by a function  $f : \mathbb{X} \times \mathbb{U} \rightarrow \mathbb{X}$ , which describes the dynamics with

$$z_p^+ = f(z_p, u_p) \quad (7)$$

and

$$u \in \mathbb{U}, \quad \mathbb{U} \subset \mathbb{Z}^{n_u} \quad (8)$$

as a zero-order-hold control to obtain the next state and

$$\mathbb{X} \subset \mathbb{Z}^{n_x} \quad (9)$$

denoted as the feasible states of the system.  $\mathbb{Z}^{n_u}$  defines the control set with dimensions  $n_u$  and  $\mathbb{Z}^{n_x}$  the state set with dimensions  $n_x$ . The trajectory for a subsystem is defined as

$$\begin{aligned} \mathbf{z}_p^{\mathbf{u}}(\cdot; z_p^0) &:= (z_p^u(0; z_p^0), z_p^u(1; z_p^0), \dots, z_p^u(N; z_p^0)) \\ &= (z_p^0, f(z_p^0, u_p(0)), f(z_p^0 + f(z_p^0, u_p(0)), u_p(1)), \dots, \\ &\quad f(z_p^0 + \dots + f(z_p^0, u_p(N-1)), u_p(N-1))) \subset \mathbb{X} \end{aligned} \quad (10)$$

where  $\mathbb{X}$  is the operation space and  $\mathbf{u}_p = (u_p(0), u_p(1), \dots, u_p(N))$  the input control sequence where we omit the index  $p$  for  $\mathbf{u}$  in  $\mathbf{z}_p^{\mathbf{u}}$ .  $z_p^0$  denotes the feasible initial positions for each subsystem  $p$ . If not stated otherwise,  $\mathbf{z}_p^{\mathbf{u}}(\cdot; z_p^0)$  runs from indices  $k = 0$  to  $N$  with  $z_p^u(k; z_p^0)_{k=0}^N$ . Each subsystem  $p$  has the same properties solving a local OCP subject to their individual assigned targets  $z_p^*$  starting from an initial state  $z_p^0$ . Each subsystem optimises

the local OCP  $J_p^N(z_p^0; u_p)$  to obtain an optimal trajectory of length  $N$  starting from initial time  $n_0$ . The OCP minimises the objective  $\ell_p: \mathbb{X} \times \mathbb{U} \rightarrow \mathbb{R}_{\geq 0}$

$$\ell_p(z_p^u(n), u_p(n)) := \|z_p(n) - z_p^*\|_2 + \lambda \|u_p(n)\|_2 \quad (11)$$

with  $\lambda \geq 0$  and measures the distance between the current state of the system and the target. We add a fraction of the control  $u_p$  to avoid scattering behaviour. To incorporate the constraints from the other subsystem in the local OCP, we define such an information broadcasted by one subsystem  $p$  as a sequence of tuples

$$\mathbf{I}_p = \mathbf{z}_p^u(\cdot; z_p^0) \quad (12)$$

with  $\mathbf{I}_{p,k}$  returning the  $k^{\text{th}}$  tuple and the received tuples obtained by other systems with

$$\mathbf{i}_p = (\mathbf{I}_1, \dots, \mathbf{I}_{p-1}, \mathbf{I}_{p+1}, \mathbf{I}_P). \quad (13)$$

where  $\mathbf{i}_{p,k}$  denotes the  $k^{\text{th}}$  tuples in the received sequences. This allows to state a simple distance law to ensure collision avoidance, which is adapted depending on the dynamics of the underlying system for two subsystems  $p, q \in [1: P]$  with  $p \neq q$  as

$$g_{q,k} := g(z_p^u(k; z_p^0), \mathbf{I}_{q,k}) = \|z_p(k) - \mathbf{I}_{q,k}\|_2 - r_{\min} \geq 0, \quad \forall k \in [1: N], \quad (14)$$

where  $r_{\min} \in \mathbb{R}_{\geq 0}$  denotes a safety margin between the subsystems. Combining all the received information in a constraint set, this is stated as

$$G(z_p(k; z_p^0), \mathbf{i}_{p,k}) = (g_{1,k}, \dots, g_{p-1,k}, \dots, g_{P,k}), g_{i,k} \geq 0, \forall i \in [1: P].$$

The overall OCP is defined to minimise the objective function based on the stage costs (11) over a finite horizon for each subsystem  $p$  given by

$$\min_{\mathbf{u}_p^*} J_p^N(z_p^0; u_p) := \sum_{k=0}^{N-1} \ell_p(z_p^u(k), u_p(k)) \quad (15)$$

$$\text{s.t.} \quad z_p^u(k) \in \mathbb{X}, \quad k \in [0: N],$$

$$u(k) \in \mathbb{U}, \quad k \in [0: N-1],$$

$$z(k+1) = f(z(k), u(k)), \quad k \in [0: N-1],$$

$$G(z_p(k; z_p^0), \mathbf{i}_{p,k}) \geq 0, \quad k \in [1: N]. \quad (16)$$

Then each subsystem executes the MPC-Alg. 1 to obtain an optimal control sequence  $\mathbf{u}_p^*$  from (15) with respect to given constraints. Constraints could for example be imposed by collision avoidance restrictions in traffic scenarios or capacity restrictions in production scenarios. Each subsystem applies the first control  $\mathbf{u}_p^*(0)$  of the obtained minimization. To handle the communication, different strategies are used. Either a central communication scheme is used based on a central controller instance or a distributed scheme is utilised, where the subsystems coordinate on their own.

### 3.1 Centralised communication

At first, we utilise a central controller instance as presented in the following scheme:

---

**Algorithm 2** DMPC procedure for the overall system with a central communication instance

---

**Require:** initial time  $n$  and number  $P$  of active subsystems and sequence of  $z_p \in \mathbb{X}$ ,  $p \in [1: P]$  with feasible initial position  $x_p^0$ .

**while**  $P > 0$  **do**

**Measure** current state  $z_p(n)$  for each subsystem  $p = [1: P]$ .

**for all** subsystems  $p \in [1: P]$  **do**

**if**  $z_p(n) = z_p^*$  **then**

**Remove** subsystem  $p$ ,  $P := P - 1$

**end if**

**while** trajectory  $\mathbf{I}_p$  is not acknowledged **do**

**Compute** optimal control  $\mathbf{u}_p^*$  for (15) over finite horizon  $N$

**Request** acknowledgement for  $\mathbf{I}_p$  from controller  $C$

**end while**

**end for**

**Implement** first control elements  $\mathbf{u}_p^*(0) \forall p = [1: P]$

**Shift** horizon  $n := n + 1$

**end while**

---

The scheme presented in Alg.2 is non-cooperative as each subsystems is assigned a local cost function according to (11), which penalised the deviation from the target state  $z_p^*$ .

**Example 3.1** (Discrete scenario)

A discrete scenario describing an intersection scenario was defined in Sprodowski and Pannek (2015), where the operational space was discretised into equidistant cells with a given dynamics for each subsystem  $p$  with  $z_p = (n, x_p, y_p) \in \mathbb{N}_0 \times \mathcal{G}$ . Here,  $\mathcal{G} = \{0, 1, \dots, a_{max}\} \times \{0, 1, \dots, b_{max}\} \subset \mathbb{N}_0^2$  is the discrete state set,  $n$  the time instant and  $x, y$  for the grid index. The control is restricted to  $u \in [-1, 1]^2 \subset \mathbb{Z}^2$  with  $|u| \leq \sqrt{2}$ . A central controller  $C$  was established, which handles the communication and assign the demanded cells to the subsystems

$p$  with  $p \in [1: P]$  where such information based on  $\mathbf{I}_p$  was exchanged between controller and subsystem via discrete time-dependent states

$$\mathbf{I}_{p,n}^d = \left\{ (n+k, z_p(n+k))_{k=0}^N \right\} \quad (17)$$

and  $\mathbf{I}_{p,n,k}^d = (n+k, z_p(n+k))$ . Collision avoidance constraints were ensured by the requirement, that each tuple  $(n, z_p(n))$  is unique. The controller instance  $C$  is running the following algorithm to handle all the requests from the subsystems and regard the collision avoidance constraints:

---

**Algorithm 3** Controller algorithm to handle communication between subsystems and ensure collision avoidance

---

**Require:**  $n = 0, P$

**while**  $P > 0$  **do**

**Set**  $M(n) = \emptyset$

**while** not all trajectories  $\mathbf{I}_p^d$  of  $p$  with  $p \in [1: P]$  are acknowledged **do**

**Receive** trajectory  $\mathbf{I}_p$

**if**  $\mathbf{I}_{p,n,k}^d \in \mathbf{I}_p$  and  $\mathbf{I}_{p,n,k}^d \notin M(n) \forall k \in [1: N]$  **then**

**Add**  $M(n) := M(n) \cup \mathbf{I}_{p,n,k}^d$

**Acknowledge**  $\mathbf{I}_p$

**else**

**Reject**  $\mathbf{I}_p$

**end if**

**end while**

**Set**  $n := n + 1$

**end while**

---

The local OCP is solved in an iterative manner, where each system requests via (17) its optimal trajectory to the controller  $C$  defined in Alg. 3, which either acknowledge or discard the requested trajectory to ensure collision avoidance. Stability was ensured via decaying Lyapunov function according to (5), where the monotonous decrease was relaxed that the overall costs of the system have to be below a decaying worst-case function, which is the serial execution of the intersection scenario.

In many applications, e.g. platooning robots (Dunbar and Caveney, 2012; Farina et al., 2015) or controlling chemical plants (Rawlings and Stewart, 2008), a continuous state set is used as the system dynamics is described as a continuous function and furthermore, no additional central coordination instance is present and therefore, decentralised communication is utilised. In that case, we follow the scheme from Richards and How (2004a), where the subsystems are coordinating their communication sequentially without a central instance. As the distributed communication is the main focus of this thesis, we present this in the following section, where the following chapters also build up on the presented scheme therein.

## 3.2 Distributed communication

The communication between the subsystems in DMPC is crucial, because necessary information about constraints, states or costs have to be transmitted to subsystems to ensure that convergence can be achieved and constraints imposed by neighbours are obeyed to ensure collision avoidance (Venkat et al., 2005). In DMPC there are several possibilities to implement the communication among them. One can consider a broadcast, which informs all subsystems about the transmitted information, a multicast, where only some chosen subsystems are included and unicast, which incorporates only client-to-client communication. Here, due to the density of subsystems on the shared operation space in a robotic setting and a shared wireless medium, we consider a broadcast approach. The information each subsystem  $p$  with  $p \in [1 : P]$  transmits as a broadcast is defined as

$$\mathbf{I}_{p,n} = \left\{ \left( p, \left( n + k, z_p^u(k; z_p^0) \right)_{k=0}^N \right) \right\}, \quad (18)$$

where the  $p^{\text{th}}$  subsystem transmit the predicted trajectory over horizon length  $N$  to all other subsystems based on the current time instant  $n$ . Then, each subsystem receives the broadcasted information from all other subsystems with

$$\mathbf{I}_{q,n} = \left\{ \left( q, \left( n + k, z_q^u(k; z_q^0) \right)_{k=0}^N \right) \mid q = [1 : P] \setminus \{p\} \right\},$$

where the  $q^{\text{th}}$  subsystem transmit the predicted trajectory over horizon length  $N$  to all other subsystems.

As the subsystems are optimising not iteratively, but sequentially, the updates of the trajectories differ as earlier optimising instances have information only from the last time instant. We stated this as the symmetry problem in Sprodowski et al. (2018a). To solve this issue, we propose in Alg. 4 an assembling of these missing information.

---

**Algorithm 4** Resolving asymmetry of communication data for robot  $p$

---

**Require:** subsystem  $p$  and communication data  $\mathbf{I}_{q,n}$  for  $q \in [1 : p - 1]$  and  $\mathbf{I}_{q,n-1}$  for  $q \in [p + 1 : P]$

- 1: **for**  $q = p + 1$  to  $P$  **do**
  - 2:     **for**  $k = 1$  to  $N - 1$  **do**
  - 3:         **Set**  $\mathbf{I}_{q,n,k} := \mathbf{I}_{q,n-1,k+1}$
  - 4:     **end for**
  - 5:     **Set**  $\mathbf{I}_{q,n,N} := \mathbf{I}_{q,n-1,N}$
  - 6: **end for**
  - 7: **Set**  $\mathbf{i}_p(n)$  according to (13)
- 

Then, utilising the solution of the symmetry problem the distributed MPC scheme is presented in Alg. 5. Here, we make the assumption that initial feasible states are given

**Algorithm 5** DMPC-Algorithm for the overall system**Require:** admissible, initial states  $z_p^0$  for all  $p \in [1 : P]$  and  $n = 0$ 


---

```

1: for  $p = 1$  to  $P$  do
2:   for  $k = 1$  to  $N$  do
3:     Set  $\mathbf{I}_{p,n,k} := \left( p, \left( k, z_p^0 \right)_{k=0}^N \right)$ 
4:   end for
5:   Broadcast  $\mathbf{I}_{p,n}$ 
6: end for
7: for  $n = 0, 1, \dots$  do
8:   for  $p = 1$  to  $P$  do
9:     Measure  $z_p(n)$ 
10:    if  $n = 0$  then
11:      Receive  $\mathbf{I}_{q,n}$  for  $q \in [1 : P] \setminus \{p\}$ 
12:    else
13:      Receive  $\mathbf{I}_{q,n}$  for  $q \in [1 : p - 1]$  and  $\mathbf{I}_{q,n-1}$  for  $q \in [p + 1 : P]$ 
14:      Assemble  $\mathbf{I}_{q,n}$ ,  $q \in [p + 1 : P]$ , using Alg. 4
15:    end if
16:    Set  $\mathbf{i}_{q,n}$  according to (13)
17:    Solve OCP (15) and Apply  $u_p^*(0)$ 
18:    Broadcast  $\mathbf{I}_{p,n}$ 
19:  end for
20: end for

```

---

for all subsystems. Then, in the first step  $n = 0$  the initial position is taken for the whole prediction horizon length and broadcasted to the other subsystems. Then, each subsystem receives and assembles the cell via Alg. 4 to the full information to impose them into the OCP. Then, the resulting trajectory is broadcasted.

As our setting allows for a variety of dynamics, we utilise underlying continuous dynamics, which is discretised. The connection between continuous systems and their discretisation especially for non-holonomic robots (D'Andréa-Novel et al., 1995) is examined in Worthmann et al. (2015) for continuous systems and Worthmann et al. (2016) for discrete systems. As we focus in the following on the communication reduction, we omit to explicitly state the conditions concerning the discretisation and refer to Worthmann et al. (2016). To coordinate the subsystems for stabilising the systems to their targets without a central controller instance, we will introduce in the next Chapter 4 the communication between the subsystems and how it is established. Later, the reduction methods considering the associated publications are presented and the effective is demonstrated in simulations.

In this chapter, we presented the problem setup and proposed a centralised communication scheme utilising a central controller instance. Furthermore, we extended this to a distributed communication scheme with communication solely between the subsystems. The order of solving the OCP and communication is in a sequential manner, which leads to an asymmetry of information, which was additionally handled by adding the missing information.

# 4

## Quantisation and stability

In the Section 3.1 of the last chapter an approach considering a centralised communication was considered (Sprodowski and Pannek, 2015). The communication was handled by a central controller as stated in Alg. 2. The communication and optimisation order, respectively, is established in this and all following chapters in a decentralised manner as presented in Alg. 5 in the last Section 3.2 following the scheme from Richards and How (2004a). In that way, a central controller is dispensable now as the agents are carrying out the optimisation and application of their control in a fixed order. As second, the dynamics of the agents was previously modelled in a discrete manner (Sprodowski and Pannek, 2015). The grid size and discrete model matched congruently in geometrical sense that a movement in one time instant matches one step to a neighboured cell. In that case, safety margins when agents are on the boundary between two cells could be disregarded as the assumption was made that the agents are always in the centre of one cell. Now, in this chapter, the underlying dynamics of the agents is based on a continuous model using a continuous operating space. The implementation of a continuous movement allows a more realistic tracking and state estimation of the real movement of agents, although the state measurement and predictions of the controller are still discretised. Hence, we have to consider that the agents are not exactly positioned in the centre of one cell but could also assign two cells or safety margins have to be considered if they are close to the border to a neighboured cell. The sufficient conditions to ensure collision avoidance under these aspects are presented jointly with the definition of the quantisation scheme in Subsection 4.1.1.

The quantisation and reduction is proposed in three stages: In a first stage, the communication is quantised, while the optimisation is still conducted in continuous space. Hence, for the quantisation the method of an occupancy grid was proposed, which is handled in a decentralised manner and validated with simulations and experiments in Section 4.1.1. In the second stage, to aim for communication reduction between the subsystems, a differential update scheme is used based on the prediction coherence, which is defined in Subsection 4.1.2. Then, derived from the principle of a revision control system, a differential communication scheme is proposed in Section 4.1.3. A short explanation about feasibility aspects is then

following in Section 4.1.4. Furthermore, utilising interval communication at the third stage allows to decrease the communication load to a minimum of two tuples per time instant, while a drawback with space consumption could be attenuated with relaxing the collision avoidance constraints over the evolving time, presented in detail in Section 4.1.5.

Convergence and practical stability conditions for the distributed system are then explored in Section 4.2 supplemented with a discussion about the sufficient prediction horizon length before the contributions are summarised and evaluated in Section 4.3.

## 4.1 Reduction of communication

For the continuous scenario from Chapter 3, we want to reduce the size of the communication load. To this end, we introduce an overlying grid, which quantises our continuous operation space onto a discretised space with equidistant cell sizes. If the cell size is appropriate, we can reduce the number of communicated predictions of the occupied cells which are less than the prediction of the full state. Furthermore, the size of data can be reduced, if we use integers as data type for the cell enumeration instead of double values for communicating the prediction denoted as occupancy grid prediction. Hence, we call this grid an occupancy grid as all cells are contained, which are ought to be used by the robots. We define several stages for the reduction of the communication: First, we motivate an occupancy grid in Section 4.1.1, which quantise the predicted state trajectory onto a fixed grid, where the cell sizes of the grid are equidistantly chosen and equal among all subsystems. We can reduce the communication significantly by exchanging only the altered cells of the predictions, which is described in Section 4.1.3. Furthermore, to reduce the communication load in each time step through the introduction of differential communication and interval arithmetic, these approaches are presented in Section 4.1.3 and 4.1.5, respectively.

### 4.1.1 Occupancy grid

We utilise a quantisation of the planar operation space  $\mathbb{X}$  via the definition of a discretised space  $\mathcal{G} := \{1 \cdots, a_{\max}\} \times \{1, \cdots, b_{\max}\} \subset \mathbb{N}^2$  with cell size  $c$ , where the following conditions for  $(x, y) \in [-\bar{x}, \bar{x}] \times [-\bar{y}, \bar{y}] \subset \mathbb{X}$  holds:

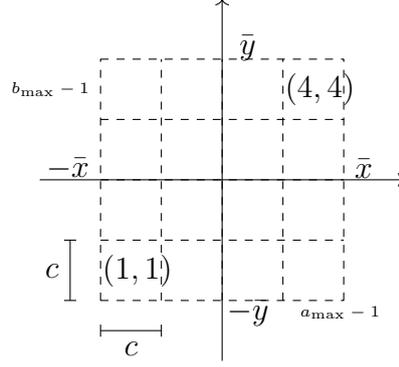
$$a_{\max} = \frac{2\bar{x}}{c}, \quad a_{\max} \in \mathbb{N}$$

and

$$b_{\max} = \frac{2\bar{y}}{c}, \quad b_{\max} \in \mathbb{N}.$$

An example for such a grid is illustrated in Fig. 7. Then, a position  $(x, y)$  of a given subsystem  $p$  can be quantised from the continuous space  $\mathbb{X}$  to  $\mathcal{G}$  in the form of cell indices. This is done via the mapping  $q: \mathbb{X} \rightarrow \mathcal{G}$  with

$$q(z_p) = \left( \left\lfloor \frac{x_p}{c} \right\rfloor + 1, \left\lfloor \frac{y_p}{c} \right\rfloor + 1 \right) = (a, b) \in \mathcal{G}.$$



**Figure 7:** Example of space quantisation  $\mathcal{G}$  with  $a_{\max} = b_{\max} = 4$ , from Sprodowski et al. (2018a)

Now, a trajectory (10) discretised via grid indices based on (12) is defined according to (18) as

$$\mathcal{I}_{p,n} = \left( n + k, q \left( z_p^u \left( k; z_p^0 \right) \right) \right)_{k=0}^N, \quad (19)$$

which represents the occupied cells by the  $p$ th subsystem with  $k \in [0 : N]$ . An explicit tuple  $(n+j, q(z_p^u(j; z_p^0)))$  is denoted as  $\mathcal{I}_{p,n,j}$ . For the receiving robots, the tuples are assembled to

$$\mathbf{i}_{p,n} = (\mathcal{I}_{1,n}, \dots, \mathcal{I}_{p-1,n}, \mathcal{I}_{p+1,n}, \dots, \mathcal{I}_{P,n}), \quad (20)$$

which is then denoted as an occupancy grid as this assembles all occupied cells by the subsystems to a comprehensive information.  $\mathbf{i}_{p,n,k}$  denotes the assembled tuples of the subsystems  $1, \dots, p-1, p+1, \dots, P$  for time instant  $k$ . The received occupied cells could be transformed back to the continuous space  $\mathbb{X}$  with  $h: \mathbb{N}_0 \times \mathcal{G} \rightarrow \mathbb{X}$

$$h(n, q(z_p)) = ((a - 0.5)c, (b - 0.5)c)^\top, \quad q(z_p) = (a, b) \in \mathcal{G}.$$

To plug in the constraints for the OCP (15), a suitable form of the constraints are formulated for a 2D-state  $z_p = (x_p, y_p)^\top$  as

$$g_{q,k} := g \left( z_p^u \left( k; z_p^0 \right), \mathcal{I}_{q,n,k} \right) = \left\| \begin{pmatrix} x_p \\ y_p \end{pmatrix} - h \left( \mathcal{I}_{q,n,k} \right) \right\|_\infty - r_{\min} \geq 0, \quad k \in [1 : N],$$

which is assembled to

$$G \left( z_p^u \left( k; z_p^0 \right), \mathbf{i}_{p,n,k} \right) = (g_{1,k}, \dots, g_{p-1,k}, g_{p+1,k}, \dots, g_{P,k}), \quad G_{i,k} \geq 0, \quad \forall i \in [1 : P] \setminus \{p\},$$

and integrated into (16).

In Mehrez et al. (2017), the problem setup was first defined for non-holonomic robots to be able to create an easy prototype to validate this approach. Hence, the state is defined as  $z_p := (x_p, y_p, \theta_p)^\top \in \mathbb{X} \subset \mathbb{R}^3$  regarding the 2D space and orientation  $\theta_p$  of a robot. The system is trivially underactuated as the control is imposed on linear and angular speed,

respectively  $(u_p = (\nu_p, \omega_p) \in \mathbb{R}^2)$ . Then the OCP for each robot is formulated according to (15) using the quantisation scheme as proposed in (19) for broadcasting and (20) for receiving tuples. Then, with the assumption that feasible position are provided at the initialisation, the DMPC scheme with quantisation is then defined in Alg. 6. To ensure that feasibility is

---

**Algorithm 6** DMPC-Algorithm with quantised communication for the overall system

---

**Require:** admissible, initial states  $z_p^0$  for all  $p \in [1 : P]$

```

1: for  $p = 1$  to  $P$  do
2:   for  $k = 1$  to  $N$  do
3:     Set  $\mathcal{I}_{p,n,k} := \left( p, \left( k, z_p^0 \right)_{k=0}^N \right)$ 
4:   end for
5:   Broadcast  $\mathbf{I}_{p,n}$ 
6:   Set  $\mathbf{Q} = \emptyset, w_p = \text{false}$ 
7: end for
8: for  $n = 0, 1, \dots$  do
9:   for  $p = 1$  to  $P$  do
10:    Measure  $z_p(n)$ 
11:    if  $n = 0$  then
12:      Receive  $\mathbf{I}_{q,n}$  for  $q \in [1 : P] \setminus \{p\}$ 
13:    else
14:      Receive  $\mathbf{I}_{q,n}$  for  $q \in [1 : p - 1]$  and  $\mathbf{I}_{q,n-1}$  for  $q \in [p + 1 : P]$ 
15:      Assemble  $\mathbf{I}_{q,n}$ , for  $q \in [p + 1 : P]$ , using Alg. 4
16:    end if
17:    Set  $\mathbf{i}_{p,n}$  according to (13)
18:    Solve OCP (15) and Apply  $u_p^*(0)$ 
19:    Broadcast  $\mathbf{I}_{p,n}$ 
20:  end for
21: end for

```

---

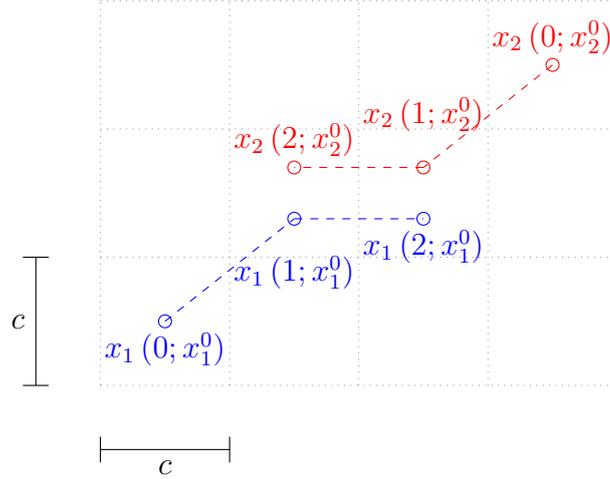
not only guaranteed at the start (initial feasibility) but also at later time instants (Grüne and Pannek, 2017, Chapter 3), the following assumption is made:

**Assumption 4.1** (Immediate Hold)

Each robot can come to an immediate hold, i.e.

$$\forall z_p \in \mathbb{X} \exists \bar{u}_p \in \mathbb{U} : z_p = f(z_p, \bar{u}_p). \quad (21)$$

This assumption ensures that the algorithm is able to find a feasible solution in every time instant. A most intuitive solution would be  $\bar{u}_p \equiv 0$  for most robotic systems although stopping the robot immediately would not be realistic in practical settings due to delays in actuators. Hence, a safety margin could be added to satisfy this assumption. Furthermore, as the control is induced in discrete time instants on a continuous model, a minimum grid size has to be calculated to prevent robots from skipping cells. We assume that the underlying



**Figure 8:** A possibility of swapping cells between two robots with states  $x_1, x_2$  and starting conditions  $x_1^0, x_2^0$  for time instants  $n = 0, 1, 2$ , which would be unnoticed (from Sprokowski (2021)).

model (7) is Lipschitz-continuous with respect to its spatial coordinates and control such that for an  $L > 0$  the following holds:

$$\|f(z_p, u_p) - f(z_p, 0)\|_2 \leq L \|u_p\|_2 \leq \underline{c}.$$

Hence, a minimum cell size  $\underline{c}$  can be calculated for arbitrary  $u_p \in \mathbb{U}$ , which is a sufficient condition for the minimum cell size to avoid skipping cells. An additional difficulty might occur with the discretised measurement and applied control that a swapping between two robots in counterpart could be unnoticed as illustrated in Fig. 8. Here, in time instants  $n = \{1, 2\}$  such an unnoticed swapping between the two robots occurs, as in both time instants (measurements) the constraints (16) are ensured, but in the continuous time between those instants constraints could be violated. Hence, the following safety margin has to be ensured

$$\left\| \begin{pmatrix} x_p \\ y_p \end{pmatrix} - \begin{pmatrix} x_q \\ y_q \end{pmatrix} \right\|_\infty \geq \max\{d_{\min}, \underline{c}\} \quad (22)$$

where  $p, q \in [1: P]$  are two robots with  $p \neq q$  and  $d_{\min} > 0$  describing a minimal physical distance. To check the impact of larger cell sizes on the amount of communicated tuples and convergence times, we performed simulations for a set of  $P = 4$  robots (Mehrez et al., 2017). Therein, the simulations show that with an increasing cell size the number of communicated tuples increases due to longer simulation times by larger detours. These detours were necessary as the increased cell size impose a larger occupancy grid, which has to be circumvented by the other robots. To this end, the numerical results was completed by experiments demonstrating that the proposed approach work with the usage of quantised communication.

To substantiate our approach with more theoretical results, we examined this on a more abstract level considering the class of control-affine systems in Sprodowski et al. (2018a): The minimum cell size was formulated as the maximum movement in one dimension  $j$  of the spatial coordinates of one robot in one time instant imposed by a control  $u_p$  over  $m$  vector fields by

$$\underline{c} := \max_{u_p \in \mathbb{U}} \left\{ c_{f_0} + \sum_{i=1}^m c_{f_i} |u_{p,i}| \right\} \quad (23)$$

with

$$c_{f_0} := \max_{z_p \in \mathbb{X}} \left\{ \max_{j \in \{1,2\}} |(f_0(z_p) - z_p)_j| \right\} \text{ and } c_{f_i} := \max_{z_p \in \mathbb{X}} \left\{ \max_{j \in \{1,2\}} |f_i(z_p)_j| \right\}. \quad (24)$$

Then, to regard the continuous time interval between the discrete time instants  $t \in [n, n + 1]$ , this safety margin is ensured such that

$$\left\| \begin{pmatrix} x_p^u(k; x_p^0) \\ y_p^u(k; y_p^0) \end{pmatrix} - h(\mathcal{I}_q(n)(k)) \right\|_{\infty} \geq \Psi_q(\mathcal{I}_q(n)(k), \mathcal{I}_q(n)(k-1)) \quad (25)$$

holds for all  $k \in [1: N]$  with

$$\Psi_q(\mathcal{I}_q(n)(k), \mathcal{I}_q(n)(k-1)) = \begin{cases} \bar{\Psi} := \Psi + \underline{c}, & \text{if } \mathcal{I}_q(n)(k) \neq \mathcal{I}_q(n)(k-1) \\ \underline{\Psi} := \Psi + \frac{\underline{c}}{2} \cdot \cos\left(\frac{\pi}{4}\right), & \text{otherwise.} \end{cases} \quad (26)$$

In (26) we consider the distinction that both robots are moving (first case) or that one robot holds its position (second case) and the minimum distance has to be such that the traversing robot retains the safety margin, which can be constructed by Pythagoras.

### 4.1.2 Prediction coherence

To reduce the communication load, in Sprodowski et al. (2017) we examined a property, which we called prediction coherence. The classical MPC scheme optimises in every time instant over the full horizon length and applies only the first obtained control value. The remaining control values are not used for further control. In the next time instant this procedure is repeated allowing us to compare two succeeding predictions. In Sprodowski et al. (2017), the intersection scenario from Sprodowski and Pannek (2015) was used in a continuous way, where additionally directional constraints were introduced to simulate lanes of a street system and force the subsystems to move only on the right lanes of the road. Hence, only the movements of traversing straight, turning left or right were allowed. However, U-turn were forbidden. The directional constraints are based on a lock function

$f_l: \mathbb{X} \rightarrow 2^{\mathcal{G}}$ , which forbids a certain subset of cells. The directional constraint is then formulated as

$$G_D(z_p, z_p^0) := \min_{z^c \in f_l(z_p^0)} \|z_p - z^c\|_{\infty} - r_{\min} \geq 0, \quad (27)$$

where the constraints are then added to the OCP (15) for each subsystem  $p$  leading to the comprehensive problem

$$\begin{aligned} \min_{\mathbf{u}_p^*} J_p^N(z_p^0; u_p) &:= \sum_{k=0}^{N-1} \ell_p(z_p^u(k), u_p(k)) \\ \text{s.t. } z_p^u(k) &\in \mathbb{X} \\ u(k) &\in \mathbb{U} \\ G(z_p(k; z_p^0), \mathbf{i}_p(k)) &\geq 0 \\ G_D(z_p, z_p^0) &\geq 0. \end{aligned} \quad (28)$$

With two successive predictions of one subsystem, given by

$$\mathbf{z}_p^{\mathbf{u}} = \left( z_p^u(k; z_p^0) \right)_{k=1}^{N+1} \quad \text{and} \quad \bar{\mathbf{z}}_p^{\mathbf{u}} = \left( z_p^u(k; z_p^0) \right)_{k=0}^N, \quad (29)$$

we define for  $r_c: \mathbb{X} \times \mathbb{U} \rightarrow \mathbb{R}_{\geq 0}$  the prediction coherence by

$$r_c(\mathbf{z}_p^{\mathbf{u}}, \bar{\mathbf{z}}_p^{\mathbf{u}}) = \sum_{k=0}^N \left\| z_p^u(k+1; z_p^0) - \bar{z}_p^u(k; z_p^0) \right\|_2. \quad (30)$$

This measures the difference in continuous space. As our focus is on the communication of the predicted trajectories, the difference has to be defined for the quantised predictions accordingly:

$$\mathcal{I}_{p,n} = \left( n+k, q \left( z_p^u(k; z_p^0) \right) \right)_{k=1}^{N+1}$$

and

$$\mathcal{I}_{p,n-1} = \left( (n-1)+k, q \left( z_p^u(k; z_p^0) \right) \right)_{k=0}^N$$

for  $n > 0$ . Then, this allows us to define accordingly the quantised prediction coherence  $r_q: \mathcal{G}^{N+1} \times \mathcal{G}^{N+1} \rightarrow \mathbb{R}_{\geq 0}$  via

$$r_q(\mathcal{I}_{p,n}, \mathcal{I}_{p,n-1}) = \sum_{k=0}^N \left\| \mathcal{I}_{p,n,k+1} - \mathcal{I}_{p,n,k} \right\|_2. \quad (31)$$

to measure the difference of the quantised predictions. Simulations were carried out for both scenarios ( $P = 4$ ), the intersectional (Sprodowski et al., 2017) and a robotic scenario (Sprodowski, 2021, Example 2.4). The latter show that the quantised prediction coherence is much lower for a comparable setting with same operational space  $\mathbb{X} = [-6, 6]^2$  due to less

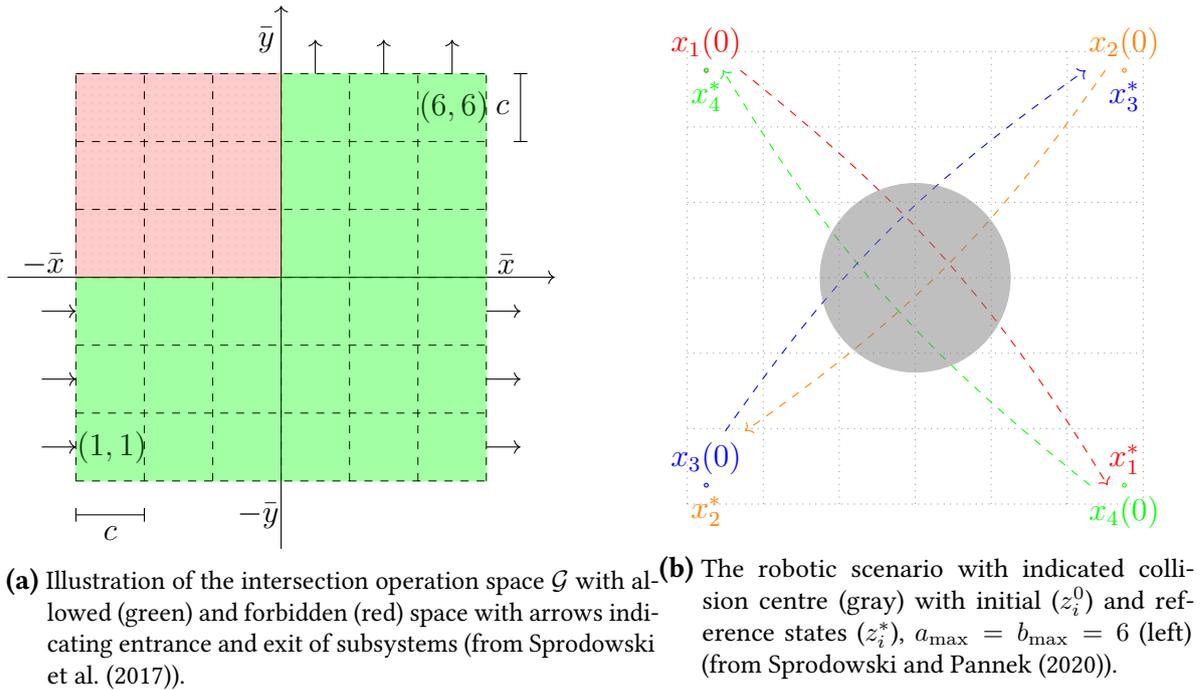
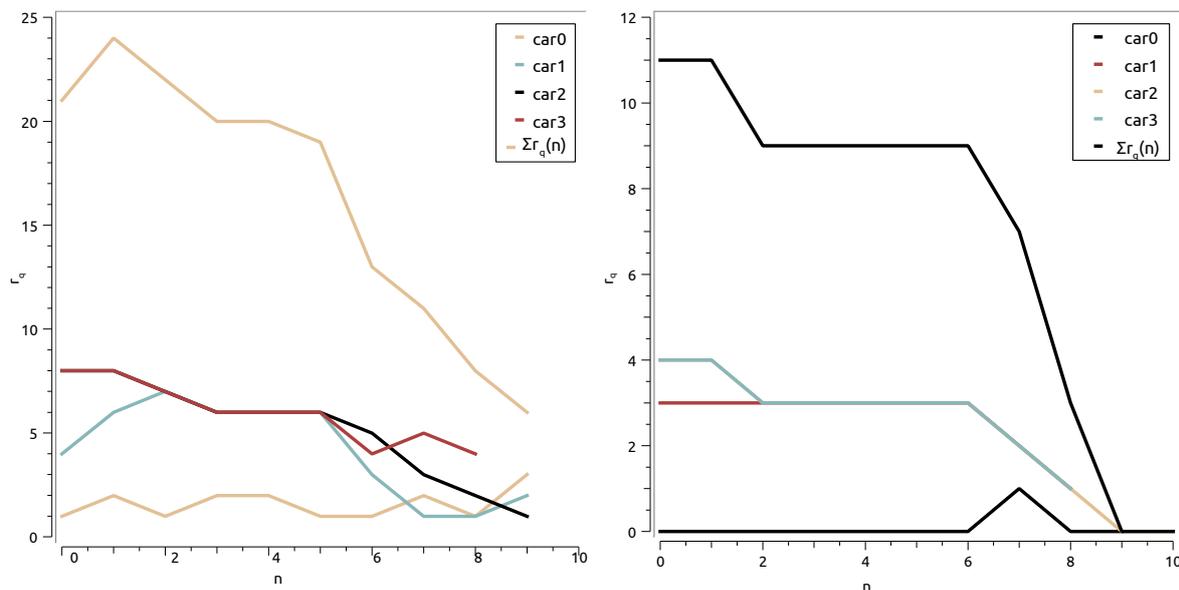
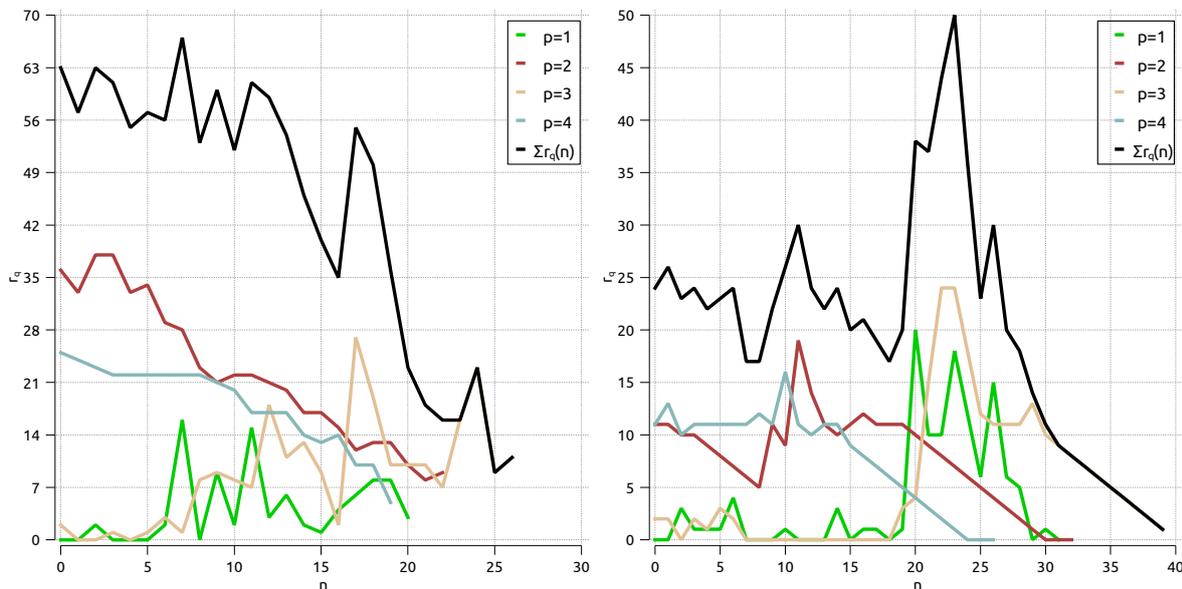


Figure 9

degrees of freedoms. Both settings are illustrated in Fig. 9, where the collision avoidance space is different for both settings in such a way that for the intersectional scenario the collision avoidance takes place mostly at the entrance or exits of the vehicles, while for the robotic scenario this is mostly carried out in the centre of the operational space. An excerpt of the results for both scenarios is depicted in Fig. 10 for the intersection scenario and in Fig. 11 for the robotic one. On the one hand, as the robots do not have to obey directional constraints and optimise with subject to (15), the collision avoidance occur mostly in the centre of the operational space and leads to larger deviations of the prediction. On the other hand, the vehicles regarding the directional constraints incorporated in (28) do not deviate much from their predictions as their degrees of freedoms are more limited and the collision avoidance is mostly obeyed with waiting times. Hence, for both cell sizes, the difference is larger for the robotic scenario (see Fig. 11) due to the less restricted constraints limits as they are not obeying directional constraints. Also, even with the small cell size difference it is recognisable that for both scenarios the predictions are much more similar with the larger cell size and hence, the prediction difference is smaller. As in both scenarios with a larger cell size the subsystem needs more time instances to traverse a cell and the predictions assign less cells, the difference between two subsequent time instants is therefore smaller than with smaller cell sizes. This advantage of less changes in successive predictions leads to the drawback that simulation time increases as the subsystems have to take larger deviations and need more time to reach the target state.



**Figure 10:** Quantised prediction coherence for the intersection scenario for cell sizes  $c = 0.5$  (left) and with  $c = 1.0$  (right), both with prediction horizon length  $N = 4$  from Sprodowski et al. (2017). The prediction coherence is depicted individually for vehicles as long as they are approaching to their target and the cumulated sum of prediction coherences.



**Figure 11:** Quantised prediction coherence for the robotic scenario for cell sizes  $c = 0.5$  (left) and  $c = 1.0$  (right) both with horizon length  $N = 4$  from Sprodowski (2021). The prediction coherence is depicted individually for the robots as long as they are approaching to their target and the cumulated sum of prediction coherences.

**Algorithm 7** Preparation of communication data (robot  $p$ )

---

```

1: if  $n = 0$  then
2:   Set  $\mathcal{I}_p^c = \mathcal{I}_p^- := \emptyset$ 
3: end if
4: Compute  $\mathcal{I}_{p,n}$  via (19)
5: for  $k = 0 : N$  do
6:   if  $\mathcal{I}_{p,n,k} \notin \mathcal{I}_p^-$  then
7:     if  $\text{time}(\mathcal{I}_{p,n,k}, \mathcal{I}_p^-) \geq 0$  then
8:        $i := \text{find}(\text{time}(\mathcal{I}_{p,n,k}, \mathcal{I}_p^-))$ 
9:       Set  $\mathcal{I}_p^- := \text{replace}(\mathcal{I}_{p,i}, \mathcal{I}_{p,n,k}; \mathcal{I}_p^-)$ 
10:    end if
11:    Set  $\mathcal{I}_p^c := \text{append}(\mathcal{I}_p^c, \mathcal{I}_{p,n,k})$ 
12:    Set  $\mathcal{I}_p^c := \text{append}(\mathcal{I}_p^c, \mathcal{I}_{p,n,k})$ 
13:  end if
14: end for
15: Broadcast  $\mathcal{I}_p^c$ 
16: Set  $\mathcal{I}_p^- := \mathcal{I}_{p,n} \setminus \mathcal{I}_{p,n,0}, \mathcal{I}_p^c := \emptyset$ 

```

---

**4.1.3 Differential communication**

In Sprokowski et al. (2018a), the aspect of differential communication was examined, which is based on the quantisation of the communication. As the form of transmitting trajectories as integers allows to establish a simple differential update scheme, the communication load between the subsystem can be reduced efficiently. Hence, each subsystem has to store the broadcasted and received tuples with time instants  $n > 0$ . The broadcasting scheme is formulated in Alg. 7. The storage of the recent broadcasted tuples is denoted as  $\mathcal{I}_p^-$  for each subsystems  $p$ . On the communication side each robot computes the differential prediction  $\mathcal{I}_p^c$ , which consists of the tuples not yet communicated or not contained in the storage yet (see Alg. 7, line 11). To keep the space consumption of the procedure constant, the first tuple is removed at the end as in the next time instant this is not needed anymore. On the receiving side, each robot has to hold a storage for the received tuples to be able to recombine the comprehensive predictions from the other robots, which are differentially updated via the received  $\mathcal{I}_q^c$  with  $q \in [1: P] \setminus \{p\}$ . The procedure to assemble the predictions is defined in Alg. 8. At the beginning the robot has to store the whole prediction, as there has not an update arrived yet. Then, with  $n > 0$ , the received differential update  $\mathcal{I}_q^c$  from each robot has to be examined (Alg. 8, line 6), if the tuples have to be replaced (line 9), where the `time` function obtains the time instant of a tuple and the `replace` function updates the affected tuple. To this end, the DMPC scheme in Alg. 6 has to be modified for the broadcast and receiving such that in Alg. 6 lines 14 and 12 are replaced with

**Receive**  $\mathcal{I}_q^c$  for all  $q \in [1: P] \setminus \{P\}$

**Assemble**  $\mathbf{i}_{p,n}$  with Alg. 8

**Algorithm 8** Assembly of  $\mathbf{i}_{p,n}$  by robot  $p$ 


---

```

1: Receive  $\mathcal{I}_q^c$  for all  $q \in [1 : P] \setminus \{p\}$ 
2: if  $n = 0$  then
3:   Set  $\mathcal{I}_q^- = \mathcal{I}_q^c$  for all  $q \in [1 : P] \setminus \{p\}$ 
4: else
5:   for  $q \in [1 : P] \setminus \{p\}$  do
6:     for  $j = 0 : |\mathcal{I}_q^c| - 1$  do
7:       if  $\mathcal{I}_{q,j}^c \notin \mathcal{I}_q^-$  then
8:          $i := \text{find}(\text{time}(\mathcal{I}_{q,j}^c), \mathcal{I}_q^-)$ 
9:         Set  $\mathcal{I}_q^- := \text{replace}(\mathcal{I}_{q,i}^-, \mathcal{I}_{q,j}^c; \mathcal{I}_q^-)$ 
10:       end if
11:     end for
12:      $\mathcal{I}_q^- := \text{append}(\mathcal{I}_q^-, \mathcal{I}_q^c(|\mathcal{I}_q^c|))$ 
13:   end for
14: end if
15: Set  $\mathbf{i}_{p,n} := (\mathcal{I}_q^-)_{q \in [1:P] \setminus \{p\}}$  and  $\mathcal{I}_q^- := \mathcal{I}_q^- \setminus \mathcal{I}_{q,n,0}^c$  for all  $q \in [1 : P] \setminus \{p\}$ 

```

---

and line 19 with

**Broadcast**  $\mathcal{I}_p^c$  with Alg. 7.

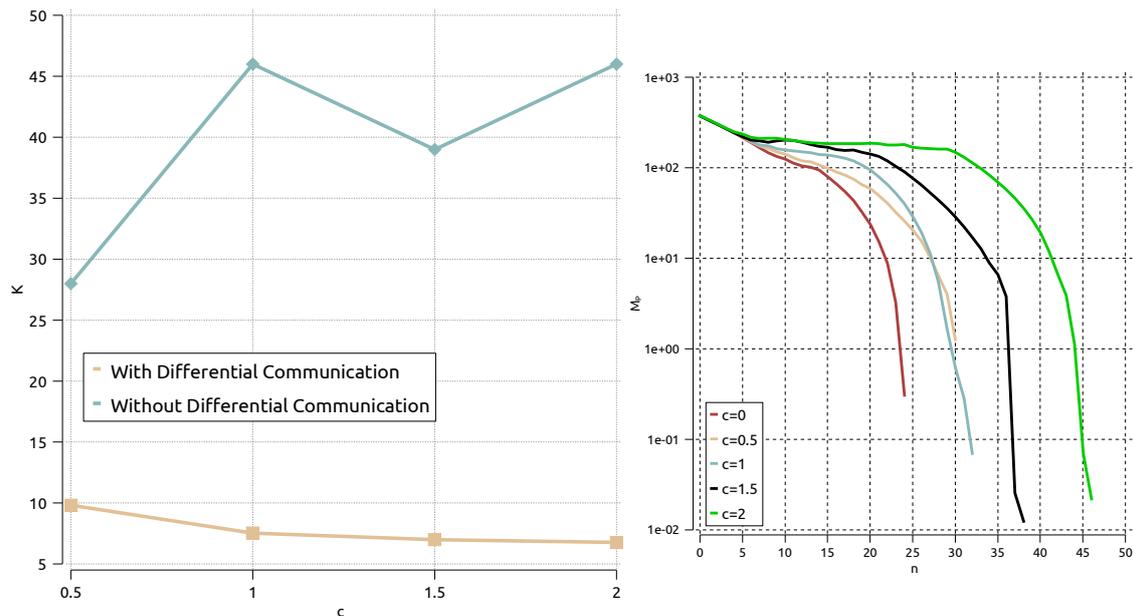
Simulations were carried out in Sprodowski et al. (2018a) for non-holonomic robots, which was already used in Mehrez et al. (2017) and additionally holonomic robots ( $P = 4$ ). To illustrate the reduction of the communication effort, we show an excerpt of the results from Sprodowski et al. (2018a). The number of communicated tuples over the simulation time for cell sizes  $c = \{0.5, 1.0, 1.5, 2.0\}$  are depicted in Fig. 12, where the number of tuples was averaged over the simulation time considering four holonomic robots with

$$K := \frac{1}{n_{\#}} \sum_{n=0}^{n_{\#}} \sum_{p=1}^P \#\mathcal{I}_p^c|_n. \quad (32)$$

The closed-loop  $\ell_p$  and cumulated closed-loop costs  $M_P(n)$ , optimised for non-holonomic settings and kept here for the holonomic for comparison Sprodowski et al. (2018a), were defined by

$$\ell_p(z_p, u_p) := \left\| \begin{pmatrix} (x_p - x_p^*)^2 \\ 5(y_p - y_p^*)^2 \\ (\theta_p - \theta_p^*)^2 \end{pmatrix} \right\|^2 + 0.2 \|u_p^2\|^2, \quad M_P(n) := \sum_{p=1}^P \ell_p(z_p(n), u_p(n)), \quad (33)$$

where  $\theta$  denotes the orientation of the robot. The individually assigned target condition for each robot  $p$  is denoted by  $z_p^*$ . The results illustrate the reduced communication effort of the differential communication method (see Fig. 12, left) as with an increasing cell size the number of tuples still reduces, which is also imposed by the fact that with larger cells the robots need more time to cross them and therefore the predictions do not changed as with



**Figure 12:** Scenario with 4 holonomic robots: averaged number of communication tuples  $K$  (left) with and without differential communication and the cumulated closed-loop costs  $M_p(n)$  for all cell sizes  $c$  over the time evolution  $n$  (right), from Sprodowski et al. (2018a).

smaller cell sizes. The closed-loop cumulated costs (see Fig. 12, right) show the suboptimality of the approach in comparison to the classical DMPC scheme without quantisation in terms of longer convergence time. While for smaller cell sizes  $c = \{0.5, 1.0\}$  the simulation execution times increases only slightly, for large cell sizes  $c = 2.0$  the simulation times nearly doubles as the robots have to take much longer detours to ensure the collision avoidance.

#### 4.1.4 Feasibility

Considering our modified DMPC scheme with quantisation, an important property to apply a MPC scheme reliable is the feasibility, where in the literature this is distinguished between *initial* and *recursive feasibility* (Richards and How, 2003; Primbs and Nevistić, 2000), respectively. As initial feasibility guarantees that an initial plan or initial states are feasible, for reaching the target is sufficient if terminal constraints are imposed as in Richards and How (2003), where the challenge is to find an initial feasible plan. As no terminal conditions (e.g. endpoint constraints or terminal costs) are imposed here for the system, we need to ensure recursive feasibility: This means that the feasibility has to hold in each time instant for the successive state of the system that the constraints are satisfied. For any admissible and feasible state, a feasible control input shall exist. Therefore, we have to ensure that the control sequence is not empty. Regarding the non-cooperative setting here, satisfying the constraints is the property to ensure. Hence, in Sprodowski et al. (2018a) this property uses the Assumption 4.1, where it is stated, that such a control can be found ( $\bar{u}_p$ ), which leads to an immediate hold, i.e. the system is able to either stop immediately or hold its position.

For an arbitrary chosen system it was shown by induction that such a control exists and can be calculated incorporating the received occupancy grid  $i_p$  from all the other robots. Additionally, this also validates that the algorithm does not terminate unexpectedly, as in every time instant  $\bar{u}_p$  is available.

### 4.1.5 Interval communication

In this section, we explore the reduction of communication beyond differential updates. The former communication pattern (occupancy grid, differential updates) is still influenced by the underlying dynamics, which may lead to a larger communication load if each robot has to take detours imposing a larger difference on the predictions in two successive time instants. Therefore, we introduce here a method, which is independent from the dynamics of the robot using a bounding box around the predicted trajectory. The idea stems from the Interval Superposition Arithmetic Principle (Zha and Houska, 2016), where addition theorems and factorial functions properties are used to exploit the dynamics or image set of a function. Here, this idea is adapted for the communication between the robots to reduce the necessary communication load for every robot in any time instant to 2 cells (Sprodowski et al., 2018c).

This is achieved via a bounding box, which takes the minimum and maximum coordinate in each dimension of the trajectory to be enclosed. Here, the collision avoidance is w.l.o.g. defined for the 2D plane, e.g.  $z = (x, y)^\top$  for a robot, while additional dimension, which might be used for orientation, are not considered. Then, the bounding box is defined using the trajectories maximum and minimum cells of the dimension in the 2D plane, which forms a rectangle defined by

$$\mathbb{X}_p = \left( \left( \min_{a_p} \{C_p\}, \min_{b_p} \{C_p\} \right), \left( \max_{a_p} \{C_p\}, \max_{b_p} \{C_p\} \right) \right) \quad (34)$$

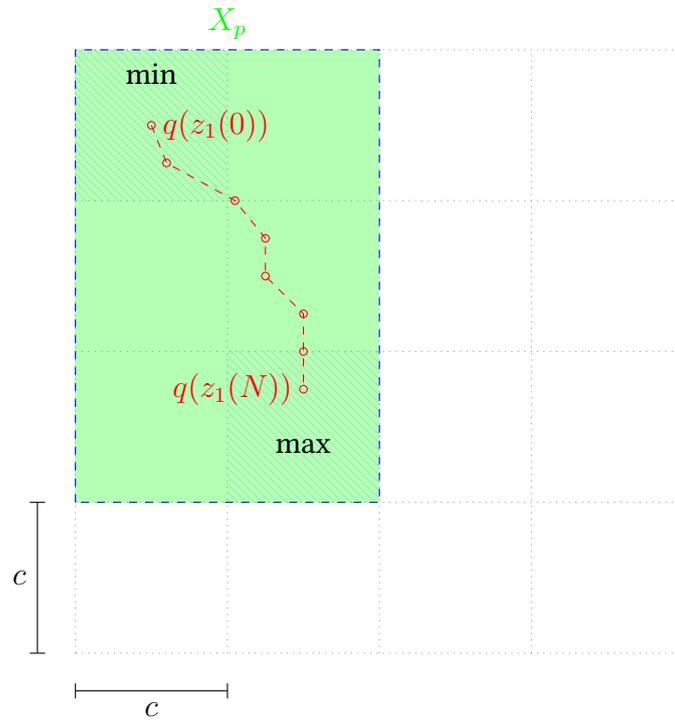
with

$$C_p := \left\{ (a_p, b_p) \mid \left( q \left( z_p^u(k; z_p^0) \right) \right)_{k=0}^N \right\}$$

as the given trajectory to be bounded. For an illustration using an example trajectory with start and end point  $z_1(0)$  and  $z_1(N)$  see Fig. 13. When a bounding box is received from another robot  $q$ , the collision avoidance constraints have to be constructed using the following Alg. 9. Then, incorporating this communication method into the DMPC scheme yields the modifications in Alg. 6 in lines 5 and 19 with

$$\mathbf{Construct} \mathbb{X}_p \text{ via (34)} \quad (35)$$

$$\mathbf{Broadcast} \mathbb{X}_p$$



**Figure 13:** Illustration of a bounding box (green) declared by min/max cells (blue dashed) for a trajectory with quantised start and endpoints  $q(z_1(0))$  and  $q(z_1(N))$ .

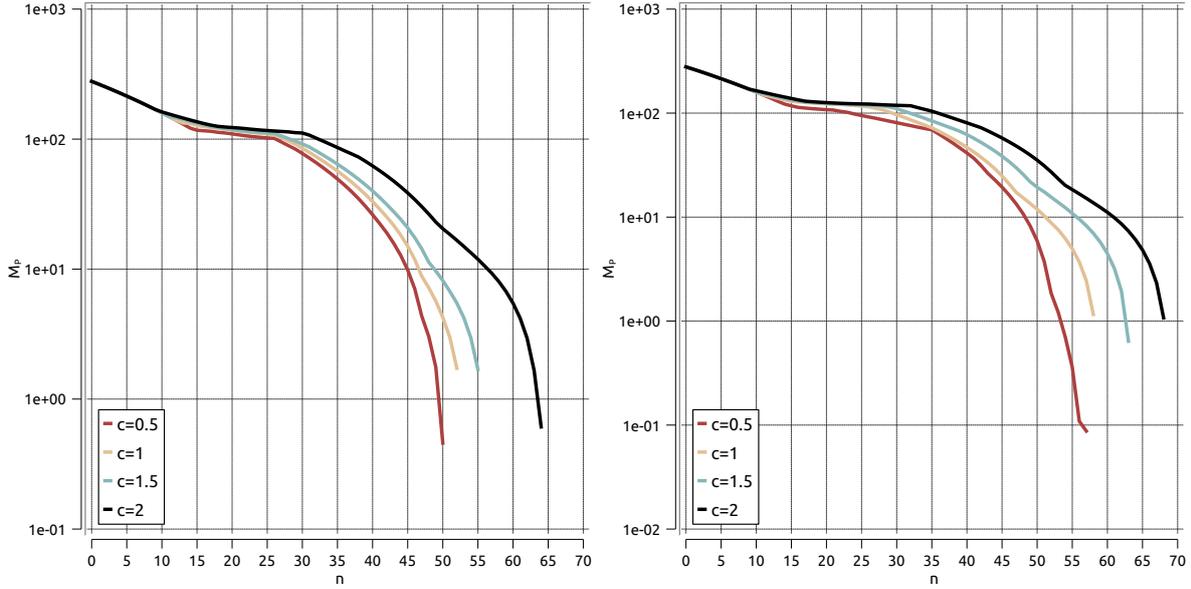
---

**Algorithm 9** Occupancy grid construction for robot  $p$  by received occupancy tuples from robots  $q \in [1: P] \setminus \{p\}$

---

**Require:**  $\mathbb{X}_q$

- 1: **Set**  $a_{\min} = \min_a \{\mathbb{X}_q\}$ ,  $a_{\max} = \max_a \{\mathbb{X}_q\}$ ,  $b_{\min} = \min_b \{\mathbb{X}_q\}$ ,  $b_{\max} = \max_b \{\mathbb{X}_q\}$
  - 2: **for**  $a = a_{\min}$  to  $a_{\max}$  **do**
  - 3:     **for**  $b = b_{\min}$  to  $b_{\max}$  **do**
  - 4:         **for**  $k = n$  to  $n + N$  **do**
  - 5:             **Append**  $\mathcal{I}_{q,n} := \mathcal{I}_{q,n} \cup (k, a, b)$
  - 6:         **end for**
  - 7:     **end for**
  - 8: **end for**
  - 9: **Set**  $i_{p,n}$  according to (20)
-



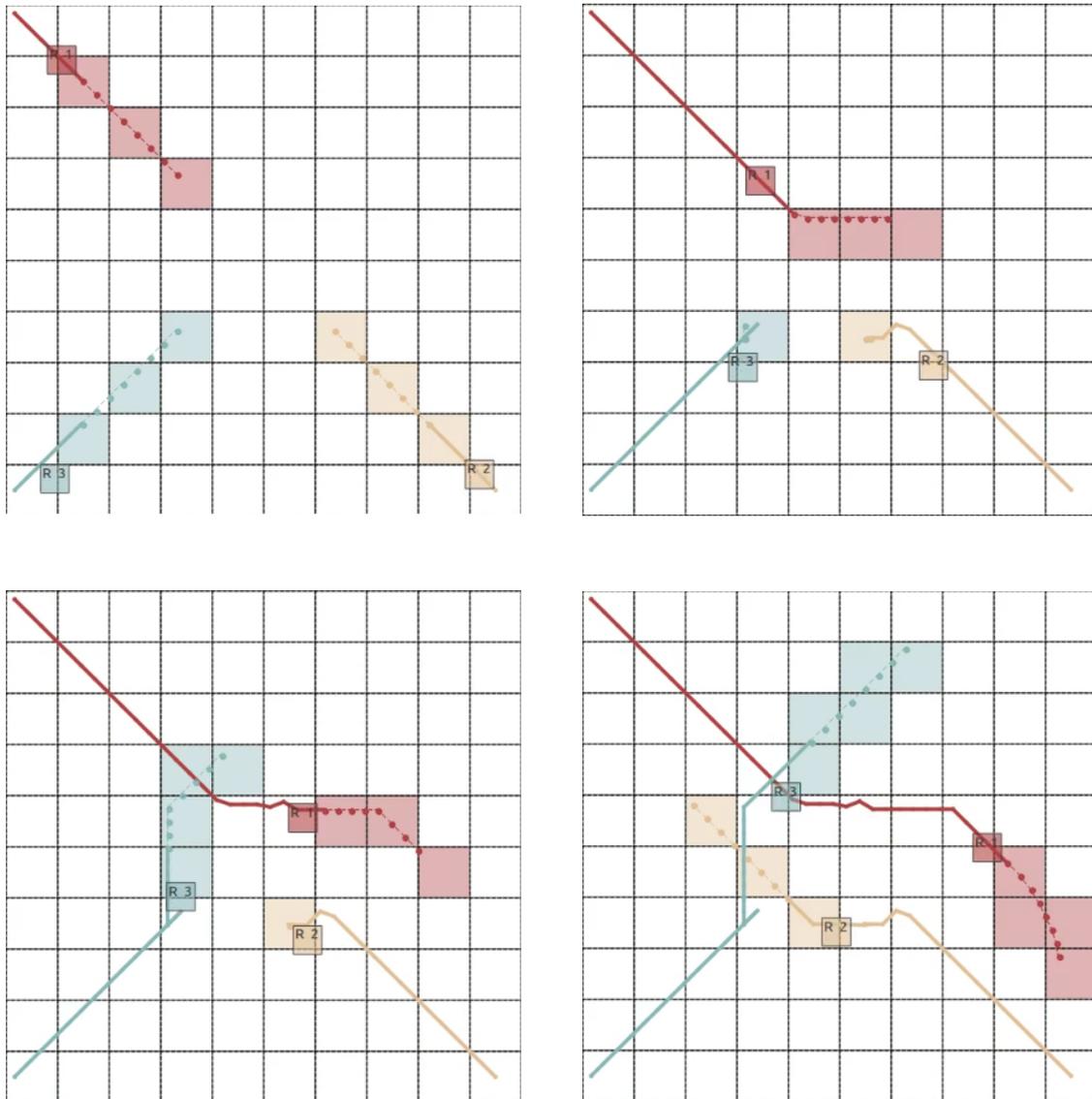
**Figure 14:** Development of  $M_P$  with differential communication (left) and interval communication (right) using 3 robots and horizon length  $N = 8$ , from Sprodowski et al. (2018c).

for broadcasting the interval and replacing in Alg. 6 lines 12 and 14 with

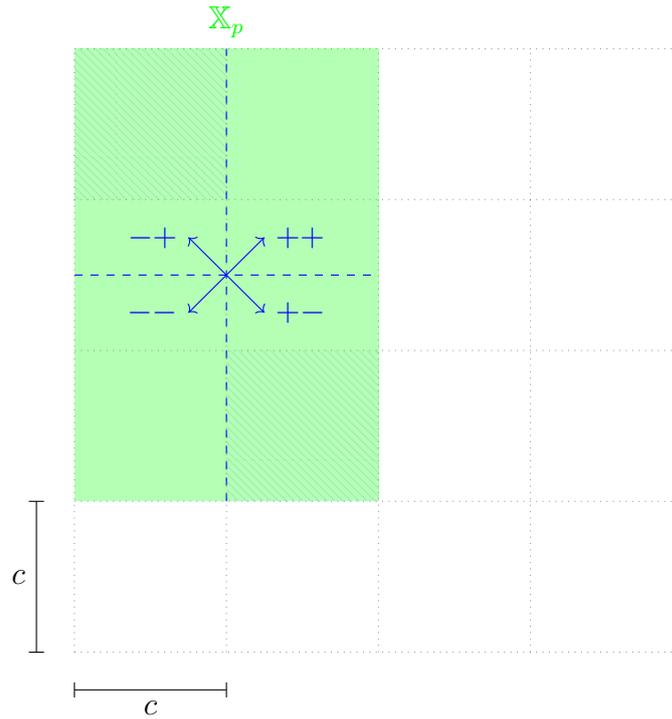
$$\begin{aligned} & \mathbf{Receive} \ \mathbb{X}_p & (36) \\ & \mathbf{Construct} \ \mathcal{I}_q \text{ via Alg. 9} \end{aligned}$$

to obtain from the interval the bounding box to construct the collision avoidance constraints in (16). The communication load of the quantised prediction, which is communicated in each time instant, is reduced to two cells per time instant. Each robot evaluates the received *occupancy tuples* from the other robots the minimum and maximum values, revealing the bounding box the other robots claim, and reconstruct the *occupancy grid* (Sprodowski et al., 2018c). It should be noted that the usage of space is much higher than in the conventional methods before, as the rectangle surrounds the full trajectory of the robot and many cells might be unused.

Simulations were carried out with  $P = 3$  robots as the bounding boxes consumed much more operational space and a setting with  $P = 4$  was infeasible. To allow comparison with the differential communication scheme from Section 4.1.3, time evolutions of cumulated closed-loop costs (33) from both approaches are illustrated in Fig. 14. From the results it can be concluded that the interval communication leads to a larger consumption of operational space and hence, the robots have to take longer detours. However, as the bounding boxes forces the robots to take detours in a straight line, the convergence time does not increase that much. A deeper insight is given by illustrating the trajectories using the interval communication scheme in Fig. 15. The trajectories show that the robots are forced to obey the bounding box, which can be clearly noted for robot 1 in Fig. 15 (right top), which has to take the detour in a straight line because of both bounding boxes from robot 2 and 3. In the later time instants robot 2 is urged to wait as from the left and upper side (left bottom)



**Figure 15:** Time evolution of three robots using the interval communication scheme with solid line as path, dashed lines with points as predictions, and coloured cells as occupied cells for time instants  $n = \{5, 15, 24, 32\}$ .



**Figure 16:** Illustration of usage of signs for  $(a, b)$ , depending on the direction (right), from Sprodowski and Pannek (2018).

robot 3 and 1 are leaving no free cells with their bounding boxes. Hence, as with this limited number of robots the simulations converges still quite fast in comparison to the differential communication scheme, it is clear that the space consumption would not allow many more systems to participate.

To attenuate the drawback of the space consumption and also keep the low communication effort, we proposed an idea by modifying the communication exchange without additional transmission load in Sprodowski and Pannek (2018). As in Section 4.1.1 the quantisation is defined as  $\mathcal{G} := [1, \dots, a_{\max}] \times [1, \dots, b_{\max}]$  and hence, only positive integers are used for transmission, the leading signs ( $\pm$ ) are unused and allows the usage for further information without increasing the communication load in terms of a larger word size or more cells to communicate. The leading signs are used here to indicate the direction of the trajectory of a system, which includes information, which cells are used immediately ( $k = 0$ ) and which ones might be used at the end of the prediction horizon ( $k = N$ ). We utilised therefore the coordinates of the first cell of  $\mathbb{X}_p$ , which are encoded such that the direction can be determined in the 2D space as indicated in Fig. 16. The directions are encoded as “left bottom to right top” as  $++$ , “left top to right bottom” as  $+-$ , “right bottom to left top” as  $-+$  and

“right top to left bottom” as  $--$ . Then, the extended bounding box to be communicated is defined for one subsystem  $p$  as

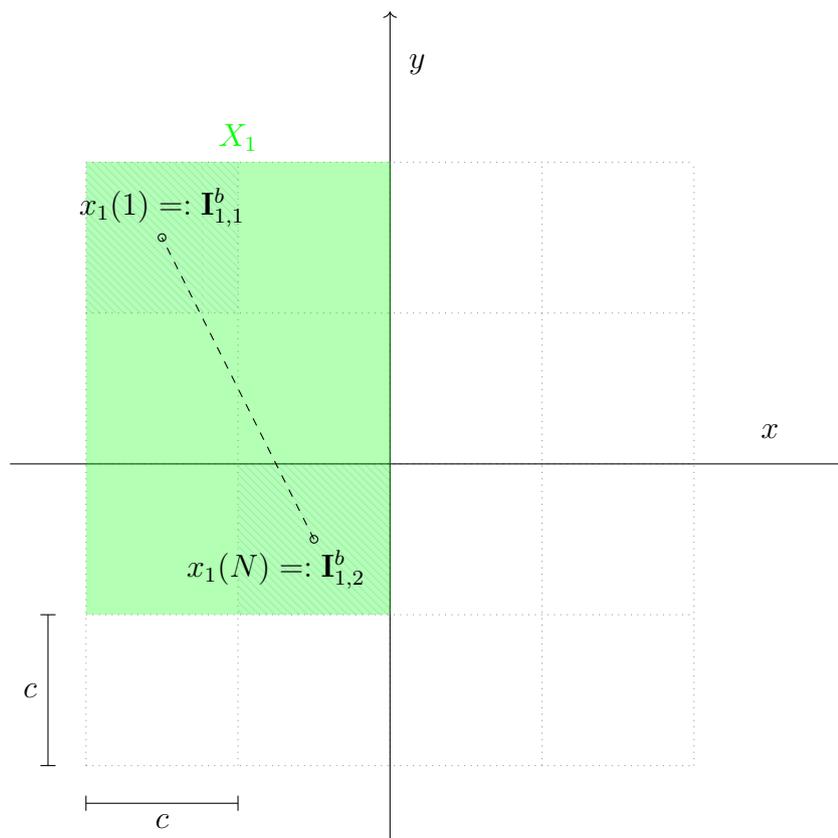
$$\begin{aligned} \mathcal{I}_p^b := \mathbb{X}_p &= \left( \underbrace{\left( \pm \min_{a_p} \{C_p\}, \pm \min_{b_p} \{C_p\} \right)}_{\text{direction}}, \left( \max_{a_p} \{C_p\}, \max_{b_p} \{C_p\} \right) \right) \\ &= \left( \underbrace{(\pm a_1, \pm b_1)}_{\text{direction}}, (a_2, b_2) \right) \end{aligned} \quad (37)$$

and extended to a 2-tuple with  $(0, (a_1, b_1)) =: \mathcal{I}_{p,1}^b$  and  $(0, (a_2, b_2)) =: \mathcal{I}_{p,2}^b$ . Note that the values of indices of the tuples are still interpreted as strictly positive letting  $C_p(-2, -2) = C_p(-2, 2) = C_p(2, 2) = C_p(2, -2)$  defined as the same cell. To evaluate the correct start and ending positions of the robot, the leading signs have to be examined to extract the received occupancy tuples by a robot  $q$  as

$$\mathcal{I}_q^b := \begin{cases} (\mathbb{X}_q), & \text{direc}(\mathbb{X}_q) = (1, 1) \\ (\mathbb{X}_{q,2}, -\mathbb{X}_{q,1}), & \text{direc}(\mathbb{X}_q) = (-1, -1) \\ ((\mathbb{X}_{q,1,1}, \mathbb{X}_{q,2,2}), (\mathbb{X}_{q,2,1}, -\mathbb{X}_{q,1,2})), & \text{direc}(\mathbb{X}_q) = (1, -1) \\ ((\mathbb{X}_{q,2,1}, \mathbb{X}_{q,1,2}), (-\mathbb{X}_{q,2,1}, \mathbb{X}_{q,1,2})), & \text{direc}(\mathbb{X}_q) = (-1, 1) \end{cases} \quad (38)$$

with  $\text{direc}(\mathbb{X}_q) = (\text{sgn}(\mathbb{X}_{q,1,1}), \text{sgn}(\mathbb{X}_{q,1,2}))$ .

Here,  $\mathbb{X}_{q,i,j}$  returns from the  $i^{\text{th}}$  tuple the  $j^{\text{th}}$  value. Then, the start and end points can be denoted as  $z_q^u(n; z_q^0) := q(\mathcal{I}_{q,1}^b)$  and  $z_q^u(n+N; z_q^0) := q(\mathcal{I}_{q,2}^b)$ , respectively. For an illustration of the reconstruction, see Fig. 17 illustrating the execution of (38). With the knowledge of the direction of one robot this is not sufficient to evaluate, which cell is free to be used by the other robots. Some more properties are required: At first, the robots should share the same dynamics or the dynamics has to be known to all robots. Second, the trajectories must not be intertwined as this would lead robot to revisit former assigned cells. Hence, the cost function has to be monotonous decreasing with respect to the defined target position. Third, from the size of the bounding box it can be estimated, at which speed (i.e. input) the robots might cross the assigned cells considering the mean. Nevertheless, the mean value is not sufficient as a robot may vary its speed: Consider that a robot evaluated the optimal control to let pass another robot and then might continue the trajectory. Then, it could either slow down or even hold until the collision avoidance constraints allow to continue on the (sub)optimal trajectory. Hence, a lower bound has to be formulated, when the cells are free to be used. From practical reasons, if a robot has assigned a large bounding box, which could be traversed by a maximum input (maximum speed) only, the cells at the



**Figure 17:** Reconstruction of the movement of one robot (green cells) by (38) based on received  $X_p$  (dashed cells), for example here  $((+a_1, -b_1), (a_2, b_2))$ , from Sprodowski (2021).

start point of the trajectory would be crossed by this robot in an early time instant and are free to be used by other robots, later. Hence, in Sprodowski and Pannek (2018, Algorithm 2) the procedure enables to calculate the minimum release time of such a reserved cell, which was formalised in a theorem in Sprodowski (2021), shortly given below:

**Theorem 4.2** (Minimum cell release time over  $N$ )

Consider two arbitrary robots  $p, q, p \neq q, p, q \in [1: P]$  with  $z_p, z_q \in \mathbb{X}$  following underlying model (1). Then, for a minimum cell size  $\underline{c}$  given by (22), a finite horizon length  $N > 1$  with  $n = 1, 2, \dots$  and  $k \in [n: n + N]$  the minimum cell release time is lower bounded by

$$\begin{aligned} & \max \left\{ \left\| h \left( \mathcal{I}_{q,2}^b \right) - h \left( \mathcal{I}_{q,1}^b \right) \right\|_{\infty} - (k - 1) \underline{c}, \right. \\ & \quad \min \left\{ \left\| h \left( \mathcal{I}_{q,2}^b \right) - h \left( \mathcal{I}_{q,1}^b \right) \right\|_{\infty}, \left\| h \left( \mathcal{I}_{q,2}^b \right) + (N - k) \underline{c} \right\|_{\infty} \right\} \\ & \leq \left\| h \left( \mathcal{I}_{p,n,k} \right) - h \left( \mathcal{I}_{q,2}^b \right) \right\|_{\infty}. \end{aligned} \quad (39)$$

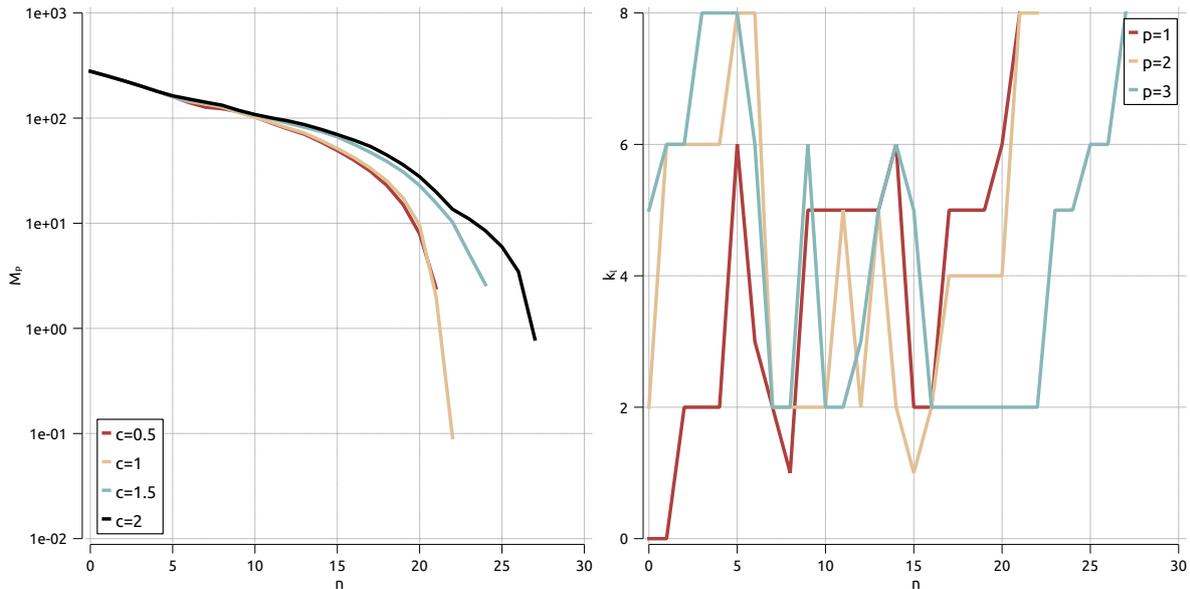
This theorem formulates the lower bound when such a cell  $\mathcal{I}_{p,n,k}$  can be used by robot  $p$ . The inner minimum function ensures that the stated bounding box is not becoming larger if only a small box is assigned. Then, with an increasing  $k$ , the minimum function reveals the shrinking second argument regarding a minimum distance to the target cell depending on  $k$ . The outer maximum function is subject to the first argument, which ensures for  $k = 1$  that the bounding box is satisfied, while for increasing  $k$  the minimum function returns the second argument, i.e. the shrinking bounding box depending on  $(N - k)$ , which guarantees that the robot is inside the shrinking bounding box by taking the maximum speed. The proof of this theorem was presented in Sprodowski (2021).

From that theorem, the minimum cell release time can be calculated via

$$k_l := \left\| h \left( \mathcal{I}_{q,2}^b \right) - N \underline{c} \right\|_2 - \left\| h \left( \mathcal{I}_{q,2}^b \right) - h \left( \mathcal{I}_{q,1}^b \right) \right\|_2, \quad (40)$$

which states, at which time instant  $n \leq k_l \leq n + N$  a cell is released by robot  $q$ , assuming that the robot moves in a straight way there with maximum speed. As non-holonomic robotic models, e.g. tricycles, vehicles, would need more time instants with possible additional turn-arounds and movement constraints to obey, they have to leave at an earlier time instant than with the assumed holonomic robotic model. Consequently, the holonomic model is a “worst-case” scenario as this is the last time instant a holonomic robot is able to reach the predicted end point at  $n + N$ .

The results were applied in the DMPC algorithm in Sprodowski (2021) showing shorter convergence times over Sprodowski et al. (2018c) with the same setting presented in Fig. 18. In Fig. 18 (left) for all cell sizes the relaxed constraint to use the cells earlier leads to shorter convergence times. For the minimum release time  $k_l$ , exemplary shown for cell size  $c = 2.0$ , the behaviour of the robots can be easily recognised based on this value: The first robot ( $p = 1$ ) is going with maximum speed, hence the minimum release time for the first cell is early as the robot is optimising at first and is capable to use the maximum control input to move to the target point. For the other robots  $p = \{2, 3\}$  these release times are higher as they have to incorporate the prediction and assigned cells from robot 1. At later time



**Figure 18:** Time evolution over closed-loop costs (left) over cell sizes  $c = \{0.5, 1.0, 1.5, 2.0\}$  (left) and minimum cell release time (40) for each robot with cell size  $c = 2.0$  (right) and  $N = 8$ , from Sprodowski and Pannek (2018).

instants, the release time also for robot 1 increases as all robots have to slow down to consider collision avoidance constraints. At the end, the time decreases nearly to the maximum  $N$  due to reached target, which is mostly close by. The excerpt of the results show that the convergence times are shorter with the relaxed condition and also the robots have to wait and slow their speed down, it is worth the effort to incorporate this underlying dynamics.

In the next section, the last aspect, convergence of the distributed system will be analysed, which could not be handled via the “classical tools” using terminal costs, constraints or Lyapunov properties. As this DMPC scheme is formulated without terminal costs or constraints, only initial and recursive feasibility was shown. Recursive feasibility only guarantees that the scheme does not terminate unexpectedly and in every time instant a solution can be found, but nevertheless, that every robot reaches the assigned target, could not sufficiently shown yet with feasibility.

## 4.2 Convergence of the distributed system

The approach to show convergence for the whole system cannot rely only on properties of the optimiser or the DMPC scheme aspects itself as the scheme is non-cooperative: the subsystems could get caught in local minima, which could result in deadlock, where subsystems are blocking each other in the operational space. Hence, to lower the strong requirements for sufficient condition as they are made with terminal constraints, that a closed, feasible path from start point  $z_p^0$  to target point  $z_p^*$  exists, an additional property, which lifts the burden of strong requirements and prevents the system from deadlocks, is derived from the idea of traffic rules: *A temporary roundabout*. The idea is to establish an additional constraint, which convert the non-cooperative control scheme temporary in a

cooperative control scheme as it is done also in a real roundabout (Sprodowski and Pannek, 2020). We make the following additional assumptions considering initial feasible conditions and target points, which shall be allowing boundary values to be reachable for the robots, as stated as follows:

**Assumption 4.3** (Feasible initial conditions and targets)

Suppose a set of  $P$  robots with underlying model (1), state (9) and control constraints (8). Then, for each robot  $p \in [1: P]$  with  $\hat{z}_j \in \{z_j^0, z_j^*\}$ ,  $j \in \{p, q\}$  and  $\hat{z}_j \in \mathbb{X}$  and for all  $p, q, q \in [1: P] \setminus \{p\}$  and for all  $\hat{z}_p$  a non-empty set  $\mathbf{M}(\hat{z}_p) \subset \mathbb{X}$  exists that

$$\mathbf{M}(\hat{z}_p) := \left\{ (a, b) \mid (a, b) \in \mathcal{G}, \|\hat{z}_p - q((a, b))\|_2 \leq 2\bar{\Psi} + c \right\} \quad (41)$$

and

$$\hat{z}_p \notin \mathbf{M}(\hat{z}_p), \forall q \in [1: P] \setminus \{p\}. \quad (42)$$

This assumptions ensures that the initial conditions and target points of two arbitrary robots  $p$  and  $q$  are yielding a safety margin, which is at least  $2\bar{\Psi} + c$  derived from (26). Then, each robot regards the distance between the predictions of each other by

$$\|h(\mathcal{I}_{p,n,N}) - h(\mathcal{I}_{q,n,N})\|_2 \leq P\bar{\Psi} \quad (43)$$

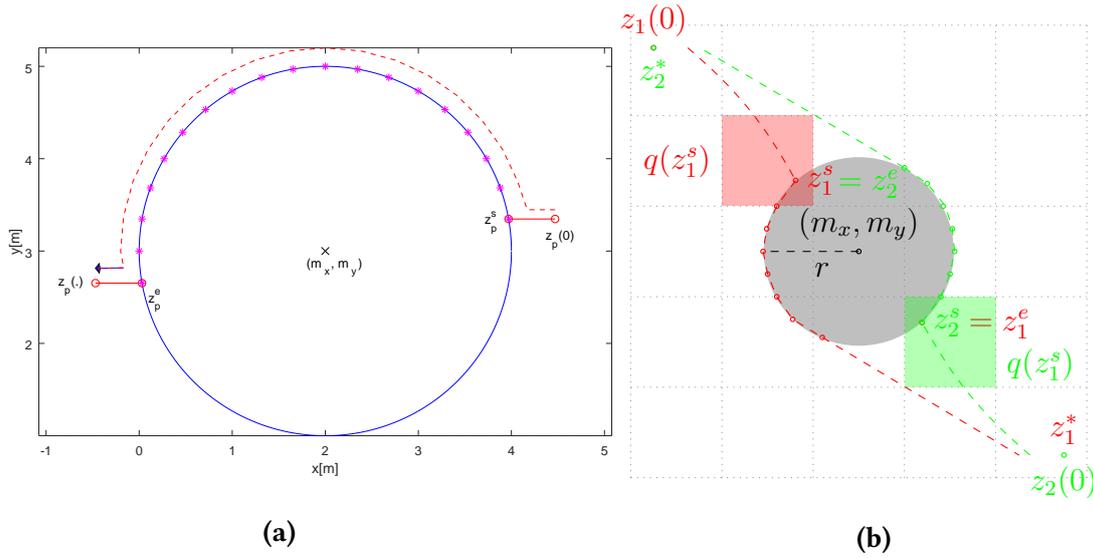
with  $p, q \in [1: P]$  and  $p \neq q$ . The construction of the circle is then calculated based on the middle point of all affected robots, which satisfy (43) based on their last predictions points  $\mathcal{I}_{j,n,N}$  utilising the minimum and maximum prediction points ( $m_x^{\min} = \min_{x_j} \mathcal{I}_{j,n,N}$ ,  $m_y^{\min} = \min_{y_j} \mathcal{I}_{j,n,N}$ ,  $m_x^{\max} = \max_{x_j} \mathcal{I}_{j,n,N}$ ,  $m_y^{\max} = \max_{y_j} \mathcal{I}_{j,n,N}$ ) to obtain with  $j \in \{p, q\}$

$$(m_x, m_y) = \begin{pmatrix} 0.5 (m_x^{\max} - m_x^{\min}) \\ 0.5 (m_y^{\max} - m_y^{\min}) \end{pmatrix}. \quad (44)$$

For the detailed trigonometric calculation of the middle points regarding all affected robots and interception points of each robot with the circle ( $(\bar{x}_{p,1}, \bar{y}_{p,1})^\top, (\bar{x}_{p,2}, \bar{y}_{p,2})^\top$ ) see Sprodowski and Pannek (2020). Then, as most of the collisions happens in the centre of the operational space (see Fig. 9 (right)), following assumption is required for a circle free from any initial conditions or target points:

**Assumption 4.4**

Suppose that for a circle with centre  $(m_x, m_y)$  and radius  $r = \frac{P\bar{\Psi}}{2\pi}$  for all  $z_p^*$ ,  $p \in [1: P]$ , the conditions  $\left| (m_x, m_y)^\top - z_p^* \right| > r$  and  $\left| (m_x, m_y)^\top - z_p^0 \right| > r$  holds.



**Figure 19:** Illustration of the circular control with centre  $(m_x, m_y)$ , and entry and exit points  $z_p^s, z_p^e$  for one robot (left) and two robots with start and endpoints  $z_1^s, z_2^s$  and  $z_1^e, z_2^e$  exiting earlier due to tangential condition (right), from Sprodownski and Pannek (2020).

Then, the interception points (entry  $z_p^s$  and exit  $z_p^e$ ) of the trajectory of the robots and the circle are calculated by

$$z_p^s := \begin{pmatrix} x_p^s \\ y_p^s \end{pmatrix} = \begin{cases} (\bar{x}_{p,1}, \bar{y}_{p,1})^\top, & \left\| z_p(N-1) - (\bar{x}_{p,1}, \bar{y}_{p,1})^\top \right\|_2 < \left\| z_p(N-1) - (\bar{x}_{p,2}, \bar{y}_{p,2})^\top \right\|_2 \\ (\bar{x}_{p,2}, \bar{y}_{p,2})^\top, & \text{else} \end{cases} \quad (45)$$

and

$$z_p^e := \begin{pmatrix} x_p^e \\ y_p^e \end{pmatrix} = \begin{cases} (\bar{x}_{p,2}, \bar{y}_{p,2})^\top, & \left\| z_p(N-1) - (\bar{x}_{p,2}, \bar{y}_{p,2})^\top \right\|_2 < \left\| z_p(N-1) - (\bar{x}_{p,1}, \bar{y}_{p,1})^\top \right\|_2 \\ (\bar{x}_{p,1}, \bar{y}_{p,1})^\top, & \text{else} \end{cases} \quad (46)$$

as entry and exit points, respectively. Here, we assume that the robots are traversing the circle and their targets are lying on the opposite sides of the circle. Depending on the individual The radius is fixed with  $r = \frac{P\bar{\Psi}}{2\pi}$ , which keeps the entries and exit points feasible guaranteed by (25). Then, to reveal the circle for the robots, the control law is then switched to

$$u_p(n) = r \begin{pmatrix} \cos \left( \arccos \left( \frac{z_p(n) - m_x}{r} \right) + u_p^c \right) \\ \sin \left( \arccos \left( \frac{z_p(n) - m_x}{r} \right) + u_p^c \right) \end{pmatrix} \quad (47)$$

to keep the robots on a circular trajectory for  $0 \leq u_p^c \leq \underline{c}$  describing the speed of the robot  $p$  on the circle. An illustration of the applied circle law is given in Fig. 19. Each robot evaluates

the distances to the other robots based on (43) to switch the control law to establish a circular control law, where the procedure is defined in Alg. 10. Here, for feasibility reasons and

---

**Algorithm 10** Evaluation for switching control law of robot  $p$  to establish a collision avoidance circle

---

```

1: for all  $p \in [1: P]$  do
2:   if  $\exists q \in [1: P] \setminus \{p\}$  where (43) holds for  $p, q$  then
3:     Set  $\mathbf{Q} := \mathbf{Q} \cup \{p, q\}$ 
4:   end if
5: end for
6: if  $\mathbf{Q} \neq \emptyset$  or  $w_p = true$  then
7:   if  $w_p = false$  then
8:     Set centre  $(m_x, m_y)$  for  $z_p$  with (44) and  $z_p^s, z_p^e \forall p \in \mathbf{Q}$  with (45) and (46)
9:   end if
10:  Set  $w_p := true$ 
11:  for all  $p \in [1: P]$  and  $p \notin \mathbf{Q}$  do
12:    Keep  $u_p = \bar{u}_p \quad \forall k \in [n: n + N]$  according to Assumption 4.1
13:  end for
14:  if  $z_p^0 = z_p^s$  and  $p \in \mathbf{Q}$  then
15:    Set  $u_p$  according to (47)
16:  end if
17: end if
18: if  $w_p = true$  and  $\left( (z_p^0 - z_p^*) \times (z_p^0 - (m_x, m_y)^\top) = 0 \right)$  or  $z_p^0 = z_p^e$  for  $p \in [1: P]$  then
19:   Set  $u_p$  back to (8)
20:   Remove  $\mathbf{Q} := \mathbf{Q} \setminus \{p\}$ 
21:   Set  $w_p := false$ 
22: end if
23: if  $|\mathbf{Q}| \leq 1$  then
24:   for all  $p \in [1: P]$  do
25:     Reset  $u_p$  according to (8)
26:     Set  $w_p := false$ 
27:   end for
28: end if

```

---

preventing deadlocks, a set  $\mathbf{Q}$  is created, which adds all robots satisfying condition (43). Then the centre of the circle is evaluated, which is guarded by  $w_p$  to prevent the centre point from shifting while the robots establish the circle (Alg. 10, line 6). All robots  $p \notin \mathbf{Q}$  not participating in this circle are urged to stop via imposing the control law  $\bar{u}_p$  (line 12). For each measurement it is checked if the exit point of the circle is reached to switch the control law back (line 12), which is shortened by a tangential examination additionally (line 19). For an illustration of the tangential law, see Fig. 19b. The control law is then switched back and the robot moves along its original control law. To guarantee Assumption 4.1 for an

immediate hold, while the robots are under the circular control regime (47), the following definition of

$$\begin{aligned}\bar{u}_p^c &:= \cos \left( \arccos \left( \frac{z_p(n) - m_x}{r} \right) + u_p^c \right) = 0 \\ \bar{u}_p^c &:= \sin \left( \arccos \left( \frac{z_p(n) - m_x}{r} \right) + u_p^c \right) = 0\end{aligned}$$

ensures that Assumption 4.1 can be satisfied. To integrate this scheme into the DMPC algorithm, in Alg. 6, after line 16 the following statement has to be inserted:

**Call Alg. 10**

From Alg. 10, we state the following theorem:

**Theorem 4.5** (Collision prevention via a fixed circular control law)

*For a number of  $P$  robots, each following (7), with restriction on state and control (9) and (8) suppose that initial feasible conditions and targets fulfilling Assumptions 4.4 and 4.3 hold. If the distance between two robots is such that (43) is fulfilled and Assumption 4.4 is satisfied for all robots  $p \in [1: P]$ , then the calculation of a circle and execution of the DMPC scheme in Alg. 6 applying the control regime by Alg. 10 will allow the robots to let them converge to their targets, i.e. steer the system practically stable.*

The proof of this theorem was shown in Sprodowski and Pannek (2020), where also a variance of this scheme using a flexible circle depending on the number of participating robots was used. Therein, the radius of the circle is then set to

$$r = (x - m_x)^2 + (y - m_y)^2 = \left( \frac{|\mathbf{Q}| \bar{\Psi}}{2\pi} \right)$$

while the algorithm follows the same scheme as presented in Alg. 10.

A further requirement to prevent robots from deadlocks is to provide a sufficient prediction horizon length to avoid the situation that robots could not circumvent each other because the advantage of taking a detour is not recognisable via the cost function and the chosen horizon. Hence, to obtain a sufficient lower bound in this quantised setting, in the next section we present a method utilising the underlying quantisation.

### 4.2.1 Sufficient prediction horizon length

In this section, Assumption 4.3 is modified by using a stronger condition, which states that initial and target conditions have to be distinguished:

**Assumption 4.6** (Disjunct initial conditions/targets)

Suppose a set of  $P$  robots modelled by the underlying dynamics (7), restricted by state (9) and control constraints (8). For each robot  $p \in [1 : P]$  let  $z_p^0, z_p^* \in \mathbb{X}$  and for all  $p, q \in [1 : P]$  with  $p \neq q$  and  $\hat{z}_j \in \{z_j^0, z_j^*\}$  with  $j \in \{p, q\}$  we assume

$$g_{q,0}^p(\hat{z}_p, q(\hat{z}_q)) \geq 2\bar{\Psi} + c, \quad (48)$$

and for  $\hat{z}_p = (x_p, y_p)^\top$

$$-\bar{x} + 2\bar{\Psi} + c \leq x_p \leq \bar{x} - (2\bar{\Psi} + c), \quad -\bar{y} + 2\bar{\Psi} + c \leq y_p \leq \bar{y} - (2\bar{\Psi} + c) \quad (49)$$

shall hold.

This assumption ensures that all initial conditions and target points are not ensuring only (25) is satisfied, but additionally another robot could utilise the space between such two initial conditions or target points and still be feasible. As our calculation of a sufficient prediction horizon length is based on the cost function, some requirements have to be taken into account for an appropriate choice of such a stage cost functional:

**Definition 4.7** (Stage cost function)

Let  $\ell_p: \mathbb{X} \times \mathbb{U} \rightarrow \mathbb{R}_{\geq 0}$  be a positive definite, continuous differentiable function with

$$\ell_p(z_p, u_p) := \|d_z(z_p, z_p^*)\|_2 + \lambda \|d_u(u_p, u_p^*)\|_2 \quad (50)$$

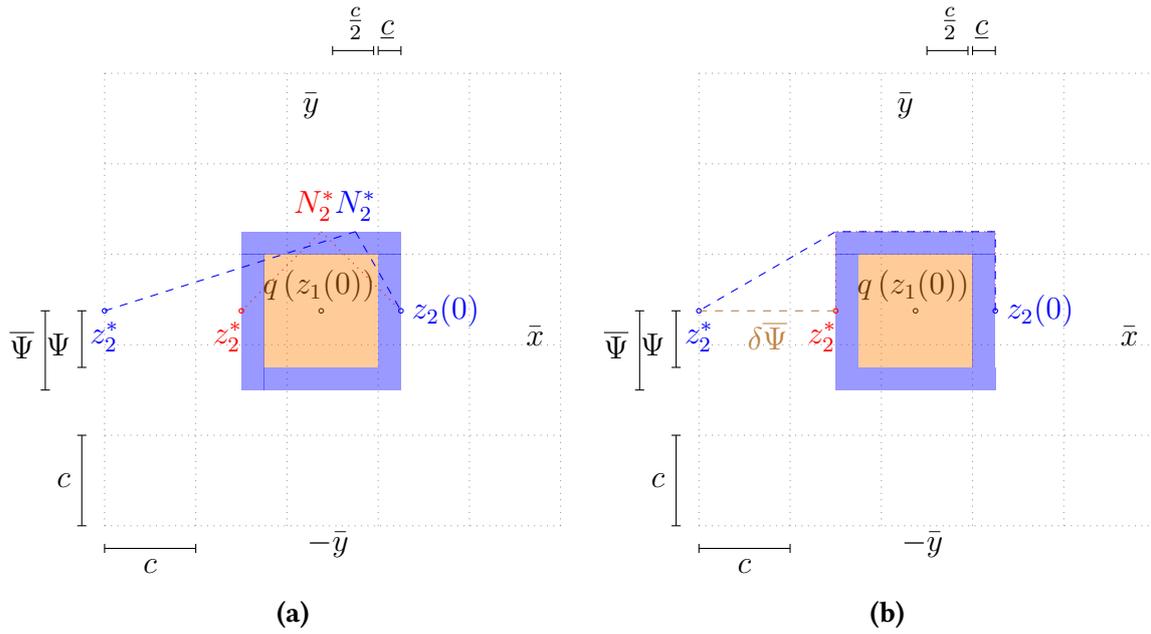
where  $d_z: \mathbb{X} \times \mathbb{X} \rightarrow \mathbb{R}_{\geq 0}$  and  $d_u: \mathbb{U} \times \mathbb{U} \rightarrow \mathbb{R}_{\geq 0}$  describes the metrics via  $\ell_p(z_p^*, 0) = 0$ ,  $\ell_p(z_p, u_p) > 0$  for  $z_p \in \mathbb{X}$  and  $u_p \in \mathbb{U}$ .  $0 \leq \lambda \leq 1$ ,  $\lambda \in \mathbb{R}_{\geq 0}$  the fraction of the imposed penalty of the control additionally to the cost function.

The usage of an occupancy grid allows to classify necessary feasible detours measured by stage cost function based on Definition 4.7, which have to be taken by robots to circumvent each other. As a robot occupies one cell, for a circumventing robot different detour possibilities can be classified in a finite way depending on the additional path length to take into account in comparison to the direct path, which is pictured in Fig. 20. The longest detour in ratio to the direct path is coloured in red, taking half of the circumference of the cell in relation to the diameter of the cell, while the blue indicates the smallest ratio taking the cathetus of the triangle instead of the hypotenuse. Hence, with this classification about the worst-positioning of the robots considering the ratio of direct path to detours we can state the following theorem. Please note that for a comprehensive view we included the derivation into this theorem:

**Theorem 4.8** (Worst-case positioning of robots)

Assume a set of robots with time discrete model (7) restricted by state (8) and control constraints (8) suited with feasible initial conditions and disjunct (intermediate) end points  $z_p(n), z_q(n), z_p(n+N), z_q(n+N) \in \mathbb{X}$  satisfying Assumption 4.3 with  $z_p(n+N) \neq z_q(n+N)$  and a minimum cell size  $\underline{c}$  obtained by (22). For two arbitrary chosen robots  $p, q \in [1 : P]$ ,  $q$  keeps its position





**Figure 21:** Illustration of the sufficient prediction horizon length  $N_2^*$  to recognise a decrease of costs by taking the detour (left) and feasible trajectories for robot 2 (right), with  $x_i(0)$  as initial position,  $q(\cdot)$  the quantised state and  $x_i^*$  the (intermediate) target for  $i \in [1 : 2]$ , from Sprodowski and Pannek (2020).

the worst-case ratio of a necessary detour in favour to its direct path can be determined, which is also enabled by the occupancy grid structure, as the classification of the detours is therefore finite. The illustration in Fig. 21b shows the different ratios and resulting necessary horizon length  $N_p^*$  for a robot  $p$  to recognise a decrease of costs despite the necessary detour and additional control effort, which has to be taken into account. As the target on the opposite site in Fig. 21b varies for  $z_2$  in terms of distance, a sufficient horizon length can be evaluated via using the revealing triangle between initial and target point ( $z_1^{0,*}$ ) and the feasible point at the top margin of the blocking robot  $z_1$ . This allows us to apply this theorem stating the ratios of worst-case positioning to the class of cost functionals based on Def. 4.7, which is defined as the sufficient prediction horizon length to recognise a decrease of costs:

**Theorem 4.9** (Sufficient prediction horizon length)

For two robots  $p, q$  suited with feasible initial conditions  $z_p^0 \equiv z_p(0)$ ,  $z_q^0 \equiv z_q(0)$  and disjunct (intermediate) targets  $z_p(K) \in \mathbb{X}$  with  $K \in \mathbb{N}_0$ ,  $K < \infty$  suppose that (9), (8) are satisfied, minimising a cost function defined by Def. 4.7 with Assumptions 4.1 and 4.3 to hold and a feasible trajectory between  $z_p^0$  and  $z_p(K)$  exists bounded by Theorem 4.8. If

$$N_p \geq N_p^* > 1 + \frac{\sum_{k=0}^{N_p^*-1} \ell_p(z_p(k; z_p^0), u_p^*(k))}{\ell_p(z_p(0; z_p^0), 0)} \quad (52)$$

holds, then robot  $p$  calculates a feasible control, which allows to decrease the costs according to Def. 4.7 and hence, is able to circumvent robot  $q$ .

Using this theorem, the sufficient horizon length was illustrated in two examples from Sprokowski and Pannek (2020), using the holonomic and non-holonomic setting, which was also utilised in the setup of Sprokowski et al. (2018a):

**Example 4.10** (Holonomic example)

Suppose two holonomic robots defined by

$$z^+ = f(z, u) = z + u \text{ with } z_p(0) = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, z_p^* = \begin{pmatrix} -4 \\ 0 \end{pmatrix} \text{ and } z_q(0) = \begin{pmatrix} -2 \\ 0 \end{pmatrix}$$

satisfying constraints (9) and (8) and control defined by  $u \in [-1, 1]^2$  with  $u = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}$ ,  $\|u\| \leq \sqrt{2}$  and stage costs

$$\ell_p(z_p, u_p) := \left\| \begin{pmatrix} (x_p - x_p^*)^2 \\ 5(y_p - y_p^*) \end{pmatrix} \right\|_2 + 0.2 \|u_p^2\|_2. \quad (53)$$

The open-loop costs are presented in Table 2, demonstrating the sufficient prediction horizon length to circumvent robot  $q$  with  $N = 7$ . The first two columns show the costs for the open-loop prediction including the control penalty with  $\lambda = 0.2$  and  $\lambda = 0$ , respectively. The last column represents the stage costs without imposing any control. For an insufficient horizon length, i. e.  $N = 6$ , the solution of the optimiser reveals  $u_p = 0$ , holding the position of the robot. As the optimiser would not find a detour for the shorter horizon length, but to be able to calculate the costs for taking the detour for  $N = 6$ , the optimal control sequence is calculated with the sufficient horizon length  $N = 7$  and stripped by the last value.

**Table 2:** Open-loop costs  $\sum_{k=0}^N \ell_p(z_p(k; z_p^0), u_p^*(k))$  with and without control impact  $\lambda$ .

$N$	$\lambda = 0.2$	$\lambda = 0$	$\sum_{k=0}^N \ell_p(z_p(k; z_p^0), 0)$
6	102.484	102.184	96.97
7	112.732	112.469	113.135

Hence, if the control is incorporated in the cost function, the prediction horizon has to be at least  $N \geq 7$  to recognise the decrease of costs by taking the detour. For an illustration see Fig. 22a.

**Example 4.11** (Non-holonomic example)

Based on Worthmann et al. (2016) an example of a non-holonomic robot defined in the 2D plane via

$$z^+ = f(z, u) = \begin{pmatrix} x \\ y \\ \theta \end{pmatrix} + \begin{pmatrix} \cos(\theta) \\ \sin(\theta) \\ 0 \end{pmatrix} \nu + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \omega,$$

$$z_p(0) = \begin{pmatrix} 0 \\ 0 \\ \frac{\pi}{4} \end{pmatrix}, z_p^* = \begin{pmatrix} -4 \\ 0 \\ 0 \end{pmatrix} \text{ and } z_q(0) = \begin{pmatrix} -2 \\ 0 \\ 0 \end{pmatrix},$$

where  $\theta$  represents the orientation of the robot. Moreover, we define the given stage costs with

$$\ell_p(z_p, u_p) := \left\| \begin{pmatrix} (x_p - x_p^*)^2 \\ 5(y_p - y_p^*) \\ (\theta_p - \theta_p^*)^2 \end{pmatrix} \right\|_2 + 0.2 \|u_p^2\|_2.$$

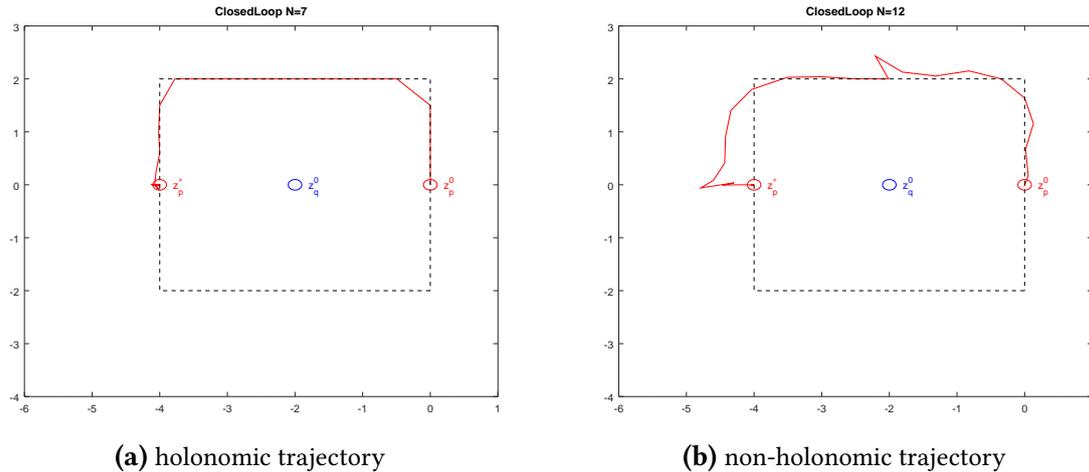
The control might be higher than for the holonomic model if the robot has to turn around due to the intricate non-holonomic constraints of the underlying non-linear model. The open-loop costs are given in Table 3 showing the costs with control impact for  $\lambda = 0.2$  and  $\lambda = 0$ . To compute the control effort for taking the detour for the shorter horizon, the optimal control sequence is computed with the longer horizon with the last value stripped as in the holonomic example.

**Table 3:** Open-loop costs  $\sum_{k=0}^N \ell_p(z_p(k; z_p^0), u_p^*(k))$  with and without control impact

$N$	$\lambda = 0.2$	$\lambda = 0$	$\sum_{k=0}^N \ell_p(z_p(k; z_p^0), 0)$
11	180.071	179.815	177.894
12	192.389	191.725	194.066

The smallest sufficient prediction horizon is  $N \geq 12$  to enable the optimiser to recognise lower costs considering the detour with the additional control impact. For an illustration of the chosen trajectory, see Fig. 22b. Note that with the non-holonomic setting the robot has to turn around, which leads to a non-straight trajectory line in comparison to the holonomic example before.

Both examples illustrate the necessary lower bound of the sufficient prediction horizon length to recognise a detour with the additional impact by penalising the control. The impact of the control values are too small to change the necessary horizon length but might have impact if a higher penalisation is chosen. A deeper and more detailed analysis about the design of the cost function and control effort analysis to stabilise such a non-holonomic robot is given in Worthmann et al. (2016) and for holonomic robots in Mehrez et al. (2020).



**Figure 22:** Trajectory of robot  $p$  in holonomic and non-holonomic with given initial conditions and targets  $z_p(0)$  and  $z_p^*$  and robot  $q$  with state  $z_q^0$  to be circumvented with the cell constraint illustrated by the dashed rectangle.

### 4.3 Contributions of the corresponding publications

The discussed publications contributed as follows: The first publication Sprodowski and Pannek (2015) considered a distributed scheme with a central controller handling the communication. The underlying dynamics was discrete and matching the chosen cell size. Stability for this particular scheme was shown and numerical results were performed. In Sprodowski et al. (2017), the prediction coherence was examined, which describes how similarities on the communication transmission occurs, which allow an estimate about savings for communication load. Then, for a special robotic scenario, i.e. non-holonomic robots, the quantisation scheme was applied and shown in numerical results and experiments to proof the concept in Mehrez et al. (2017). Then, to shift this approach of communication and quantisation to a broader class of control system, i.e. control-affine systems (LaValle, 2006), theoretical advances considering initial and recursive feasibility regarding DMPC properties and sufficient conditions were derived to ensure collision avoidance for the quantisation approach in Sprodowski et al. (2018a). Therein, also the differential update scheme was proposed to use the former defined prediction coherence. Then, to extend the scheme to lower the communication load further, in Sprodowski et al. (2018c) and Sprodowski and Pannek (2018), the approach of using interval communication was introduced in combination with relaxed collision avoidance constraints to attenuate the consumption of space. Further theoretical approaches were discussed in Sprodowski and Pannek (2020) examining the practical stability based on a traffic roundabout rule and sufficient conditions for the prediction horizon length to ensure that the system is able to achieve convergence. At last, a review of the presented publications and deeper literature review was given in Sprodowski (2021) adding a theoretical result on the lower bound of relaxation of collision avoidance constraints considering the interval communication approach.

# 5

## Priority rules

Until now, we considered a fixed sequential order of the subsystems to optimise and to apply their solution based on the scheme of Richards and How (2004a). Unfortunately, subsystems optimising first may block subsystems with a later order, if they try to assign a shared resource like a grid cell in a given occupancy grid. In that case, the latter subsystems have to wait until the occupied states are unblocked. Often, a fixed order might be inappropriate due to changes in the dynamics or position of the subsystems, which demands for another order or an adaptive model to change order between subsystems.

This motivates to establish dynamic priority rules and evaluate, how to apply them to such a distributed system and to reduce the overall costs quickly, i.e. that all subsystems are able to reach their target points. To calculate an order, a suitable criterion has to be found, which will be evaluated in the first section. Then, the priority rules are established to be used in the DMPC scheme in different ways: Either, in each time instant the order is calculated based on a sort mechanism using the chosen criterion or a hierarchical order is established based on the dependencies between subsystems, where for each dependency the criterion assigns the order of the depending subsystems. For example, dependencies could be defined as constraints to be imposed from other subsystems, which have to be obeyed by the current one. All schemes and numerical simulations were carried out in Sprodowski et al. (2020a) and are presented here in a comprehensive way.

### 5.1 Criteria for priority rules

To derive certain criteria for priority rules, we have to look for properties provided by the overall systems which are common for all subsystems. This ensures that we obtain a property which we can use to compare all subsystems. Suitable properties of systems are either system specific, e.g. energy consumptions, speed, or time-specific, e.g. waiting-time, duration time for reaching the target as discussed in Section 2.2.6. A suitable and more abstract criterion utilises the cost functional  $\ell_p(z_p, u_p)$  of each subsystem  $p$ , as this is related to the calculated trajectory of the subsystems in a general way and connected to the OCP of

---

**Algorithm 11** Initialisation phase for the DMPC algorithms using dynamic priority rules

---

**Require:** admissible states  $z_p$  for initial states  $z_p^0$  for all  $p \in [1 : P]$  and time instant  $n$

```

1: for all  $p = 1$  to  $P$  do
2:   if  $n = 0$  then
3:     for  $k = 1$  to  $N$  do
4:       Set  $\mathcal{I}_{p,0,k} := (k, q(z_p^0))$ 
5:       Set  $\Gamma_p := J_p^N(z_p^0, \bar{u}_p(z_p^0))$ 
6:     end for
7:   else
8:     Measure  $z_p(n)$ 
9:   end if
10:  Broadcast  $\mathcal{I}_{p,n}$  and criterion value  $\Gamma_p := J_p^N(z_p, u_p(z_p))$ 
11:  Receive  $i_{p,n}$  and  $\Gamma_q$  with  $q \in [1 : P] \setminus \{p\}$ 
12: end for

```

---

each subsystems. In a simple approach, the costs are obtained by each subsystems, sorted in ascending order and naturally this leads to a priority rule. Additionally, the costs can be distinguished into open-loop or closed-loop (stage costs), as the systems follows here the proposed DMPC scheme, which was also used in Sprodowski et al. (2020a). Here, such a criterion is stated as  $\Gamma := \{V_p^N\}$  utilising the open-loop costs. Then, a sorting function is defined as  $s: \mathbb{N}^P \times \mathbb{R}^P \rightarrow \mathbb{N}^P$

$$\Gamma_p := s(1, \dots, P, \Gamma_1, \dots, \Gamma_P), \quad (54)$$

which take values of the criterion and the indices of the robots and return an ordered sequence. Then, the initialisation for each variant of the chosen priority algorithm inherits a common initialisation phase, which is stated in Alg. 11. Here, in the initialisation phase in contrast to the previous DMPC schemes defined in Algs. 2, 5 and 6 the priority criteria is calculated in Alg. 11, line 4 based on the stop condition defined in Assumption 4.1 as the subsystems keep their initial feasible position in the first time instant. Additionally to the occupancy tuples  $\mathcal{I}_{p,n}$ , the priority criteria is broadcasted to all other subsystems (see line 10).

## 5.2 Priority schemes

In this section, we introduce different schemes spanning from a simple priority rule assignment to hierarchical methods, which consider the current dependencies, i.e. constraints imposed from the other subsystems to establish a dependency tree and hence, a execution order tree for all subsystems. Additionally, the latter allows to examine the system for concurrent execution, as subsystems without dependencies to others may carry out their prediction as long as no dependencies occur. Note that the criterion values may be calculated

decentralised as with each time instant, these values are transmitted additionally to the occupancy tuples. Hence, no further synchronisation is necessary among the subsystems. Furthermore, the scheme could also handle false values, which might lead to a suboptimal order, but still will reveal a feasible solution.

### 5.2.1 Simple priority assignment

In a first stage, we apply a simple priority rule, which sorts the criterion values instantaneously after each subsystems receives the occupancy grid and the criterion values without regarding former states or informations. The scheme is defined in Alg. 12. Here, we chose

---

**Algorithm 12** DMPC-scheme considering dynamic priority without memory.

---

```

1: for  $n = 0, 1, \dots$  do
2:   Call Alg. 11
3:   for  $p = 1$  to  $p = P$  do
4:      $m_p := s(1, \dots, P, \Gamma_1, \dots, \Gamma_P)$ 
5:   end for
6:   for  $p = m_p(1)$  to  $m_p(P)$  do
7:     Receive  $\mathbf{i}_{p,n}$  and  $\Gamma_q$  with  $q \in [1 : p - 1]$ 
8:     Solve OCP (15)
9:     Apply  $\mathbf{u}_p^*(0)$ 
10:    Broadcast  $\mathcal{I}_{p,n}$  and criterion value  $\Gamma_p := J_p^N(z_p, u_p(z_p))$ 
11:  end for
12: end for

```

---

to separate the initialisation phase from the execution scheme of the DMPC algorithm to illustrate the different plugged-in priority schemes. In line 4, the execution order is performed by each subsystem independently. Then, based on the obtained order the systems solve the OCP and apply their solutions (line 6 ff.) until a target condition is matched.

We want to emphasise that this scheme does not incorporate any deadlocks or cycles, which might hinder a system from reaching convergence in finite time. Such cycles may occur, even with the choice of an underlying priority criterion, if we refer to the example mentioned in (Grüne and Pannek, 2017, Example 9.35): Suppose a one-lane bridge, which should be crossed by two opposite placed vehicles and with the absence of any traffic rule. Assume that both vehicles are equipped with a priority criterion, which takes the maximum open-loop costs, each vehicle provides. Then, if one vehicle steps aside, and the other one will join the bridge, the priority criterion will favour the side-stepping vehicles due to the higher open-loop costs and force the vehicle on the bridge to step back. Hence, this will be repeated infinitely and show the motivation to establish a memory or dependency rule to attenuate the occurrence of cycles or deadlocks.

### 5.2.2 Priority scheme with memory function

We extend the simple priority assignment scheme with a memory rule to attenuate the occurrence of deadlocks or cycles. Note, that this does not verify the guaranteed absence of cycles, as this is still an open question for non-cooperative schemes without further modifications. The memory scheme is then defined in Alg. 13. The scheme evaluates at first,

---

**Algorithm 13** Memory function for priority sorting.

---

**Require:** priority order map  $m_p$ , time instant  $n$ ,  $\mathbf{i}_p$  and  $\Gamma_p$

```

1: Set  $\bar{m}_p := \emptyset$ 
2: for  $i = m_p(1)$  to  $m_p(P)$  do
3:   Set  $w := 0$ 
4:   for  $j = m_p(1)$  to  $m_p(P)$  do
5:     if  $\mathcal{I}_{i,n,k} = \mathcal{I}_{j,n,k}, k \in [1: N]$  then
6:       Set  $w := 1$ 
7:     end if
8:   end for
9:   if  $w = 0$  then
10:    Append  $\bar{m}_p \cup i$ 
11:   end if
12: end for
13:  $\bar{m}_p := s(\bar{m}_p(1), \dots, \bar{m}_p(|\bar{m}_p|), \Gamma_{\bar{m}_p(1)}, \dots, \Gamma_{\bar{m}_p(|\bar{m}_p|)})$ 
14: Set  $b := 1$ 
15: for  $j = \bar{m}_p(1), \dots, \bar{m}_p(|\bar{m}_p|)$  do
16:   while  $m_p(b) \notin \bar{m}_p$  do
17:     Set  $b := b + 1$ 
18:   end while
19:   Set  $m_p(b) := j$ 
20:   Set  $b := b + 1$ 
21: end for
22: return  $m_p$ 

```

---

if there are any dependencies of subsystems, i.e. they are addressing the same occupancy grid cells (see Alg. 13, line 5). If this condition is satisfied, the affected subsystems are added to a sort map  $m_p$  (line 10). Then, each subsystems is able to calculate a sort order in a decentralised manner and inserts its new calculated positions in the map (line 15) by swapping the affected positions. Then, the scheme can be plugged-in into the DMPC Alg. 12 by replacing line 4 with

**Set**  $m_p$  by calling Alg. 13.

As this scheme is simple to implement and does not impose an extensive analysis or hierarchy, the drawback comes with the binary distinction of dependent and independent subsystems: As the dependency is decided by  $w$  (Alg. 13, line 6), this reveals a set of subsystems to be sorted, which are not directly dependent by each other, but in a transitive relation, i.e. subsystem 1 depends on subsystem 2 and 2 depends on subsystem 3. Hence, not necessarily

subsystems 1 and 3 should be sorted as they might not be dependent on each other and therefore, hinders the overall system from a better execution order. Nevertheless, also cycles may occur, as the scheme does not prevent systems from entering and leaving the sort map periodically. Hence, to evaluate concurrent execution possibilities and a dependency hierarchy, the following subsection illustrates such a scheme.

### 5.2.3 Dependencies and concurrent execution

In this section, we introduce priority hierarchies, which establish an execution order. The hierarchy is defined as a level assigned for each robot  $\Pi_p \in \mathbb{N}, p \in [1: P]$ . The ascending numbers describes the priority, on which the subsystems can carry out their optimisation and apply their solution. For  $\Pi_p = 1$  that is the highest hierarchy level and hence, without any dependencies. The dependencies are defined such, that for  $\Pi_p < \Pi_q$  subsystem  $q$  has to obey the imposed constraints from subsystem  $p$ . In the initialisation phase, all values are set to  $\Pi_p = 1$ , for all  $p \in [1: P]$ . The evaluation of the dependencies and resulting hierarchy is defined in Alg. 14. Here, each subsystem  $p$  evaluates the occupancy grid  $\mathbf{i}_p$  for the same

---

**Algorithm 14** Dependency evaluation among systems via hierarchies

---

**Require:** Given system  $p$  with  $\Pi_p$

- 1: **for all**  $q \in [1: M]$  **with**  $\Pi_p = \Pi_q$  **do**
- 2:     **for**  $k = 0$  to  $N$  **do**
- 3:         **if**  $\mathcal{I}_{p,n,k} = \mathcal{I}_{q,n,k}$  **then**
- 4:              $m := s(p, q, \Gamma_p, \Gamma_q)$
- 5:             **if**  $m(1) = p$  **then**
- 6:                 **Set**  $\Pi_q := \Pi_q + 1$
- 7:             **else**
- 8:                 **Set**  $\Pi_p := \Pi_p + 1$
- 9:             **end if**
- 10:             **Call** recursively Alg. 14 for  $i := m(2)$  with  $\Gamma_{m(2)}$
- 11:         **end if**
- 12:     **end for**
- 13: **end for**
- 14: **return**  $\Pi_p$

---

hierarchy level (line 1) if a conflict exists for the current calculated prediction. If such a conflict is found (line 3), a priority is evaluated between the two affected subsystems  $p, q$ ,  $p \neq q$  and the level is increased for the system with the lower priority (line 7). Then, the subsystem with lower priority evaluates the next hierarchy level, if additionally conflicts exists on this level. The adopted DMPC scheme is defined in Alg. 15. After initialisation of the overall system with start values (Alg. 15, line 2), the priority values are evaluated decentralised based on the initial communicated occupancy tuples from Alg. 11. Then, the optimisation is carried out hierarchy-wise from  $i = 1$  to  $\max_i \{\Pi_i\}$ . After performing the optimisation, each subsystem recognises if the priority value has changed, which prevents the system from applying the conflicting solution and to enter the next hierarchy level anyway. If the hierarchy level was preserved ( $\Pi_p = i$ ), the solution can be applied. As

**Algorithm 15** DMPC-scheme for the overall system considering priority hierarchies

---

```

1: for  $p = 1$  to  $P$  do
2:   Set  $\Pi_p = 1$ 
3: end for
4: for  $n = 0, 1, \dots$  do
5:   Call Alg. 11
6:   for  $p = 1$  to  $P$  do
7:     Calculate priority variable  $\Pi_p$  by Alg. 14
8:   end for
9:   for  $i := 1$  to  $\max_i \{\Pi_i\}$  do
10:    for all  $p$ , with  $\Pi_p = i$  in parallel do
11:      Solve OCP (15)
12:      Broadcast  $\mathcal{I}_p(n), \Gamma_p$ 
13:      Calculate  $\Pi_p$  for system  $p$  by Alg. 14
14:      if  $\Pi_p = i$  then
15:        Apply  $u_p^*(0)$ 
16:      end if
17:    end for
18:  end for
19: end for
20: for  $p = 1$  to  $P$  do
21:   Call Alg. 16
22: end for

```

---

the subsystems normally starts with only a small number of conflicts in the beginning, if they are far away from each other, then enter many conflicts if they are in the centre of the operational space, the number of conflicts decreases if they are closer to their targets. Hence, at the end of each time instant Alg. 16 is called. In this algorithm, each subsystems with a higher hierarchy level ( $\Pi_p > 1$ ) has to evaluate if current dependencies still exist. From the definition above,  $\Pi_p = 1$  denotes that no dependencies exist. If at least one conflict is found (Alg. 16, line 6), a further examination is not necessary and the hierarchy level is kept. Otherwise, if no conflict exists, the subsystems is set to the higher hierarchy level and continues the procedure until either a conflict is found or the highest hierarchy level is reached (line 3).

Note that from a computational perspective a subsystem has to calculate  $\max_i \{\Pi_i\}$ -times in the worst-case if there a conflict is found for every level and a lower priority is assigned. From the communication perspective, this scheme adds finite additional communication load as at maximum the predictions have to be broadcasted  $\max \{\Pi\}$  times among all subsystems to ensure full information to establish a valid optimisation order in a decentralised manner.

---

**Algorithm 16** Check existing dependencies among systems via hierarchies
 

---

```

1: if  $\Pi_p > 1$  then
2:   Set  $w := 0$ 
3:   while  $w = 0$  and  $\Pi_p > 1$  do
4:     for all  $q \in [1: M]$  with  $\Pi_q = \Pi_p - 1$  do
5:       if  $\mathcal{I}_{p,n,k} = \mathcal{I}_{q,n,k}$ ,  $k \in [1: N]$  then
6:         Set  $w := 1$ 
7:       else
8:         Set  $\Pi_p = \Pi_p - 1$ 
9:       end if
10:    end for
11:  end while
12: end if

```

---

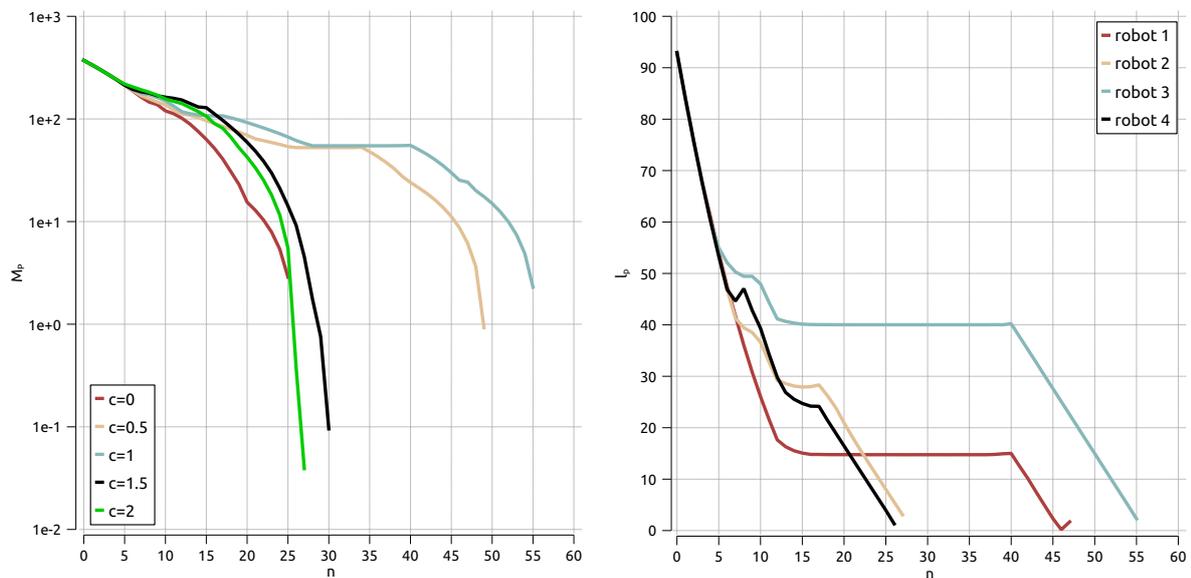
### 5.2.4 Impact of priority rules

In this subsection, we shortly present excerpts of the numerical results to illustrate the influence of priority rules on the convergence time of the overall system. For the reason of conciseness, we present only the variant based on the open-loop criterion  $\Gamma_V$  with cumulated closed-loop costs (33) and individual closed-loop costs for each robot to illustrate the behaviour of the robots considering the applied priority rule. The simulation setup follows the scenario used in Sprodownski et al. (2018a). To give the reader the possibility to compare the influence of the priority rules solely, the classic DMPC scheme without quantisation is additionally referenced in the results with  $c = 0.0$ .

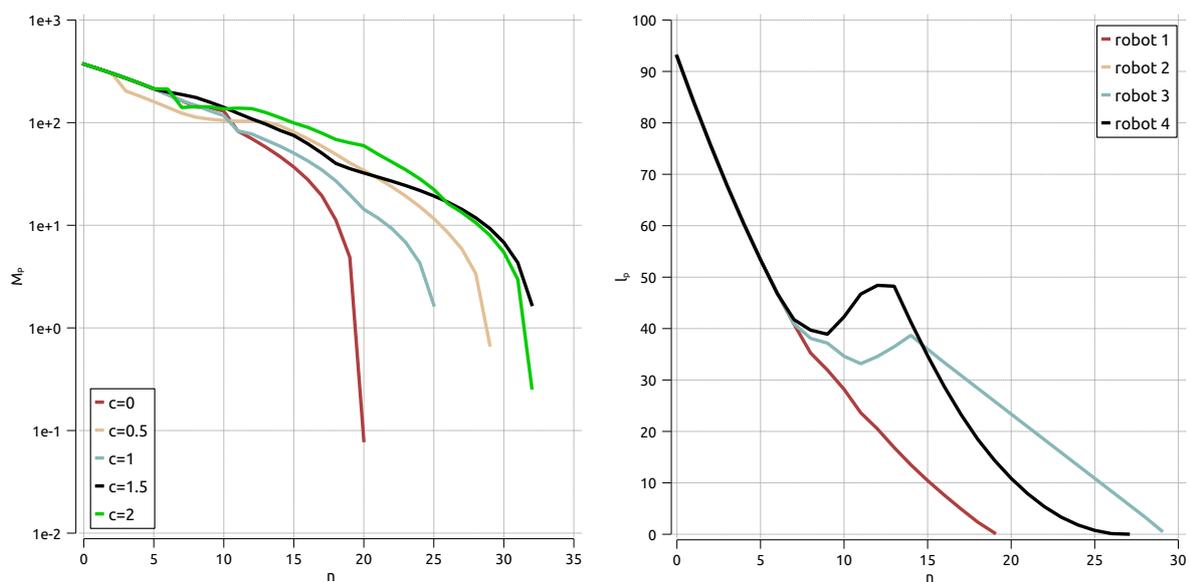
Here, in the first Fig. 23 the simple priority rule without a memory (Section 5.2.1) is presented. The results show the drawback of this simple rule as for a longer period a temporary cycle occurs, which leads to longer converge times especially for the middle-sized cell sizes  $c = \{0.5, 1.0\}$ . The costs depicted for each robot individually shows the effect with the monotonous costs of robot 2 and 3 (see Fig. 23, right). For the other cell sizes, the improvement over the fixed priority scheme can be shown with convergence times below  $n = 30$  in comparison to Fig. 12.

For the priority with a memory rule (Section 5.2.2), this scheme shows an overall better performance considering both, the classical DMPC scheme and the applied quantisation with larger cell sizes (see Fig. 24). Although the scheme might still be affected easily for deadlocks, the configuration here is stable enough to ensure a comparable good performance with respect to the convergence time. Furthermore, the individual costs of each robot (Fig. 24 right) show the tendency to elitism as robot 1 keeps the high priority level and all other robots have to take and consider the detour. Nevertheless, the overall convergence time is still shorter than with the fixed scheme.

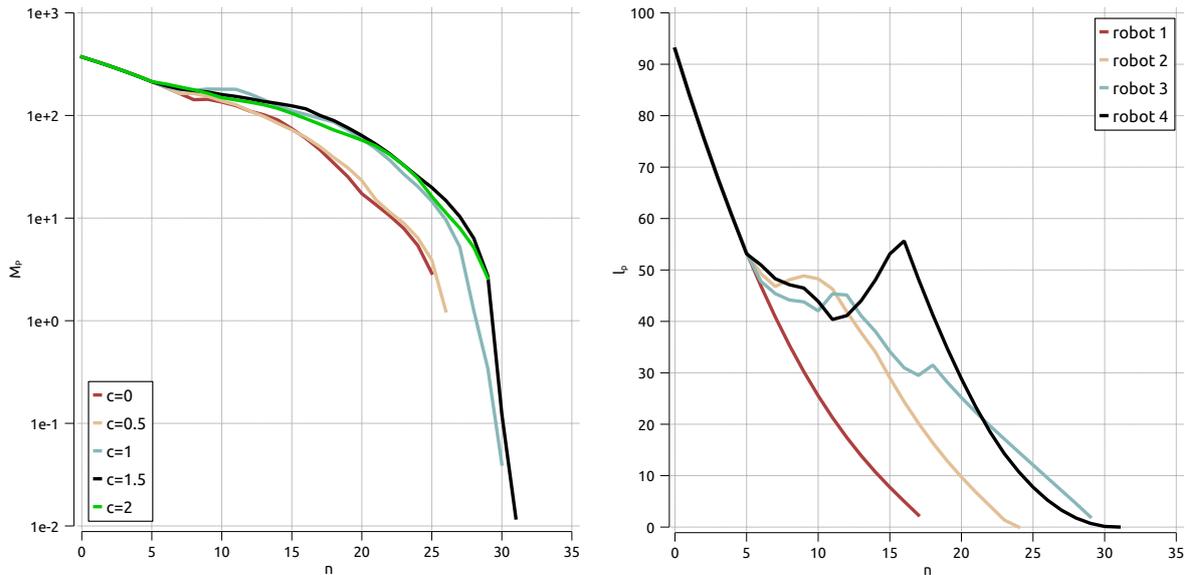
Considering the priority hierarchies (see Fig. 25), the scheme shows also shorter convergence times as in comparison to the fixed setting (see Fig. 12). But in comparison to the results of the simple priority rule with memory (see Fig. 24), the classical variant of the DMPC scheme without quantisation and middle-sized cell sizes  $c = \{0.5, 1.0\}$  are slightly larger. Nevertheless, the convergence times are close for all cell size showing that the scheme



**Figure 23:** Four holonomic systems applying the priority rule based on  $\Gamma_\ell$ : evolution of  $M_P$  and  $N = 9$  (left). Evolution of  $l_p$  for each robot with  $c = 1.5$  and  $N = 9$  (right), from Sprodowski et al. (2020a).



**Figure 24:** Four holonomic systems applying the priority rule based on  $\Gamma_V$  with memory: development of  $M_P$  with  $N = 9$  (left) and of  $l_p$  for each robot with  $c = 0.5$  and  $N = 9$  (right), from Sprodowski et al. (2020a).



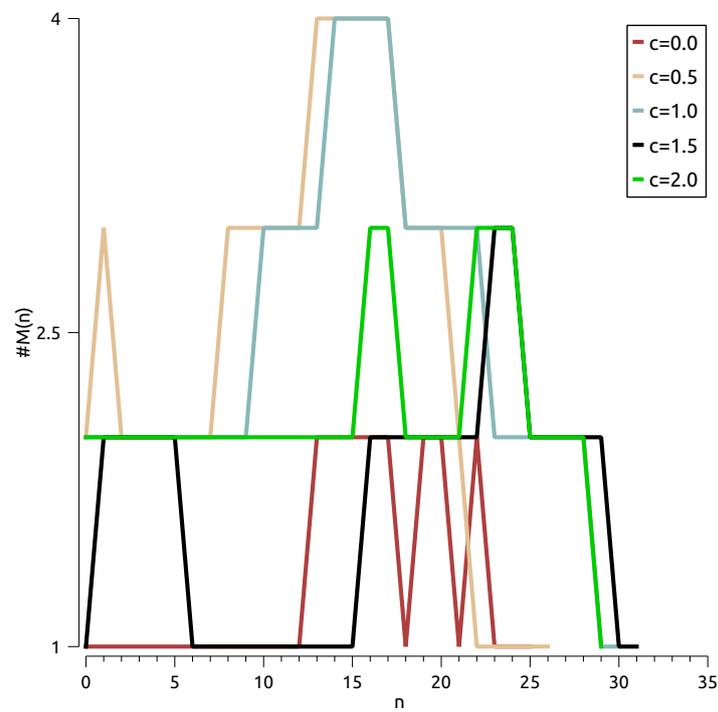
**Figure 25:** Four holonomic systems applying the priority rule based on the open-loop cost criterion  $T_V$  using a priority hierarchy: development of accumulated closed-loop costs  $M_P$  with horizon length  $N = 9$  (left) and of closed-loop costs  $\ell_p$  for each robot with cell size  $c = 1.5$  and  $N = 9$  (right), from Sprodowski et al. (2020a).

seems to be robust in terms of preventing deadlocks and ensuring stable convergence times. The individual costs for each robot (see Fig. 25 right) shows that robot 4 is taking even a longer detour compared to the simple rule with memory (see Fig. 24, right). This allows the conclusion that once a priority hierarchy is established, it is retained as long as the conflict exists.

To illustrate lastly the number of priority hierarchies and their time evolution for the application of the priority hierarchy rule, this is shown in Fig. 26. The diagram shows the time evolution of number of priority hierarchies  $\#M(n)$  for all cell sizes  $c$ . As the scenario considers at maximum 4 robots, the number of priority hierarchies is at maximum 4. For smaller cell sizes  $c = \{0.5, 1.0\}$  a maximum dependency between all robots occurs at time instants  $n = 12$  to  $19$  in contrast to the larger cell sizes, where the maximum number of level is at 3. With smaller cell sizes, the robots are closer and hence, it is more likely, a dependency between all robots in the centre of the operation space is established.

### 5.2.5 Contribution of the corresponding publication

Our contribution in Sprodowski et al. (2020a) presents an extension of the DMPC scheme in fixed order, which is used by Richards and How (2004a). Different priority schemes were defined and applied from simple priority rules to priority hierarchies with the allowance to execute in parallel. Extensive simulations were performed to validate the proposed schemes and show their better convergence time in comparison to the fixed-order scheme. Adapted initial and feasible proofs to validate this scheme were additionally discussed.



**Figure 26:** Four holonomic systems applying a priority rule based on the open-loop cost criterion  $I_V$  using a priority hierarchy: maximum priority queue length  $\#M(n)$  with horizon length  $N = 9$  for each time instant  $n$  for all cell sizes  $c$ , from Sprodowski et al. (2020a).

# 6

## Applications and outlook

In this chapter, we will give a short overview over the different application scenarios which were considered and conclude then with an outlook on further research perspectives. As the MPC and DMPC, respectively allows a broad application range from chemical reactors with slow dynamics covering the field of process control (Longhi et al., 2005), robotic applications or vehicle traffic scenarios covers the research field of necessary real-time control approaches with fast dynamics. In parallel, also macroscopic models could be utilised for this approach as considered for traffic flows Camponogara et al. (2009) or production scenarios or supply chains, which were considered in Zeng et al. (2009) and Jang et al. (2013), respectively.

### **6.1 Applications for quantised distributed model predictive control**

In our publications, we focused on the necessity of communication and their reductions. This covers mostly traffic scenarios, which demand for a steady link between subsystems due to safety reasons ensuring mainly collision avoidance constraints. In the first publications (Sprodowski and Pannek, 2015; Sprodowski et al., 2017) vehicle scenarios were considered. The proposed DMPC scheme (with quantisation) was evaluated in intersectional settings, where collisions may occur without active input of control. The focus on the communication part utilising the quantisation was then performed on robotic settings, mainly concerning holonomic and non-holonomic settings (Mehrez et al., 2017; Sprodowski et al., 2018a; Sprodowski and Pannek, 2018, 2020). Using also the robotic examples, priority rules were applied extensively for non-holonomic robotic scenarios in Sprodowski et al. (2020a) and briefly evaluated in correlation with prediction coherence in Sprodowski and Pannek (2017). Looking for production scenarios to apply the DMPC method, first results were presented in Sprodowski et al. (2018b), where this scheme was applied on a distributed machine scenario with a directed material flow, where the machines' material flow supply depend on their predecessors.

## 6.2 Conclusions & outlook

In this thesis, different methods considering the reduction of communication load was applied to a non-cooperative DMPC scheme mostly considering scenarios with fast dynamics, i.e. vehicle or robotic scenarios. We recap the research questions to conclude the presented results:

**Research question 1:** What is the necessary and sufficient communication overhead to guarantee stability and a certain degree of suboptimality in the distributed non-cooperative setting?

This issue was addressed in Section 4.1 with introducing the occupancy grid in Section 4.1.1, which allows transmitting integers instead of floating point values and hence, a quantisation of the communication exchange between the subsystems enables us to use differential updates explored in Section 4.1.3. The depicted results show the minimal number of communication tuples, which have to be exchanged. Further extensions as the interval communication were presented in Section 4.1.5 with two cells to be transmitted per time instance. However, as this reduction comes with the drawback of a higher space consumption, constraint relaxation attenuates the consumption of operational space.

**Research question 2:** Which effects does a relaxation from a fixed optimization order to dynamic priority orders have on stability and suboptimality?

Different priority rules were discussed in detail in Chapter 5, where an excerpt of the results was shown for the open-loop costs priority rule. This criterion was used in a simple priority rule without regarding any history states. Then, this rule was equipped with a memory to regard only the affected dependencies. To this end, a priority hierarchy utilising a deordering to allow for parallel execution was established. Overall, the latter two approaches show that a dynamic order, which is derived from the objective of the subsystems, leads to shorter convergence times with breaking up the fixed sequence of the subsystems. Further criteria might be considered as other available properties of the systems might also be appropriate.

**Research question 3:** How can convergence be achieved in a setting using a quantisation of states, i.e. here an occupancy grid?

At first, important properties are initial and recursive feasibility, which ensures that an initial plan (i.e. a feasible solution for each subsystem) can be calculated and applied. The recursive feasibility then ensures that the algorithm does not terminate unexpectedly and is able to calculate a feasible solution in every time instant. Furthermore, convergence that each subsystem achieves the assigned targets was shown in Section 4.2, where a assumption

based on a switching control law was made, which illustrates an alternative to imposing terminal constraints. The switching control law is applicable to robotic or traffic scenarios. However, the switching control law has to be modified if other scenarios are considered as production scenarios or applications in process control.

Considering limitations, convergence was shown in Section 4.1.4 based on the assumption that the subsystems can come to an immediate hold. In practice this is not possible in such a case that actuators are imposed with delays. Hence, for practical implementations a safety margin has to be added to ensure the stopping distance.

Further considerations are the applications to other examples with a complete different dynamics as manufacturing or production scenarios, where in many cases linear control mechanisms or heuristic scheduling rules were utilised in many publications. First results considering a distributed machine scenario were examined in Sprodowski et al. (2018b). A more extensive analysis on this approach is carried out in the accepted article in Sprodowski et al. (2020b), where different horizon lengths and objectives considering the overall operational target, were considered.

With our contributions, we open a broad field for future research considering a more detailed investigation of traffic scenarios with priority rules to see how transferable and ubiquitous these rules are applied. It is an open question, how and if different priority criteria should be incorporated in the cost function itself to obtain an optimal order. These priority schemes allows to adapt the DMPC scheme and also the underlying quantisation for the microscopic level, e.g. controlling the individual vehicle, up to the macroscopic level, e.g. managing traffic flow or routing planning. Also, future research will give insights for the evaluation for the quantisation scheme, which is suitable to be applied for e.g. in production scenarios to evaluate the impact on the overall behaviour of a manufacturing scenario. Furthermore, the quantisation scheme, elaborated in application perspectives, allows for establishing a grid for traffic management and developing a heat map to cluster high-used spaces. On the theoretical side, based on our theoretical fundamentals of the quantisation approach, conditions and rules should be elaborated and derived to choose an optimal cell size considering the convergence times. An open question to be explored is an optimal choice of the cell size according to a sufficient short simulation time and a low difference in the predictions. Concerning priority rules, conditions for showing properties as stability and convergence should be adapted for DMPC schemes, which incorporate priority rules. Extending these schemes with event-triggering methods, an open question will be, which impact this will impose on the communication load especially in combination with traffic scenarios, where degrees of freedoms are more restricted than in the robotic settings.

To this end, disturbance and external influences might be considered: Equipped with sensors, the prediction scheme could be incorporating non-participating entities as, e.g. in traffic scenarios pedestrians or cyclists, which might be included via an estimated prediction. A realistic, but from research point of view interesting scenario would be the consideration of electric vehicles in combination with the stabilisation of electric power grid, which was outlined in Sprodowski and Pannek (2016), which combines lastly the class of macroscopic and microscopic scenarios.



## Publicised work by the author

---

- T. Sprodowski and J. Pannek. Stability of distributed MPC in an intersection scenario. *Journal of Physics: Conference Series*, 659(1):12049, 2015. doi: 10.1088/1742-6596/659/1/012049. URL <http://stacks.iop.org/1742-6596/659/i=1/a=012049>  
*My share was 70%. I formulated partially the literature review, model and partially the proof for stability and have done model implementation, simulations and conclusions.*
- T. Sprodowski, J. K. Sagawa, and J. Pannek. Connection between Quantisation and Bandwidth Requirements of Distributed Model Predictive Control. *IFAC-PapersOnLine*, 50(1):10329–10334, 2017. ISSN 2405-8963. doi: 10.1016/j.ifacol.2017.08.1666. URL <http://www.sciencedirect.com/science/article/pii/S2405896317322723>  
*My share was 80%. I formulated the literature review, underlying model and partially the constraints. I carried out simulations and conclusions. The paper was presented by Jürgen Pannek at IFAC World Congress 2017 in Toulouse, France.*
- M. W. Mehrez, T. Sprodowski, K. Worthmann, G. K. I. Mann, R. G. Gosine, J. K. Sagawa, and J. Pannek. Occupancy Grid based Distributed Model Predictive Control of Mobile Robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4842–4847, Vancouver, Canada, 2017. \textcopyright IEEE. doi: 10.1109/IROS.2017.8206360. URL <https://doi.org/10.1109/IROS.2017.8206360>  
*My share was 30%. I partially formulated the literature review and underlying model, formulated the issue about skipping cells and designed the simulation setup. The work was presented by Mohamed W. Mehrez at the IROS Conference in Vancouver, Canada.*
- T. Sprodowski, M. W. Mehrez, K. Worthmann, G. K. Mann, R. G. Gosine, J. K. Sagawa, and J. Pannek. Differential communication with distributed MPC based on occupancy grid. *Information Sciences*, 453:426–441, 2018a. ISSN 00200255. doi: 10.1016/j.ins.2018.04.034. URL <https://doi.org/10.1016/j.ins.2018.04.034>

*My share was 50%. I have done the literature review, formulated partially the underlying model and formulated the conditions for the safety margins. I formulated also the differential update scheme. I carried out the simulations for the holonomic part and have written the interpretations for the simulations and conclusions.*

- T. Sprodowski, Y. Zha, and J. Pannek. Interval Superposition Arithmetic Inspired Communication for Distributed Model Predictive Control. In M. Freitag, H. Kotzab, and J. Pannek, editors, *Proceedings of the 6th International Conference on Dynamics in Logistics (LDIC 2018)*, pages 327–334, Bremen, Germany, 2018c. Springer International Publishing. doi: 10.1007/978-3-319-74225-0\_44. URL

[https://doi.org/10.1007/978-3-319-74225-0\\_{\\_}44](https://doi.org/10.1007/978-3-319-74225-0_{_}44)

*My share was 80%. I have done the literature review, formulated partially the model and carried out simulations, interpretations and conclusions. I presented the work at LDIC 2018 in Bremen.*

- T. Sprodowski and J. Pannek. Relaxed collision constraints based on interval superposition principle in a DMPC scheme. In *Proceedings of the 24th International Conference on Parallel and Distributed Systems*, pages 831–838, Singapore, Singapore, 2018. IEEE. doi: 10.1109/PADSW.2018.8644621. URL

<https://ieeexplore.ieee.org/document/8644621>

*My share was 90%. I have done literature review, formulated the underlying model and algorithm to establish the interval communication. I carried out simulations and draw conclusions. The work was presented by me at ICPADS 2018 in Singapore.*

- T. Sprodowski, M. W. Mehrez, and J. Pannek. Dynamic-Priority-based DMPC with an Occupancy Grid for Mobile Systems. *International Journal of Control*, pages 1–29, 2020a. doi: 10.1080/00207179.2019.1707291. URL

<https://doi.org/10.1080/00207179.2019.1707291>

*My share was 90%. I have done the literature review, formulated the priority schemes and carried out simulations and interpretations of the results.*

- T. Sprodowski and J. Pannek. Analytical Aspects of Distributed MPC based on an Occupancy Grid for Mobile Robots. *Applied Science*, 10(3), 2020. doi: 10.3390/app10031007. URL

<https://www.mdpi.com/2076-3417/10/3/1007>

*My share was 90%. I have done the literature review, proposed the underlying model, proposed the sufficient prediction horizon length conditions and for convergence.*

- T. Sprodowski. Collision avoidance for mobile robots based on an Occupancy Grid. In T. Faulwasser, M. A. Müller, and K. Worthmann, editors, *Recent Advances in Model Predictive Control - Theory, Algorithms and Applications*, chapter 9, pages 215–237. Springer Berlin Heidelberg, to appear, 2021. doi: 10.1007/978-3-030-63281-6. URL

<https://www.springer.com/gp/book/9783030632809>

*My share was 100%.*

- T. Sprodowski and J. Pannek. Impact of Quantisation on Consistency of DMPC in Street Traffic with Dynamic Priority Rules. *Proceedings in Applied Mathematics and Mechanics*, 17(1):819–820, 2017. doi: 10.1002/pamm.201710377. URL <https://doi.org/10.1002/pamm.201710377>  
*My share was 90%. I proposed the underlying model, carried out implementations and partially draw the conclusions.*
- T. Sprodowski, J. K. Sagawa, and J. Pannek. Frequency Based Model Predictive Control of a Manufacturing System. In *IFAC-PapersOnLine*, volume 51, pages 801–806, Vienna, Austria, 2018b. IFAC. doi: 10.1016/j.ifacol.2018.04.012. URL <https://www.sciencedirect.com/science/article/pii/S240589631830140X>  
*My share was 80%. I formulated the literature review, proposed the underlying model, carried out simulations and partially the interpretations.*
- T. Sprodowski, J. K. Sagawa, A. S. Maluf, M. Freitag, and J. Pannek. A Multi-Product Scenario utilizing Model Predictive Control. *Expert Systems with Applications*, 162: 113734, 2020b. doi: 10.1016/j.eswa.2020.113734. URL <https://doi.org/10.1016/j.eswa.2020.113734>  
*My share was 80%. I formulated the literature review, proposed the underlying model, carried out simulations and partially interpreted the results.*

# Bibliography

---

- W. Al-Gherwi, H. Budman, and A. Elkamel. A robust distributed model predictive control based on a dual-mode approach. *Computers and Chemical Engineering*, 50:130–138, 2013. ISSN 00981354. doi: 10.1016/j.compchemeng.2012.11.002. URL <https://dx.doi.org/10.1016/j.compchemeng.2012.11.002>.
- A. Alessio and A. Bemporad. Decentralized model predictive control of constrained linear systems. In *Proceedings of the European Control Conference*, pages 2813–2818, Kos, Greece, 2007. ISBN 9783952417386. doi: 10.23919/ECC.2007.7068856. URL <https://doi.org/10.23919/ECC.2007.7068856>.
- N. Altmüller and L. Grüne. Distributed and boundary model predictive control for the heat equation. *GAMM-Mitteilungen*, 35(2):131–145, 2012. doi: 10.1002/gamm.201210010. URL <http://onlinelibrary.wiley.com/doi/10.1002/gamm.201210010/references>.
- A. Anderson, A. H. González, A. Ferramosca, and E. Kofman. Finite-time convergence results in Model Predictive Control. In *Proceedings of the European Control Conference*, pages 3203–3208, Limassol, Cyprus, 2018. IEEE. ISBN 9783952426982. doi: 10.23919/ECC.2018.8550143. URL <https://dx.doi.org/10.23919/ECC.2018.8550143>.
- A. Balluchi, L. Benvenuti, S. Engell, T. Geyer, K. H. Johansson, J. Lygeros, M. Morari, G. Papafotiou, F. Santucci, and O. Stursberg. Hybrid Control of Networked Embedded Systems. *European Journal of Control*, 11(4-5):478–508, 2005. ISSN 0947-3580. doi: 10.1016/S0947-3580(05)71047-5. URL [https://dx.doi.org/10.1016/S0947-3580\(05\)71047-5](https://dx.doi.org/10.1016/S0947-3580(05)71047-5).
- M. Bennewitz, W. Burgard, and S. Thrun. Finding and optimizing solvable priority schemes for decoupled path planning techniques for teams of mobile robots. *Robotics and Autonomous Systems*, 41(2-3):89–99, 2002. ISSN 09218890. doi: 10.1016/S0921-8890(02)00256-7. URL [https://dx.doi.org/10.1016/S0921-8890\(02\)00256-7](https://dx.doi.org/10.1016/S0921-8890(02)00256-7).

- N. G. Bourbakis. A traffic priority language for collision-free navigation of autonomous mobile robots in dynamic environments. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 27(4):573–587, 1997. ISSN 10834419. doi: 10.1109/3477.604097. URL <https://dx.doi.org/10.1109/3477.604097>.
- S. J. Buckley. Fast motion planning for multiple moving robots. In *International Conference on Robotics and Automation*, pages 322–326, Scottsdale, AZ, USA, 1989. IEEE. ISBN 0818619384. doi: 10.1109/ROBOT.1989.100008. URL <https://doi.org/10.1109/ROBOT.1989.100008>.
- E. Camponogara and M. L. de Lima. Distributed optimization for MPC of linear networks with uncertain dynamics. *IEEE Trans. Autom. Control*, 57(3):804–809, 2012. ISSN 00189286. doi: 10.1109/TAC.2011.2168070. URL [http://ieeexplore.ieee.org/xpls/abs/\\_all.jsp?arnumber=6018991](http://ieeexplore.ieee.org/xpls/abs/_all.jsp?arnumber=6018991).
- E. Camponogara and L. Oliveira. Distributed optimization for model predictive control of linear-dynamic networks. *Systems, Man, and Cybernetics - Part A: Systems and Humans*, 39(6):1331–1338, 2009. ISSN 1540-1413. URL [http://ieeexplore.ieee.org/xpls/abs/\\_all.jsp?arnumber=5233809](http://ieeexplore.ieee.org/xpls/abs/_all.jsp?arnumber=5233809).
- E. Camponogara and H. F. Scherer. Distributed optimization for model predictive control of linear dynamic networks with control-input and output constraints. *Automation Science and Engineering*, 8(1):233–242, 2011. URL [http://ieeexplore.ieee.org/xpls/abs/\\_all.jsp?arnumber=5556047](http://ieeexplore.ieee.org/xpls/abs/_all.jsp?arnumber=5556047).
- E. Camponogara, D. Jia, B. Krogh, and S. Talukdar. Distributed model predictive control. *IEEE Control Systems Magazine*, 22(1):44–52, 2002. ISSN 02721708. doi: 10.1109/37.980246. URL <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=980246>.
- E. Camponogara, H. F. Scherer, and L. V. Moura. Distributed optimization for predictive control with input and state constraints: Preliminary theory and application to urban traffic control. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, pages 3726–3732, San Antonio, TX, USA, 2009. IEEE. ISBN 978-1-4244-2793-2. doi: 10.1109/ICSMC.2009.5346887. URL <https://doi.org/10.1109/ICSMC.2009.5346887>.
- B. Chen, G. Hu, W.-a. Zhang, and L. Yu. Distributed Mixed H<sub>2</sub>/H Fusion Estimation With Limited Communication Capacity. *IEEE Transactions on Automatic Control*, 61(3):805–810, 2016. doi: 10.1109/TAC.2015.2450271. URL <https://dx.doi.org/10.1109/TAC.2015.2450271>.
- H. Chen and F. Allgöwer. A Quasi-Infinite Horizon Nonlinear Model Predictive Control Scheme with Guaranteed Stability. *Automatica*, 34(10):1205–1217, 1998. ISSN 00051098. doi: 10.1016/S0005-1098(98)00073-9. URL [https://dx.doi.org/10.1016/S0005-1098\(98\)00073-9](https://dx.doi.org/10.1016/S0005-1098(98)00073-9).

- L. Chen and C. Englund. Cooperative Intersection Management: A Survey. *IEEE Transactions on Intelligent Transportation Systems*, 17(2):570–586, 2016. ISSN 15249050. doi: 10.1109/TITS.2015.2471812. URL <https://dx.doi.org/10.1109/TITS.2015.2471812>.
- P. S. Cisneros and H. Werner. A Dissipativity Formulation for Stability Analysis of Nonlinear and Parameter Dependent MPC. In *Proceedings of the American Control Conference*, pages 3894–3899, Wisconsin, MI, USA, 2018. AACC. ISBN 9781538654286. doi: 10.23919/ACC.2018.8431417. URL <https://dx.doi.org/10.23919/ACC.2018.8431417>.
- A. Colvin. CSMA with collision avoidance. *Computer Communications*, 6(5):227–235, 1983. ISSN 0140-3664. doi: [https://doi.org/10.1016/0140-3664\(83\)90084-1](https://doi.org/10.1016/0140-3664(83)90084-1). URL <http://www.sciencedirect.com/science/article/pii/0140366483900841>.
- B. D’Andréa-Novel, G. Campion, and G. Bastin. Control of nonholonomic wheeled mobile robots by state feedback linearization. *The International Journal of Robotics Research*, 14(6):543–559, 1995. ISSN 0278-3649. doi: 10.1177/027836499501400602. URL <https://doi.org/10.1177/027836499501400602>.
- Y. Deswarte, L. Blain, and J. C. Fabre. Intrusion tolerance in distributed computing systems. In *Proceedings of the IEEE Computer Society Symposium on Research in Security and Privacy*, pages 110–121, Oakland, CA, USA, 1991. IEEE. doi: 10.1109/RISP.1991.130780. URL <https://dx.doi.org/10.1109/RISP.1991.130780>.
- M. D. Doan, T. Keviczky, and B. De Schutter. An iterative scheme for distributed model predictive control using Fenchel’s duality. *Journal of Process Control*, 21(5):746–755, 2011. ISSN 0959-1524. doi: 10.1016/j.jprocont.2010.12.009. URL <https://dx.doi.org/10.1016/j.jprocont.2010.12.009>.
- A. Dua. Blockchain: A CEO’s Perspective On Distributed Ledger Technology!, 2018. URL <https://trak.in/tags/business/2018/03/12/blockchain-distributed-ledger-technology/>.
- W. B. Dunbar. Distributed receding horizon control of coupled nonlinear oscillators: Theory and application. In *Proceedings of the IEEE Conference on Decision and Control (CDC)*, pages 4854–4860, San Diego, CA, USA, 2006. ISBN 1424401712; 9781424401710. doi: 10.1109/Cdc.2006.376959. URL <http://www.scopus.com/inward/record.url?eid=2-s2.0-39649105214&partnerID=40&md5=f81333a5043f3eac821842bfa1bf07ab>.
- W. B. Dunbar. Distributed receding horizon control of dynamically coupled nonlinear systems. *IEEE Transactions on Automatic Control*, 52(7):1249–1263, 2007a. ISSN 0018-9286. doi: 10.1109/Tac.2007.900828. URL <https://dx.doi.org/10.1109/Tac.2007.900828>.

- W. B. Dunbar. Distributed receding horizon control of cost coupled systems. In *Proceedings of the 46th IEEE Conference on Decision and Control*, pages 2510–2515, New Orleans, LA, USA, 2007b. IEEE. doi: 10.1109/CDC.2007.4434033. URL <https://dx.doi.org/10.1109/CDC.2007.4434033>.
- W. B. Dunbar and D. S. Caveney. Distributed receding horizon control of vehicle platoons: Stability and string stability. *IEEE Transactions on Automatic Control*, 57(3):620–633, 2012. ISSN 00189286. doi: 10.1109/TAC.2011.2159651. URL <https://dx.doi.org/10.1109/TAC.2011.2159651>.
- M. El Mongi Ben Gaid and A. Çela. Trading Quantization Precision for Sampling Rates in Networked Systems with Limited Communication. In *Proceedings of the 45th IEEE Conference on Decision and Control*, pages 1135–1140, San Diego, CA, USA, 2006. IEEE. ISBN 1-4244-0171-2. doi: 10.1109/CDC.2006.377761. URL <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4177871>.
- A. Eqtami, D. V. Dimarogonas, and K. J. Kyriakopoulos. Novel event-triggered strategies for Model Predictive Controllers. In *Proceedings of the 50th IEEE Conference on Decision and Control and European Control Conference*, pages 3392–3397, Orlando, FL, USA, 2011. IEEE. ISBN 9781612848006. doi: 10.1109/CDC.2011.6161348. URL <https://dx.doi.org/10.1109/CDC.2011.6161348>.
- M. Erdmann and T. Lozano-Pérez. On multiple moving objects. *Algorithmica*, 2(1-4): 477–521, 1987. ISSN 01784617. doi: 10.1007/BF01840371. URL <https://dx.doi.org/10.1007/BF01840371>.
- R. Evers and S. Proost. Optimizing intersections. *Transportation Research Part B*, 71:100–119, 2015. ISSN 0191-2615. doi: 10.1016/j.trb.2014.10.006. URL <https://dx.doi.org/10.1016/j.trb.2014.10.006>.
- B. Fankhauser, L. Makarem, and D. Gillet. Collision-free intersection crossing of mobile robots using decentralized navigation functions on predefined paths. In *Proceedings of the IEEE 5th International Conference on Cybernetics and Intelligent Systems*, pages 392–397, Qingdao, China, 2011. IEEE. ISBN 9781612841984. doi: 10.1109/ICCIS.2011.6070361. URL <https://doi.org/10.1109/ICCIS.2011.6070361>.
- M. Farina, G. Betti, and R. Scattolini. Distributed predictive control of continuous-time systems. *Systems & Control Letters*, 74:32–40, 2014. ISSN 01676911. doi: 10.1016/j.sysconle.2014.10.001. URL <http://linkinghub.elsevier.com/retrieve/pii/S0167691114002084>.
- M. Farina, A. Perizzato, and R. Scattolini. Application of distributed predictive control to motion and coordination problems for unicycle autonomous robots. *Robotics and Autonomous Systems*, 72:248–260, 2015. ISSN 09218890. doi: 10.1016/j.robot.2015.06.007. URL <http://linkinghub.elsevier.com/retrieve/pii/S0921889015001359>.

- P. Giselsson. Adaptive nonlinear model predictive control with suboptimality and stability guarantees. In *Proceedings of the 49th IEEE Conference on Decision and Control*, pages 3644–3649, Atlanta, GA, USA, 2010. IEEE. ISBN 9781424477456. doi: 10.1109/CDC.2010.5717024. URL <https://dx.doi.org/10.1109/CDC.2010.5717024>.
- P. Giselsson and A. Rantzer. On feasibility, stability and performance in distributed model predictive control. *IEEE Transactions on Automatic Control*, 59(4):1031–1036, 2014. ISSN 00189286. doi: 10.1109/TAC.2013.2285779. URL <https://dx.doi.org/10.1109/TAC.2013.2285779>.
- G. C. Goodwin, H. Haimovich, D. E. Quevedo, and J. S. Welsh. A Moving Horizon Approach to Networked Control System Design. *IEEE Transactions on Automatic Control*, 49(9): 1427–1445, 2004. ISSN 1066033X. doi: 10.1109/MCS.2007.284513. URL <https://ieeexplore.ieee.org/document/1333197>.
- G. Grimm, M. Messina, S. Tuna, and A. Teel. Model predictive control: for want of a local control Lyapunov function, all is not lost. *IEEE Transactions on Automatic Control*, 50(5): 546–558, 2005. ISSN 0018-9286. doi: 10.1109/TAC.2005.847055. URL <https://dx.doi.org/10.1109/TAC.2005.847055>.
- S. Gros. A distributed algorithm for NMPC-based wind farm control. In *Proceedings of the 53rd IEEE Conference on Decision and Control*, pages 4844–4849, Los Angeles, CA, USA, 2014. IEEE. ISBN 978-1-4673-6090-6. doi: 10.1109/CDC.2014.7040145. URL <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7040145>.
- L. Grüne. NMPC without terminal constraints. *IFAC Proceedings Volumes (IFAC-PapersOnline)*, 45(17):1–13, 2012. ISSN 14746670. doi: 10.3182/20120823-5-NL-3013.00030. URL <https://doi.org/10.3182/20120823-5-NL-3013.00030>.
- L. Grüne and F. Müller. An algorithm for event-based optimal feedback control. In *Proceedings of the 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference*, pages 5311–5316, Shanghai, China, 2009. IEEE. ISBN 9781424438723. doi: 10.1109/CDC.2009.5399849. URL <https://dx.doi.org/10.1109/CDC.2009.5399849>.
- L. Grüne and J. Pannek. *Nonlinear Model Predictive Control: Theory and Algorithms*. Communications and Control Engineering. Springer, London, 2017. ISBN 978-3-319-46024-6. doi: 10.1007/978-3-319-46024-6. URL <http://link.springer.com/book/10.1007/978-3-319-46024-6>.

- L. Grüne and A. Rantzer. On the infinite horizon performance of receding horizon controllers. *IEEE Transactions on Automatic Control*, 53(9):2100–2111, 2008. ISSN 00189286. doi: 10.1109/TAC.2008.927799. URL <https://dx.doi.org/10.1109/TAC.2008.927799>.
- L. Grüne and K. Worthmann. A distributed NMPC scheme without stabilizing terminal constraints. In *Distributed Decision Making and Control*, pages 261–287. Springer, London, 2012. ISBN 978-1-4471-2265-4. doi: 10.1007/978-1-4471-2265-4\_12. URL [https://dx.doi.org/10.1007/978-1-4471-2265-4\\_{\\_}12](https://dx.doi.org/10.1007/978-1-4471-2265-4_{_}12).
- L. Grüne, J. Pannek, M. Seehafer, and K. Worthmann. Analysis of unconstrained nonlinear MPC schemes with varying control horizon. *SIAM Journal on Control and Optimization*, 48(8):4938–4962, 2010. doi: 10.1137/090758696. URL <https://doi.org/10.1137/090758696>.
- M. Guo and D. V. Dimarogonas. Consensus with quantized relative state measurement. *Automatica*, 49(8):2531–2537, 2013. ISSN 0005-1098. doi: 10.1016/j.automatica.2013.05.001. URL <http://dx.doi.org/10.1016/j.automatica.2013.05.001>.
- M. Heidarinejad, J. Liu, D. Muñoz de la Peña, J. F. Davis, and P. D. Christofides. Handling communication disruptions in distributed model predictive control. *Journal of Process Control*, 21(1):173–181, 2011a. ISSN 09591524. doi: 10.1016/j.jprocont.2010.11.005. URL <http://linkinghub.elsevier.com/retrieve/pii/S0959152410002234>.
- M. Heidarinejad, J. Liu, D. Muñoz De La Peña, J. F. Davis, and P. D. Christofides. Multirate Lyapunov-based distributed model predictive control of nonlinear uncertain systems. *Journal of Process Control*, 21(9):1231–1242, 2011b. ISSN 09591524. doi: 10.1016/j.jprocont.2011.07.016. URL <https://dx.doi.org/10.1016/j.jprocont.2011.07.016>.
- M. Höger and L. Grüne. On the Relation between Detectability and Strict Dissipativity for Nonlinear Discrete Time Systems. *IEEE Control Systems Letters*, 3(2):458–462, 2019. ISSN 24751456. doi: 10.1109/LCSYS.2019.2899241. URL <https://dx.doi.org/10.1109/LCSYS.2019.2899241>.
- H. Jang, T. Y. Jung, K. Yeh, and J. H. Lee. A model predictive control approach for fab-wide scheduling in semiconductor manufacturing facilities. In *IFAC Proceedings Volumes*, volume 46, pages 493–498, Fortaleza, Brazil, 2013. IFAC. ISBN 9783902823502. doi: 10.3182/20130911-3-BR-3021.00061. URL <http://www.sciencedirect.com/science/article/pii/S1474667016322364>.

- D. Jia and B. Krogh. Min-max feedback model predictive control for distributed control with communication. In *Proceedings of the American Control Conference*, pages 4507–4512, Anchorage, AK, USA, 2002. IEEE. ISBN 0-7803-7298-0. doi: 10.1109/ACC.2002.1025360. URL <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1025360>.
- D. Jia and B. H. Krogh. Distributed model predictive control. In *Proceedings of the American Control Conference*, pages 2767–2772, Arlington, VA, USA, 2001. IEEE. ISBN 0-7803-6495-3. doi: 10.1109/ACC.2001.946306. URL <https://ieeexplore.ieee.org/document/946306>.
- K. Kant and S. W. Zucker. Trajectory Planning : The Path-Velocity Decomposition. *The International Journal of Robotics Research*, 5(3):72–89, 1986. doi: 10.1177/027836498600500304. URL <https://doi.org/10.1177/027836498600500304>.
- M. A. Karabulut, A. F. M. Shahan Shah, and H. Ilhan. The Performance of the IEEE 802.11 DCF for Different Contention Window in VANETs. In *Proceedings of the 41st International Conference on Telecommunications and Signal Processing (TSP)*, pages 1–4, Athens, Greece, 2018. IEEE. doi: 10.1109/TSP.2018.8441481. URL <https://dx.doi.org/10.1109/TSP.2018.8441481>.
- A. Kashyap, T. Ba, and R. Srikant. Quantized consensus. *Automatica*, 43:1192–1203, 2007. doi: 10.1016/j.automatica.2007.01.002. URL <https://dx.doi.org/10.1016/j.automatica.2007.01.002>.
- S. S. Keerthi and E. G. Gilbert. Optimal Infinite Horizon Feedback Laws for a General Class of Constrained Discrete-Time Systems: Stability and Moving Horizon Approximations. *Journal of Optimization Theory and Applications*, 57:265–293, 1988. doi: 10.1007/BF00938540. URL <https://doi.org/10.1007/BF00938540>.
- K. A. Khaliq, A. Qayyum, and J. Pannek. Synergies of Advanced Technologies and Role of VANET in Logistics and Transportation. *International Journal of Advanced Computer Science and Applications(IJACSA)*, 7(11):359–369, 2016. doi: 10.14569/IJACSA.2016.071148. URL <https://dx.doi.org/10.14569/IJACSA.2016.071148>.
- K. A. Khaliq, A. Qayyum, and J. Pannek. Performance Analysis of Proposed Congestion Avoiding Protocol for IEEE 802.11s. *International Journal of Advanced Computer Science and Applications*, 8(2):356–369, 2017. ISSN 21565570. doi: 10.14569/IJACSA.2017.080246. URL <https://dx.doi.org/10.14569/IJACSA.2017.080246>.
- S. M. LaValle. *Planning Algorithms*. Cambridge University Press, Cambridge, U.K., 2006. ISBN 9780521862059. URL <http://planning.cs.uiuc.edu/>.

- H. Li, S. Liu, Y. C. Soh, and L. Xie. Event-Triggered Communication and Data Rate Constraint for Distributed Optimization of Multiagent Systems. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 48(11):1908–1919, 2018. doi: 10.1109/TSMC.2017.2694323. URL <https://dx.doi.org/10.1109/TSMC.2017.2694323>.
- J. Liu, D. Muñoz de la Peña, B. J. Ohran, P. D. Christofides, and J. F. Davis. A two-tier architecture for networked process control. *Chemical Engineering Science*, 63(22): 5394–5409, 2008. ISSN 00092509. doi: 10.1016/j.ces.2008.07.030. URL <https://dx.doi.org/10.1016/j.ces.2008.07.030>.
- S. Liu, D. Sun, and C. Zhu. A dynamic priority based path planning for cooperation of multiple mobile robots in formation forming. *Robotics and Computer-Integrated Manufacturing*, 30(6):589–596, 2014. ISSN 07365845. doi: 10.1016/j.rcim.2014.04.002. URL <http://dx.doi.org/10.1016/j.rcim.2014.04.002>.
- S. Liu, L. Xie, and D. E. Quevedo. Event-triggered Quantized Communication Based Distributed Convex Optimization. *IEEE Transactions on Control of Network Systems*, 5870 (99):1–11, 2016. ISSN 2325-5870. doi: 10.1109/TCNS.2016.2585305. URL <https://dx.doi.org/10.1109/TCNS.2016.2585305>.
- H. Lödding, K.-W. Yu, and H.-P. Wiendahl. Decentralized WIP-oriented manufacturing control Decentralized WIP-oriented manufacturing control ( DEWIP ). *Production Planning & Control*, 14(1):42–54, 2003. doi: 10.1080/0953728021000078701. URL <https://doi.org/10.1080/0953728021000078701>.
- S. Longhi, R. Trillini, and M. Vaccarini. Application of a Networked Decentralized MPC To Syngas Process in Oil Industry. In *IFAC-PapersOnLine*, volume 38, pages 431–436, Prague, Czech Republic, 2005. IFAC. doi: 10.3182/20050703-6-CZ-1902.01646. URL <http://linkinghub.elsevier.com/retrieve/pii/S1474667016376583>.
- S. Lucia, M. Kögel, and R. Findeisen. Contract-based Predictive Control of Distributed Systems with Plug and Play Capabilities. *IFAC-PapersOnLine*, 48(23):205–211, 2015. ISSN 24058963. doi: 10.1016/j.ifacol.2015.11.284. URL <http://dx.doi.org/10.1016/j.ifacol.2015.11.284>.
- W. Luo, N. Chakraborty, and K. Sycara. Distributed dynamic priority assignment and motion planning for multiple mobile robots with kinodynamic constraints. In *Proceedings of the American Control Conference*, pages 148–154, Boston, MA, USA, 2016. IEEE. ISBN 9781467386821. doi: 10.1109/ACC.2016.7524907. URL <https://doi.org/10.1109/ACC.2016.7524907>.
- J. M. Maestre and R. R. Negenborn. *Distributed Model Predictive Control Made Easy*. Springer Dordrecht, Dordrecht, intelligent edition, 2014. ISBN 978-94-007-7006-5. doi: 10.1007/978-94-007-7006-5. URL <http://link.springer.com/978-94-007-7006-5>.

- J. M. Maestre, P. Giselsson, and A. Rantzer. Distributed receding horizon Kalman filter. In *Proceedings of the 49th IEEE Conference on Decision and Control*, pages 5068–5073, Atlanta, Georgia, USA, 2010. IEEE. ISBN 0191-2216 VO -. doi: 10.1109/CDC.2010.5717370. URL <https://dx.doi.org/10.1109/CDC.2010.5717370>.
- J. M. Maestre, D. Muñoz de la Peña, E. F. Camacho, and T. Alamo. Distributed model predictive control based on agent negotiation. *Journal of Process Control*, 21(5):685–697, 2011. ISSN 09591524. doi: 10.1016/j.jprocont.2010.12.006. URL <http://linkinghub.elsevier.com/retrieve/pii/S0959152410002404>.
- D. Mayne and P. Falugi. Generalized Stabilizing Conditions for Model Predictive Control. *Journal of Optimization Theory and Applications*, 169(3):719–734, 2016. ISSN 15732878. doi: 10.1007/s10957-015-0838-1. URL <https://dx.doi.org/10.1007/s10957-015-0838-1>.
- D. Mayne, J. Rawlings, C. Rao, and P. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000. ISSN 00051098. doi: 10.1016/S0005-1098(99)00214-9. URL [https://dx.doi.org/10.1016/S0005-1098\(99\)00214-9](https://dx.doi.org/10.1016/S0005-1098(99)00214-9).
- M. W. Mehrez, T. Sprodowski, K. Worthmann, G. K. I. Mann, R. G. Gosine, J. K. Sagawa, and J. Pannek. Occupancy Grid based Distributed Model Predictive Control of Mobile Robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4842–4847, Vancouver, Canada, 2017. \textcopyright IEEE. doi: 10.1109/IROS.2017.8206360. URL <https://doi.org/10.1109/IROS.2017.8206360>.
- M. W. Mehrez, K. Worthmann, J. P. V. Cenerini, M. Osman, W. W. Melek, and S. Jeon. Model Predictive Control without terminal constraints or costs for holonomic mobile robots. *Robotics and Autonomous Systems*, 127:103468, 2020. ISSN 0921-8890. doi: <https://doi.org/10.1016/j.robot.2020.103468>. URL <http://www.sciencedirect.com/science/article/pii/S0921889019306232>.
- M. Mercangöz and F. J. Doyle. Distributed model predictive control of an experimental four-tank system. *Journal of Process Control*, 17(3):297–308, 2007. ISSN 09591524. doi: 10.1016/j.jprocont.2006.11.003. URL <http://linkinghub.elsevier.com/retrieve/pii/S0959152406001387>.
- H. Michalska and D. Q. Mayne. Robust Receding Horizon Control of Constrained Nonlinear Systems. *IEEE Transactions on Automatic Control*, 38(1):1623–1633, 1993. doi: 10.1109/9.262032. URL <https://doi.org/10.1109/9.262032>.
- T. Monteiro, D. Roy, and D. Ancaix. Multi-site coordination using a multi-agent system. *Computers in Industry*, 58(4):367–377, 2007. ISSN 0166-3615. doi: 10.1016/j.compind.2006.07.005. URL <http://www.sciencedirect.com/science/article/pii/S0166361506001114>.

- M. A. Müller, M. Reble, and F. Allgöwer. Cooperative control of dynamically decoupled systems via distributed model predictive control. *International Journal of Robust and Nonlinear Control*, 22(12):1376–1397, 2012. ISSN 1049-8923. doi: 10.1002/rnc.2826. URL <https://dx.doi.org/10.1002/rnc.2826>.
- G. N. Nair, M. Huang, and R. J. Evans. Optimal infinite horizon control under a low data rate. In *14th IFAC Symposium on System Identification*, pages 1115–1120, Newcastle, Australia, 2006. IFAC. doi: 10.3182/20060329-3-AU-2901.00179. URL <https://dx.doi.org/10.3182/20060329-3-AU-2901.00179>.
- G. D. Nicolao, L. Magni, and R. Scattolini. Stabilizing Receding-Horizon Control of Nonlinear Time-Varying Systems. *IEEE Trans. on Automatic Control*, 43(7):1030–1036, 1998. doi: 10.1109/9.701133. URL <https://doi.org/10.1109/9.701133>.
- J. Pannek. Parallelizing a state exchange strategy for noncooperative distributed NMPC. *Systems and Control Letters*, 62(1):29–36, 2013. ISSN 01676911. doi: 10.1016/j.sysconle.2012.10.015. URL <https://dx.doi.org/10.1016/j.sysconle.2012.10.015>.
- J. Pannek and K. Worthmann. Stability and performance guarantees for model predictive control algorithms without terminal constraints. *ZAMM Zeitschrift für Angewandte Mathematik und Mechanik*, 94(4):317–330, 2014. ISSN 15214001. doi: 10.1002/zamm.201100133. URL <https://dx.doi.org/10.1002/zamm.201100133>.
- J. A. Primbs and V. Nevistić. Feasibility and stability of constrained finite receding horizon control. *Automatica*, 36(7):965–971, 2000. ISSN 00051098. doi: 10.1016/S0005-1098(00)00004-2. URL [https://dx.doi.org/10.1016/S0005-1098\(00\)00004-2](https://dx.doi.org/10.1016/S0005-1098(00)00004-2).
- Y. Pu, M. N. Zeilinger, and C. N. Jones. Quantization design for distributed optimization with time-varying parameters. In *Proceedings of the 54th IEEE Conference on Decision and Control*, pages 2037–2042, Osaka, Japan, 2015. IEEE. ISBN 9781479986866. doi: 10.1109/CDC.2015.7402506. URL <https://dx.doi.org/10.1109/CDC.2015.7402506>.
- S. J. Qin and T. A. Badgwell. A survey of industrial model predictive control technology. *Control Engineering Practice*, 11(7):733–764, 2003. ISSN 09670661. doi: 10.1016/S0967-0661(02)00186-7. URL [https://dx.doi.org/10.1016/S0967-0661\(02\)00186-7](https://dx.doi.org/10.1016/S0967-0661(02)00186-7).
- D. E. Quevedo, P. K. Mishra, R. Findeisen, and D. Chatterjee. A Stochastic Model Predictive Controller for Systems with Unreliable Communications. *IFAC-PapersOnLine*, 48(23): 57–64, 2015. ISSN 24058963. doi: 10.1016/j.ifacol.2015.11.262. URL <https://dx.doi.org/10.1016/j.ifacol.2015.11.262>.

- A. Rantzer. Dynamic dual decomposition for distributed control. *Proceedings of the American Control Conference*, pages 884–888, 2009. ISSN 07431619. doi: 10.1109/ACC.2009.5160224. URL <https://dx.doi.org/10.1109/ACC.2009.5160224>.
- J. B. Rawlings and B. T. Stewart. Coordinating multiple optimization-based controllers: New opportunities and challenges. *Journal of Process Control*, 18(9):839–845, 2008. ISSN 09591524. doi: 10.1016/j.jprocont.2008.06.005. URL <https://dx.doi.org/10.1016/j.jprocont.2008.06.005>.
- J. B. Rawlings, D. Q. Mayne, and M. Diehl. *Model Predictive Control: Theory and Design*. Nob Hill Publishing, 2nd edition, 2017. ISBN 978-0975937730. URL <http://www.nobhillpublishing.com/mpc/index-mpc.html>.
- A. Richards and J. How. Decentralized model predictive control of cooperating UAVs. In *Proceedings of the 43th IEEE Conference on Decision and Control*, volume 4, pages 4286–4291, Nassau, Bahamas, 2004a. IEEE. ISBN 0780386825. doi: 10.1109/CDC.2004.1429425. URL <https://doi.org/10.1109/CDC.2004.1429425>.
- A. Richards and J. How. A decentralized algorithm for robust constrained model predictive control. In *Proceedings of the American Control Conference*, pages 4261–4266, Boston, MA, USA, 2004b. IEEE. ISBN 0-7803-8335-4. doi: 10.23919/ACC.2004.1383977. URL <https://ieeexplore.ieee.org/document/1383977/>.
- A. Richards and J. P. How. Model predictive control of vehicle maneuvers with guaranteed completion time and robust feasibility. In *Proceedings of the American Control Conference*, pages 4034–40, Denver, CO, USA, 2003. IEEE. ISBN 0-7803-7896-2. doi: 10.1109/ACC.2003.1240467. URL <https://dx.doi.org/10.1109/ACC.2003.1240467>.
- Y. Rochefort, H. Piet-Lahanier, S. Bertrand, D. Beauvois, and D. Dumur. Model predictive control of cooperative vehicles using systematic search approach. *Control Engineering Practice*, 32:204–217, 2014. ISSN 09670661. doi: 10.1016/j.conengprac.2014.01.003. URL <https://dx.doi.org/10.1016/j.conengprac.2014.01.003>.
- C. Savorgnan, C. Romani, A. Kozma, and M. Diehl. Multiple shooting for distributed systems with applications in hydro electricity production. *Journal of Process Control*, 21(5):738–745, 2011. ISSN 09591524. doi: 10.1016/j.jprocont.2011.01.011. URL <https://dx.doi.org/10.1016/j.jprocont.2011.01.011>.
- R. Scattolini. Architectures for distributed and hierarchical Model Predictive Control – A review. *Journal of Process Control*, 19(5):723–731, 2009. ISSN 0959-1524. doi: 10.1016/j.jprocont.2009.02.003. URL <https://doi.org/10.1016/j.jprocont.2009.02.003>.

- H. Scheu, J. Busch, and W. Marquardt. Nonlinear distributed dynamic optimization based on first order sensitivities. In *Proceedings of the American Control Conference*, pages 1574–1579, Baltimore, MD, USA, 2010. IEEE. ISBN 9781424474271. doi: 10.1109/ACC.2010.5531587. URL <https://doi.org/10.1109/ACC.2010.5531587>.
- P. O. M. Scokaert, D. Q. Mayne, and J. B. Rawlings. Suboptimal Model Predictive Control (Feasibility Implies Stability). *IEEE Trans. on Automatic Control*, 44(3):648–654, 1999. doi: 10.1109/9.751369. URL <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp={&}arnumber=751369>.
- T. Sprodowski. Collision avoidance for mobile robots based on an Occupancy Grid. In T. Faulwasser, M. A. Müller, and K. Worthmann, editors, *Recent Advances in Model Predictive Control - Theory, Algorithms and Applications*, chapter 9, pages 215–237. Springer Berlin Heidelberg, to appear, 2021. doi: 10.1007/978-3-030-63281-6. URL <https://www.springer.com/gp/book/9783030632809>.
- T. Sprodowski and J. Pannek. Stability of distributed MPC in an intersection scenario. *Journal of Physics: Conference Series*, 659(1):12049, 2015. doi: 10.1088/1742-6596/659/1/012049. URL <http://stacks.iop.org/1742-6596/659/i=1/a=012049>.
- T. Sprodowski and J. Pannek. Harmonisierung von Elektromobilität und verteilten Energienetzen. *Industrie Management*, 6:6–11, 2016. URL <http://www.industrie-management.de/node/22>.
- T. Sprodowski and J. Pannek. Impact of Quantisation on Consistency of DMPC in Street Traffic with Dynamic Priority Rules. *Proceedings in Applied Mathematics and Mechanics*, 17(1):819–820, 2017. doi: 10.1002/pamm.201710377. URL <https://doi.org/10.1002/pamm.201710377>.
- T. Sprodowski and J. Pannek. Relaxed collision constraints based on interval superposition principle in a DMPC scheme. In *Proceedings of the 24th International Conference on Parallel and Distributed Systems*, pages 831–838, Singapore, Singapor, 2018. IEEE. doi: 10.1109/PADSW.2018.8644621. URL <https://ieeexplore.ieee.org/document/8644621>.
- T. Sprodowski and J. Pannek. Analytical Aspects of Distributed MPC based on an Occupancy Grid for Mobile Robots. *Applied Science*, 10(3), 2020. doi: 10.3390/app10031007. URL <https://www.mdpi.com/2076-3417/10/3/1007>.
- T. Sprodowski, J. K. Sagawa, and J. Pannek. Connection between Quantisation and Bandwidth Requirements of Distributed Model Predictive Control. *IFAC-PapersOnLine*, 50(1):10329–10334, 2017. ISSN 2405-8963. doi: 10.1016/j.ifacol.2017.08.1666. URL <http://www.sciencedirect.com/science/article/pii/S2405896317322723>.

- T. Sprodowski, M. W. Mehrez, K. Worthmann, G. K. Mann, R. G. Gosine, J. K. Sagawa, and J. Pannek. Differential communication with distributed MPC based on occupancy grid. *Information Sciences*, 453:426–441, 2018a. ISSN 00200255. doi: 10.1016/j.ins.2018.04.034. URL <https://doi.org/10.1016/j.ins.2018.04.034>.
- T. Sprodowski, J. K. Sagawa, and J. Pannek. Frequency Based Model Predictive Control of a Manufacturing System. In *IFAC-PapersOnLine*, volume 51, pages 801–806, Vienna, Austria, 2018b. IFAC. doi: 10.1016/j.ifacol.2018.04.012. URL <https://www.sciencedirect.com/science/article/pii/S240589631830140X>.
- T. Sprodowski, Y. Zha, and J. Pannek. Interval Superposition Arithmetic Inspired Communication for Distributed Model Predictive Control. In M. Freitag, H. Kotzab, and J. Pannek, editors, *Proceedings of the 6th International Conference on Dynamics in Logistics (LDIC 2018)*, pages 327–334, Bremen, Germany, 2018c. Springer International Publishing. doi: 10.1007/978-3-319-74225-0\_44. URL [https://doi.org/10.1007/978-3-319-74225-0\\_{\\_}44](https://doi.org/10.1007/978-3-319-74225-0_{_}44).
- T. Sprodowski, M. W. Mehrez, and J. Pannek. Dynamic-Priority-based DMPC with an Occupancy Grid for Mobile Systems. *International Journal of Control*, pages 1–29, 2020a. doi: 10.1080/00207179.2019.1707291. URL <https://doi.org/10.1080/00207179.2019.1707291>.
- T. Sprodowski, J. K. Sagawa, A. S. Maluf, M. Freitag, and J. Pannek. A Multi-Product Scenario utilizing Model Predictive Control. *Expert Systems with Applications*, 162:113734, 2020b. doi: 10.1016/j.eswa.2020.113734. URL <https://doi.org/10.1016/j.eswa.2020.113734>.
- B. T. Stewart, A. N. Venkat, J. B. Rawlings, S. J. Wright, and G. Pannocchia. Cooperative distributed model predictive control. *Systems & Control Letters*, 59(8):460–469, 2010. ISSN 01676911. doi: 10.1016/j.sysconle.2010.06.005. URL <http://linkinghub.elsevier.com/retrieve/pii/S0167691110000691>.
- B. T. Stewart, S. J. Wright, and J. B. Rawlings. Cooperative distributed model predictive control for nonlinear systems. *Journal of Process Control*, 21(5):698–704, 2011. ISSN 09591524. doi: 10.1016/j.jprocont.2010.11.004. URL <http://linkinghub.elsevier.com/retrieve/pii/S0959152410002222>.
- S. Talukdar, D. Jia, P. Hines, and B. H. Krogh. Distributed model predictive control for the mitigation of cascading failures. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 4440–4445, Seville, Spain, 2005. IEEE. ISBN 0780395689. doi: 10.1109/CDC.2005.1582861. URL [http://ieeexplore.ieee.org/xpls/abs\\_{\\_}all.jsp?arnumber=1582861](http://ieeexplore.ieee.org/xpls/abs_{_}all.jsp?arnumber=1582861).

- P. Trodden and A. Richards. Distributed model predictive control of linear systems with persistent disturbances. *International Journal of Control*, 83(8):1653–1663, 2010. ISSN 0020-7179. doi: 10.1080/00207179.2010.485280. URL <https://dx.doi.org/10.1080/00207179.2010.485280>.
- R. J. Troutbeck and S. Kako. Limited priority merge at unsignalized intersections. *Transportation Research Part A*, 33(3-4):291–304, 1999. doi: 10.1016/S0965-8564(98)00046-9. URL [https://doi.org/10.1016/S0965-8564\(98\)00046-9](https://doi.org/10.1016/S0965-8564(98)00046-9).
- M. van Steen and A. S. Tanenbaum. *Distributed Systems*. CreateSpace Independent Publishing Platform, 3rd edition, 2017. ISBN 978-1543057386. URL <https://www.distributed-systems.net/index.php/books/ds3/>.
- P. Varutti, B. Kern, T. Faulwasser, and R. Findeisen. Event-based model predictive control for Networked Control Systems. In *Proceedings of the 48th IEEE Conference on Decision and Control*, pages 567–572, Shanghai, China, 2009. ISBN 978-1-4244-3871-6. doi: 10.1109/CDC.2009.5400921. URL <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5400921>.
- P. Varutti, T. Faulwasser, B. Kern, M. Kögel, and R. Findeisen. Event-based reduced-attention predictive control for nonlinear uncertain systems. In *Proceedings of the IEEE International Symposium on Computer-Aided Control System Design, part of Multi-Conference on Systems and Control*, pages 1085–1090, Yokohama, Japan, 2010. IEEE. ISBN 978-1-4244-5354-2. doi: 10.1109/CACSD.2010.5612775. URL <https://dx.doi.org/10.1109/CACSD.2010.5612775>.
- A. Venkat, J. Rawlings, and S. Wright. Stability and optimality of distributed model predictive control. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 6680–6685, Seville, Spain, 2005. IEEE. ISBN 0-7803-9567-0. doi: 10.1109/CDC.2005.1583235. URL <http://dx.doi.org/10.1109/CDC.2005.1583235>.
- A. Von Eichhorn, M. Werling, P. Zahn, and D. Schramm. Maneuver prediction at intersections using cost-to-go gradients. In *Proceedings of the IEEE International Conference on Intelligent Transportation Systems*, pages 112–117, The Hague, Netherlands, 2013. IEEE. ISBN 9781479929146. doi: 10.1109/ITSC.2013.6728219. URL <https://dx.doi.org/10.1109/ITSC.2013.6728219>.
- Q. Wang, Y. Zou, and Y. Niu. Event-triggered Model Predictive Control for Wireless Networked Control Systems with Packet Losses. In *5th Annual IEEE International Conference on Cyber Technology in Automation, Control and Intelligent Systems*, pages 1281–1286, Shenyang, China, 2015. ISBN 9781479987306. doi: 10.1109/CYBER.2015.7288128. URL <https://dx.doi.org/10.1109/CYBER.2015.7288128>.

- W. W. Wierwille, R. J. Hanowski, J. M. Hankey, C. A. Kieliszewski, S. E. Lee, A. Medina, and T. A. Dingus. Identification of driver errors : overview and recommendations. URL <https://rosap.ntl.bts.gov/view/dot/922>.
- K. Worthmann, M. W. Mehrez, M. Zanon, G. K. I. Mann, R. G. Gosine, and M. Diehl. Regulation of Differential Drive Robots using Continuous Time MPC without Stabilizing Constraints or Costs. *IFAC-PapersOnLine*, 48(23):129–135, 2015. doi: 10.1016/j.ifacol.2015.11.272. URL <http://dx.doi.org/10.1016/j.ifacol.2015.11.272>.
- K. Worthmann, M. W. Mehrez, M. Zanon, G. K. I. Mann, R. G. Gosine, and M. Diehl. Model Predictive Control of Nonholonomic Mobile Robots Without Stabilizing Constraints and Costs. *IEEE Transactions on Control Systems Technology*, 24(4):1394–1406, 2016. ISSN 1063-6536. doi: 10.1109/TCST.2015.2488589. URL <http://dx.doi.org/10.1109/TCST.2015.2488589>.
- N. Wu. A universal procedure for capacity determination at unsignalized (priority-controlled) intersections. *Transportation Research Part B*, 35:593–623, 2001. doi: 10.1016/S0191-2615(00)00012-6. URL [https://doi.org/10.1016/S0191-2615\(00\)00012-6](https://doi.org/10.1016/S0191-2615(00)00012-6).
- P. Yi and Y. Hong. Quantized Sub-gradient Algorithm and Data Rate Analysis for Distributed Optimization. *IEEE Transactions on Control of Network Systems*, 1(4):380–392, 2014. ISSN 2325-5870. doi: 10.1109/TCNS.2014.2357513. URL <https://dx.doi.org/10.1109/TCNS.2014.2357513>.
- Y. Yongjie. Collision Avoidance Planning in Multi-robot based on Improved Artificial Potential Field and Rules. In *IEEE International Conference on Robotics and Biomimetics*, pages 1026–1031, Bangkok, Thailand, 2009. IEEE. ISBN 9781424426799. doi: 10.1109/ROBIO.2009.4913141. URL <https://doi.org/10.1109/ROBIO.2009.4913141>.
- B. Zeng, L. Yao, and Y. He. An Optimal Scheduling Model for Robust Management of Supply Chain. In *International Conference on E-Business and Information System Security*, pages 1–5, Wuhan, China, 2009. IEEE. ISBN 978-1-4244-2909-7. doi: 10.1109/EBISS.2009.5137927. URL <https://doi.org/10.1109/EBISS.2009.5137927>.
- Y. Zha and B. Houska. Interval Superposition Arithmetic. (October), 2016. URL <http://arxiv.org/abs/1610.05862>.
- S. Zhu, M. Hong, and B. Chen. Quantized consensus ADMM for multi-agent distributed optimization. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4134–4138, Shanghai, China, 2016. ISBN 9781479999880. doi: 10.1109/ICASSP.2016.7472455. URL <https://dx.doi.org/10.1109/ICASSP.2016.7472455>.

# List of Tables

---

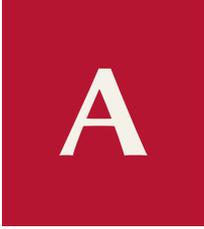
1	State of the art: Categorisation of coordination in DMPC . . . . .	18
2	Sufficient prediction horizon length: lower boundary for holonomic robots	60
3	Sufficient prediction horizon length: lower boundary for non-holonomic robots . . . . .	61



# List of Figures

---

1	Motivation: centralised and distributed architecture . . . . .	1
2	State of the art: illustration of an MPC cycle for two successive time instants	7
3	Classification of control schemes . . . . .	12
4	Classification of communication schemes . . . . .	18
5	Classification of the quantisation approaches . . . . .	20
6	Classification of the stability approaches . . . . .	21
7	Reduction of Communication: Example of a grid . . . . .	34
8	Example for unnoticed swapping cells of two robots . . . . .	36
9	Illustration of the restricted intersection scenario and robotic scenario . . .	39
10	Quantised prediction coherence for the intersection scenario comparing cell sizes . . . . .	40
11	Quantised prediction coherence for the robotic scenario comparing cell sizes	40
12	Robotic scenario: Average number of communicated tuples with diff. communication and full communication and convergence . . . . .	43
13	Interval Communication: Illustration of a bounding box . . . . .	45
14	Robotic scenario: Comparison of differential and interval communication .	46
15	Robotic scenario: Trajectories while performing interval communication .	47
16	Interval Communication: Leading Signs illustration . . . . .	48
17	Interval Communication: Reconstruction of the bounding box . . . . .	50
18	Robotic scenario: Relaxed constraints with interval communication . . . .	52
19	Convergence: Illustration of the circular control law for robots . . . . .	54
20	Sufficient prediction horizon length: Classification of detours . . . . .	58
21	Sufficient prediction horizon length: Recognition of decrease of costs . . .	59
22	Sufficient prediction horizon length: Examples for (non-)holonomic robots	62
23	Priority rules: Simple priority rule . . . . .	70
24	Priority rules: Simple priority rule with memory . . . . .	70
25	Priority rules: priority hierarchies . . . . .	71
26	Priority rules: Priority hierarchies size . . . . .	72



A

Attached publications by the author

## Stability of distributed MPC in an intersection scenario

T Sprodowski<sup>1</sup> and J Pannek<sup>1,2</sup>

<sup>1</sup> Department of Production Engineering, University of Bremen

<sup>2</sup> BIBA Bremer Institut für Produktion und Logistik GmbH, 28359 Bremen, Germany

E-mail: {spr<sup>1</sup>, pan<sup>2</sup>}@biba.uni-bremen.de

**Abstract.** The research topic of autonomous cars and the communication among them has attained much attention in the last years and is developing quickly. Among others, this research area spans fields such as image recognition, mathematical control theory, communication networks, and sensor fusion. We consider an intersection scenario where we divide the shared road space in different cells. These cells form a grid. The cars are modelled as an autonomous multi-agent system based on the Distributed Model Predictive Control algorithm (DMPC). We prove that the overall system reaches stability using Optimal Control for each multi-agent and demonstrate that by numerical results.

Today the development of autonomous cars and the insurance of their correct behavior are still a great challenge in both academic research and the automotive industry. To ensure that networked cars are reliable in shared-space scenarios, the car-to-car communication implemented as a cooperative strategy is a possibility for getting a solution in the intersection resource problem. The ideal situation will be if all participants support them. Yet, non-communicating participants should also be included within such scenarios. Here, we assume that even if communication data is missing, a networked car is able to estimate the future route of a respective participant sufficiently accurate to guarantee collision avoidance.

To incorporate the behavior of the other cars and of disturbance factors, we propose to utilize a Distributed Nonlinear Model Predictive Control Scheme to formulate our problem as an optimal control problem. Nonlinear Model Predictive control (NMPC) is nowadays a widespread method, which is used in many feedback control system applications [1]. A survey of the nonlinear version is given by [2] and [3]. NMPC itself is a three step procedure: First, the current state of the system is estimated. Then, based on that estimate, an internal model of the system dynamics is used to predict the state trajectory over a finite time horizon and to compute a control  $u$  minimizing a given cost functional subject to possible constraints. Last, only the initial segment of the computed control  $u$  is applied, which renders the algorithm to be iteratively applicable and repeat the procedure until the equilibrium is achieved. The goal of NMPC is to (practically) asymptotically stabilize the system to a target  $x^*$ . To guarantee this property, endpoint constraints [4] or terminal constraints and costs [5] as well as a relaxed Lyapunov inequality [3] have been proposed. Since terminal constraints are inadequate for a networked car scenario due to the size of the problem and also limit the feasible set [6], we follow the latter approach.

For our networked car scenario, a distributed control setting appears naturally. Stemming



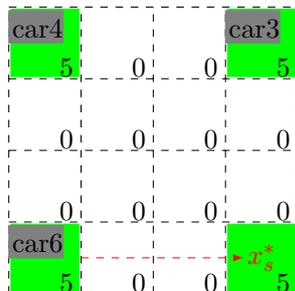
from a centralized idea, Richards and How [7], among others, proposed to split up the centralized optimal control problem into subproblems connected by a communication network, which lead to a Distributed Model Predictive Controller (DMPC). The general idea is to reduce the complexity of the problem induced by the number of constraints and control variables, which are needed by the centralized problem. While the total number of constraints and control variables remains unchanged, each subproblem is much smaller and can be solved more easily. We model these subproblems as an independent NMPC problems, and connect them to obtain all information about the chosen solution of the other subproblems. Here, we follow the approach to a fixed order, in which subproblems are solved.

The paper is organized as follows: In the following section, we describe the intersection scenario, which we consider throughout our work. In Section 2, we formally introduce the DMPC setting for our problem scenario. Respective stability conditions are shown to hold in Section 3 and illustrated using numerical experiments in Section 4. Last, we draw some conclusions in Section 5 and give a short outlook on future work.

### 1. Intersection Scenario

We consider a scenario based on an intersection without traffic lights similar to [8]. In this scenario, the authors proposed to divide the shared space of an intersection into multiple cells. These cells form a two-dimensional grid, which is administrated centrally. This scenario illustrates a simple form of a shared resource allocation problem. The cars are managed by one multi-agent each, which aims to cross the intersection in two directions straight ahead. The resulting problem is solved by a centralized approach utilizing the Organic Computing Principle [9]. An observer/controller-architecture is introduced to calculate a path through the intersection for each incoming networked car and to detect deviations using the  $A^*$  Search Algorithm [10]. The path is formulated as a sequence of tuples. Each tuple contains time ( $n$ ) and space coordinates ( $k, m$ ) to clarify when the cells will be occupied ( $p = (n_1, k_1, m_1), \dots, (n_i, k_i, m_i)$ ). Here,  $n$  represents the  $n$ th discrete time step,  $k$  and  $m$  are the coordinates within the grid. Since the cars are modelled as multi-agents which can re-plan their original path, deviations are possible. The observer recognizes the latter and influences the affected cars including the causer by renewing the paths and spreading the information. The communication is centralized and controlled by the observer-/controller-system. Hence, only car-to-infrastructure communication is considered.

We propose to modify this scenario and implement a distributed control. Different from [11] where the paths are predefined and the optimization function considers speed and acceleration only, our aim is to model both the car and the scenario more generally. The collision avoidance in [11] is introduced as a constraint, which is imposed to a car following the priority assignment. We consider the reservation of grid cells to avoid any collisions. An illustration of our scenario is illustrated in Figure 1. To this end, we allow traffic from each direction within our extended



**Figure 1.** Example of a  $4 \times 4$  - Intersection scenario divided in grid cells which shows the requested reservations made by cars, dashed line illustrates the planned path for car 6 to reach its target  $x_s^*$

scenario. This raises the risk of a deadlock, which, for instance, may occur if two cars stand opposite to each other and cannot give way. Here, we avoid this possibility by introducing lanes, which are a necessity for conventional cars during the introduction phase of networked cars. The lanes determine the direction in which a car can move forward. Additionally, we introduce one multi-agent per car, which performs the path calculation via a DMPC algorithm to compute an optimal collision-free solution through the grid. To follow the approach of a fixed order, each car is set up with an index by initialization. The simulation procedure is executed as described in Algorithm 1.

---

**Algorithm 1** Simulation procedure
 

---

Given: Initial time and set of active cars  $S$ .

While  $S \neq \emptyset$  do

(i) **State estimation:**

Get current state for each car  $x_s \in S$ .

(ii) **Control computation:**

For each car  $x_s \in S$  do

(a) If current position is target position, then remove car  $x_s$  from  $S$  and break.

Else set completed state to false.

(b) While completed state is false

1. Compute optimal path over finite horizon for given constraints.

2. Try to reserve time/space tuples. If successful, set completed state to true. Else, increment time value and try again.

(iii) **Control implementation:**

Implement all first control elements and shift horizon forward in time.

---

We model the target distance as a cost function and let the path be calculated and optimized by each vehicle separately. The actual path optimization in every time step  $n$  is necessary as we have multiple lanes for each direction so the dynamics is in the path calculation. We consider the reservation of grid cells to avoid any collisions.

## 2. Formulation in DMPC

### 2.1. System definition

We suppose the cars to be modeled as multi-agents and denote the overall state of the system by  $x$ , where there can be  $S$  subsystems

$$x = \{x_1, \dots, x_S\}.$$

where  $s$  denotes the  $s$ th car. From the physical point of view  $x$  describes the position of the car in the intersection scenario. The path  $p$  through the intersection scenario is given as a sequence of tuples  $(n, k, m) \in \mathbb{N}_0 \times \{1, \dots, L_k\} \times \{1, \dots, L_m\}$  with  $L_k, L_m \in \mathbb{N}$ , where the latter set denotes the time-space grid. The term  $(n, k, m)$  denotes the coordinates, at which a car can be situated, i.e.

$$x_s(n) = (n, k, m) \in \mathbb{N}_0 \times \{1, \dots, L_k\} \times \{1, \dots, L_m\}$$

where  $n$  denotes the discrete time, and  $k$  and  $m$  are the coordinates within the grid. The dynamics of the  $s$ th subsystem

$$x_s(n+1) = f_s(x_s(n), u_s(n)) \quad (1)$$

connects these tuples to generate a path, where the control  $u_s(n)$  implements the path item for the next time step. Hence, the path can be written as

$$\text{Path } p_s = (x_s(n))_{n \in \mathbb{N}_0}$$

The target for the overall system is denoted by  $x^*$  containing the targets

$$x^* = (x_s^*)_{s=\{1, \dots, S\}}$$

for each subsystem, which we consider to be time invariant for simplicity of exposition. The control  $u_s(n)$  of a subsystem  $s$  is given also as absolute coordinates of the grid similar to  $x_s(n)$  which contains also the path tuple  $(n, k, m)$  for each system change.

### 2.2. Constraints of Subsystems

Each subsystem is contemplated with constraints that have to be satisfied. Here, we introduce three different kinds of constraints, the first to incorporate restrictions solely based on the local state and control, the second to include collision avoidance, and the third to enforce stability of each subsystem. The latter is modeled via

$$\#x_s = |((n_1, k_1, m_1), \dots, (n_i, k_i, m_i))| < M, \quad (2)$$

where  $M$  defines an upper bound on the length of a path sequence. As a consequence, each subsystem has to reach its target in at most  $M$  steps.

To avoid collisions, we do not allow two subsystems  $x_r$  and  $x_v$ ,  $r \neq v$ , to take identical time-space coordinates. Hence, a tuple  $(n_i, k_i, m_i)$  can only appear at most once in all paths  $p_s$ . More formally, we obtain

$$\forall (n_i, k_i, m_i) \in x_r \Rightarrow (n_i, k_i, m_i) \notin x_v \quad (3)$$

for all  $r, v \in \{1, \dots, S\}$  with  $r \neq v$ .

Last, using local restrictions we restrict the motion of each car by either staying at the current position, i.e.  $x_s(n+1) = x_s(n)$ , or moving to a neighbouring cell, that is from  $x_s(n) = (n, k_1, m_1)$  to  $x_s(n+1) = (n+1, k_2, m_2)$ , where

$$(|k_2 - k_1|, |m_2 - m_1|) \leq (1, 1) \quad (4)$$

### 2.3. Costs for each Subsystem

Incorporating the dynamics (1) and the constraints (2), (3) and (4), each multi-agent optimizes its subsystem with respect to the stage cost

$$\ell(x_s(k), u_s(k)) = \|x_s(k) - x_s^*\|^2 + \lambda \|u_s(k)\|^2,$$

where  $\lambda > 0$  is a regularization factor to avoid bang-bang behavior of optimal solutions. Utilizing the DMPC approach, we define the cost function for each subsystem via

$$J_s^N(x_s^0, u_s) := \sum_{k=0}^{N-1} \ell_s(x_s(k), u_s(k)) \quad (5)$$

where  $x_s^0$  denotes the initial value of the path to be optimized, i.e.  $x_s(0) := x_s^0$ , and  $J_s^N$  the cumulated costs for the each system state  $k$  over an horizon of length  $N$ .

#### 2.4. Costs of Overall System

The cost function can also be interpreted as energy, i.e. the overall system loose potential energy if the cars are moving towards the target. Here, we assume that after each time step of the DMPC procedure, the overall system loses potential energy. While appearing restrictive, this assumption always holds due to our fixed order of computations for subsystems. Hence, at least one car  $x_s$  can move forward and reduces its own costs, and thereby also the cumulative system costs. The DMPC procedure implements a minimizer of (5), which we denote by

$$u_s^*(\cdot, x_s) = \underset{u_s}{\operatorname{argmin}} J_s^N(x_s, u_s).$$

Hence, the cumulative costs of the overall system for the DMPC setting are given by

$$V^N(x^0) = \sum_{i=1}^S J_s^N(x_i^0, u_s^*(\cdot, x_i^0)) \quad (6)$$

#### 2.5. Feedback of the subsystem

Last, we obtain the feedback  $\mu_s^N$  for each subsystem as the first element of the minimizer  $u_s^*(\cdot, x_s)$  from (6), i.e.

$$\mu_s^N(x_s^0) = u_s^*(0, x_s^0) \quad (7)$$

Note that the feedback is implicitly coupled with the success of reserving a path item  $(n, k, m)$  of the grid in the optimization and coordination step. If more than one system attempt to get a reservation for the same cell at the same time  $n$ , only the highest priority path item is accepted. This fulfills the constraint formulated in (3). The other inquires are rejected and the systems have to attempt a new reservation at a later time or consider a different field to attempt a reservation.

### 3. Stability conditions

Our goal is to show asymptotic stability of the overall system. We use [3, Def. 2.14] to derive an upper bound for our system, which will guarantee the asymptotic stability of the system. To define the required bounding function  $\beta(r, n) \in \mathcal{KL}$ , we introduce the set  $\mathcal{K}$  describing the functions  $\alpha : \mathbb{R}_0^+ \rightarrow \mathbb{R}_0^+$ , which are strictly increasing and satisfy  $\alpha(0) = 0$ . If additionally  $\alpha$  is unbounded, then it called a class  $\mathcal{K}_\infty$  function. Secondly, we call functions  $\delta : \mathbb{R}_0^+ \rightarrow \mathbb{R}_0^+$  to be of class  $\mathcal{L}$ , if  $\delta$  is continuous and strictly decreasing with  $\lim_{n \rightarrow \infty} \delta(n) = 0$ . Combining these properties, we call  $\beta(r, n)$  to be a class  $\mathcal{KL}$  function if  $\beta(r, n)$  is continuous and  $\beta(\cdot, n) \in \mathcal{K}$  for all  $n \geq 0$  and  $\beta(r, \cdot) \in \mathcal{L}$  for all  $r \geq 0$ . Based on the asymptotically stability we can show that our system is optimal controllable. This is presented in the numerical results 4 where the suboptimality  $\alpha$  is used to evaluate the performance of the controller.

Note that for our intersection scenario, we cannot guarantee that a minimizer  $u_s^*$  to exist such that costs are decreasing for each car in each time step. Indeed, the constraints (2) and (3) allow cars to hold their position for a number of time steps to let other cars pass. Yet, some cars move closer to their target and the respective costs of these subsystems are decreasing. Hence, only an overall consideration is promising.

In the following, we prove stability of the presented intersection scenario by showing that for  $r := |x_s - x_s^*|$

$$\beta(r, n) = S \max\{\bar{M} - \underline{M} - n, 0\} \underline{M}^2 + \lambda S \underline{M} + \sum_{j=1}^{\underline{M} - \max\{0, n - \bar{M} + \underline{M}\}} j^2$$

is a class  $\mathcal{KL}$  function and by constructing an upper bound for the overall costs. To this end, we first derive a bound  $\beta(r, 0)$  for the initial conditions, which we extend in a second step to an upper bound for all time instances  $n$ . Here, we abbreviate the lower bound of the path length (2) as  $\underline{M} = \min(\#x_s)$  and the upper bound as  $\overline{M} = \max(\#x_s)$ .

First, we ignore (3) and define  $\beta(r, 0)$  as a function, where all  $S$  cars stay in their initial position until they have to leave to get to the target without violating  $\overline{M}$ . Due to ignoring the collision constraint (3),  $\beta(r, 0)$  represent an upper bound on the costs. We split this expression into two subsums:

$$\beta(r, 0) = \sum_{s=1}^S \left( \sum_{k=0}^{\overline{M}-\underline{M}-1} \ell(x_s, 0) + \sum_{k=\overline{M}-\underline{M}}^{\overline{M}-1} \ell(x_s(k), u_s(k)) \right).$$

Since by assumption cars remain at their initial position, we have  $u_s(k) = 0, k = 0, \dots, \overline{M} - \underline{M} - 1$  and can dissolve the inner sums to

$$\begin{aligned} \beta(r, 0) &= \sum_{s=1}^S \left( (\overline{M} - \underline{M}) \|x_s^0 - x_s^*\|^2 + \sum_{k=\overline{M}-\underline{M}}^{\overline{M}-1} \|x_s(k) - x_s^*\|^2 + \lambda \|u_s\|^2 \right) \\ &= \sum_{s=1}^S \left( (\overline{M} - \underline{M}) \|x_s^0 - x_s^*\|^2 + \sum_{k=0}^{\underline{M}-1} \|x_s(k) - x_s^*\|^2 + \lambda \|u_s\|^2 \right), \end{aligned}$$

which allows us to shift the interval for the second sum to  $k = 0, \dots, \underline{M} - 1$ . By our model definition, we obtain  $\|x_s^0 - x_s^*\| = \underline{M}$  for the initial position and  $\|x_s(k) - x_s^*\| = \underline{M} - k$  as a car moves towards its target with  $\|u_s\|^2 = 1$ . Hence, we obtain

$$\beta(r, 0) = \sum_{s=1}^S \left( (\overline{M} - \underline{M}) \underline{M}^2 + \sum_{k=0}^{\underline{M}-1} (\underline{M} - k)^2 + \lambda \underline{M} \right).$$

Dissolving now the inner sum over the systems we get

$$\beta(r, 0) = S \left( -\underline{M}^3 + \underline{M}^2 \overline{M} + \underline{M} \lambda + \sum_{k=1}^{\underline{M}} k^2 \right). \quad (8)$$

With the application of square pyramidal number for the last remaining sum in (8)

$$\sum_{k=1}^{\underline{M}} k^2 = \frac{1}{6} (\underline{M} + 1) \underline{M} (2\underline{M} + 1)$$

we can simplify (8) to

$$\beta(r, 0) = S \left( -\frac{2}{3} \underline{M}^3 + \underline{M}^2 \left( \overline{M} + \frac{1}{2} \right) + \underline{M} \left( \lambda + \frac{1}{6} \right) \right), \quad (9)$$

which reveals an upper bound on the initial cost. To construct the decrease of the bound over time, we consider the sum over the subsystems  $S$

$$\beta(r, 0) = \sum_{s=1}^S \left( \frac{1}{3} \|x_s(k) - x_s^*\|^3 + \|x_s(k) - x_s^*\|^2 \left( k + \frac{1}{2} \right) + \|x_s(k) - x_s^*\| \left( \lambda + \frac{1}{6} \right) \right).$$

Now, we can use (9) to incorporate the time  $n$  in  $\beta(r, n)$ . The upper bound over time resembles the sequential steps of all cars. Hence, as time progresses, we get

$$\beta(r, n) = \sum_{s=1}^S \left( \sum_{k=0}^{\max\{\bar{M}-1-n, 0\}} \ell(x_s(0), 0) + \sum_{k=\max\{\bar{M}-\underline{M}-n, 0\}}^{\max\{\bar{M}-1-n, 0\}} \ell(x_s(k+n), u_s(k+n)) \right).$$

Again, the first inner sum describes those cars waiting at their position until they must start to reach their target. The second sum represents the costs over the interval during which the cars move towards their target. Dissolving the first sum reveals

$$\beta(r, n) = S \max\{\bar{M} - \underline{M} - n, 0\} \underline{M}^2 + S \sum_{k=\max\{\bar{M}-\underline{M}-n, 0\}}^{\max\{\bar{M}-1-n, 0\}} \ell(x_s(k), u_s(k)).$$

By shifting the index of the remaining sum to  $k = 0$ , we get

$$\begin{aligned} \beta(r, n) &= S \max\{\bar{M} - \underline{M} - n, 0\} \underline{M}^2 \\ &\quad + S \sum_{k=0}^{\max\{\bar{M}-1-n, 0\} - \max\{\bar{M}-\underline{M}-n, 0\}} \ell(x_s(k + \max\{\bar{M} - \underline{M} - n, 0\}), u_s(k + \max\{\bar{M} - \underline{M} - n, 0\})). \end{aligned}$$

As we combine the difference of the maxima of the upper limit, we get

$$\begin{aligned} \beta(r, n) &= S \max\{\bar{M} - \underline{M} - n, 0\} \underline{M}^2 \\ &\quad + S \sum_{k=0}^{\underline{M}-1} \ell(x_s(k + \max\{\bar{M} - \underline{M} - n, 0\}), u_s(k + \max\{\bar{M} - \underline{M} - n, 0\})). \end{aligned}$$

Resetting  $k := k - n$ , the latter expression simplifies to

$$\begin{aligned} \beta(r, n) &= S \max\{\bar{M} - \underline{M} - n, 0\} \underline{M}^2 \\ &\quad + S \sum_{k=0}^{\underline{M}-1} \ell(x_s(k - n + \max\{\bar{M} - \underline{M}, n\}), u_s(k - n + \max\{\bar{M} - \underline{M}, n\})). \end{aligned}$$

As  $\bar{M} - \underline{M} \geq n$ , we can eliminate the maximum terms in the index and get therefore

$$\begin{aligned} \beta(r, n) &= S \max\{\bar{M} - \underline{M} - n, 0\} \underline{M}^2 \\ &\quad + S \sum_{k=0}^{\underline{M}-1} \ell(x_s(k - n + \bar{M} - \underline{M}), u_s(k - n + \bar{M} - \underline{M})). \end{aligned} \quad (10)$$

and for  $\bar{M} - \underline{M} \leq n$

$$\beta(r, n) = S \max\{\bar{M} - \underline{M} - n, 0\} \underline{M}^2 + S \sum_{k=0}^{\underline{M}-1} \ell(x_s(k), u_s(k)). \quad (11)$$

as distinction of both cases.

As  $k + \bar{M} - \underline{M} - n \leq k + \bar{M} - \underline{M} \leq k + \bar{M}$  for case (10) we obtain

$$\beta(r, n) = S \max\{\bar{M} - \underline{M} - n, 0\} \underline{M}^2 + S \sum_{k=0}^{\underline{M}-1} (k + \bar{M})^2 + \lambda S 1$$

and for the second case (11)

$$\beta(r, n) = S \max\{\bar{M} - \underline{M} - n, 0\} \underline{M}^2 + S \sum_{k=0}^{\underline{M}-1} k^2 + \lambda S 1$$

For the first case (10) we can assume that for the index  $k + \bar{M} - \underline{M} - n$  the inequality  $k \leq n - \bar{M} + \underline{M}$  and for the second case (11)  $k \leq \underline{M} - 1$  as given by the interval.

Therefore we can dissolve both cases by substitute with  $j := \underline{M} + k$  and  $\max\{0, n - \bar{M} + \underline{M}\} = \underline{M} - j$  to get  $\underline{M} - 1 = \underline{M} - j$  which leads to  $j = 1$  for the lower bound and we get

$$\beta(r, n) = S \max\{\bar{M} - \underline{M} - n, 0\} \underline{M}^2 + \lambda S \underline{M} + \sum_{j=1}^{\underline{M} - \max\{0, n - \bar{M} + \underline{M}\}} j^2$$

Hence, by construction we have that  $\beta(r, n) \in \mathcal{KL}$  and an upper bound of the overall costs satisfying the conditions of [3, Def. 2.14]. Utilizing the properties of the system and of  $\underline{M}$ ,  $\bar{M}$ , the function  $\beta(r, n)$  may largely overestimate the costs of the overall system. These properties allowed us to exclude (3) and avoid a collision detection as required in the closed loop. Thereby, we can conclude asymptotic stability of the overall closed loop system.

Within the analysis of the constructed feedback law, we are not only interested in stability of the closed loop, but also in the performance of the control. Since the bound  $\beta(r, n)$  is a worst case bound, the performance estimate that can be derived from it is rather poor. Additionally, the values  $\underline{M}$ ,  $\bar{M}$  may be unknown for example in that case as we cannot predict the maximum wait time of one car which can prolong the path length. To circumvent these issues, we make use of the concept of relaxed Lyapunov arguments in the trajectory based setting. In particular, [3, Proposition 7.6] shows that if there exist a constant  $\alpha \in (0, 1]$  such that the relaxed Lyapunov inequality

$$V^N(x(n)) \geq \alpha \ell(x(n), \mu^N(x(n))) + V^N(x(n+1)) \quad (12)$$

holds for all  $n \in \mathbb{N}_0$  and additionally there exist functions  $\alpha_1, \alpha_2, \alpha_3 \in \mathcal{K}_\infty$  such that

$$\alpha_1(\|x(n)\|) \leq V^N(x(n)) \leq \alpha_2(\|x(n)\|), \quad (13)$$

$$\alpha_3(\|x(n)\|) \leq \ell(x(n), \mu^N(x(n))), \quad (14)$$

then the closed loop solution  $x(n)$  behaves like a trajectory of an asymptotically stable system and the performance bound

$$J^\infty(x(0), \mu^N(x_s(0))) \leq V^N(x(0)) / \alpha \leq V^\infty(x(0)) / \alpha$$

holds.

Since we have shown the existence of a bound  $\beta(r, n)$ , we can apply converse Lyapunov results to guarantee the existence of  $\alpha_1, \alpha_2, \alpha_3 \in \mathcal{K}_\infty$  such that (13), (14) hold, cf. [12]. This allows us to concentrate on the performance evaluation of the proposed control scheme via (12).

#### 4. Experiments

For the implementation we use the discrete event simulator framework adevs [13] along with the Qt framework [14]. The cars are modelled as multi-agents objects, which include a controller instance, dynamical properties, constraints and a cost function with individual targets. The grid is implemented as an array containing all grid cells. Upon initialization, each car reserves a time and grid cell tuple in first-come-first-serve fashion. Hence, if a car enters the scenario at a later time instant, its initial time will also be later. To retain a unique order for the reservation process, the cars are indexed. For the optimization step within the MPC, we apply the COBYLA Algorithm (Constrained Optimization BY Linear Approximations) developed by Powell [15] provided by the NLOpt library [16]. Minimizing the cost function including the individual target of the car implicitly defines the direction and route of the car. The model of the middle lane is given as a set of constraints which are defined for each car depending on its start position. Both the path and control values  $x_s$ ,  $u_s$  are mapped from tuples to absolute values of the cell grid. In particular, we map the cell coordinate tuples  $(n, k, m)$  to global scalars  $w$  via

$$\vec{w} = (w_1, \dots, w_N), w_i \in \{0, \dots, L_k L_m - 1\}$$

with

$$w_i = k_i + m_i L_k.$$

If a car reaches its target, the car is removed from the scenario and is not considered in further calculations. We set up a standard intersection scenario with a  $4 \times 4$ -intersection grid and 20 cars starting in the corners. To guarantee a connected path, we imposed a penalty term  $\eta$ . Here, a path is connected if each pair of consecutive path items are neighbors in the grid. The penalty is defined as

$$\eta_s = Bv,$$

where  $v$  is the number of not connected pairs of consecutive path items and  $B$  a constant.

To analyze the performance of the controller, we utilize the minimal and maximal path lengths

$$p_{\min} = \min_s \#x_s \geq \underline{M}, \quad p_{\max} = \max_s \#x_s \leq \overline{M},$$

the mean path length

$$\bar{p} = \frac{1}{S} \sum_{s=0}^S \#x_s.$$

and the suboptimality bound  $\alpha$  from (12). We evaluate the bound  $\alpha$  as the minimum over all time steps  $n \in \mathbb{N}_0$  and compute the mean over all systems  $x_1, \dots, x_S$  via

$$\alpha_n = \frac{\sum_{i=1}^S \min\left(\alpha_{n-1}, \frac{V_{n-1} - V_n}{\ell_{n-1}}\right)}{S} \quad \text{with } \alpha_{-1} = 1.$$

For  $\alpha$  we used the value in the last step named here  $\alpha_n$  and  $\bar{\alpha}$  as the mean over all time steps.

For our experiments, we considered the parameters  $\lambda = 0.2$  and  $B = 100$  and a horizon length of  $N = 3$ .

We set up configurations with different quantities of cars and the intersection size. Pursuing a realistic approach we consider a  $4 \times 4$ -intersection scenario, i.e. two lanes for each direction,

**Table 1.** Experimental results for test scenarios,  $\alpha_n$  as last time step,  $\bar{\alpha}$  as mean value over all time steps in the simulation

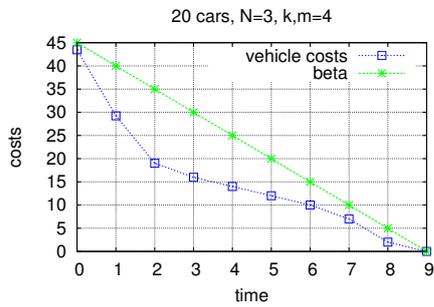
$\#S$	Grid	$N$	$p_{\min}$	$p_{\max}$	$\bar{p}$	$\alpha_n$	$\bar{\alpha}$
20	$4 \times 4$	3	4	11	6.65	0.25	0.2155
30	$4 \times 4$	3	4	10	6.2	0.27	0.2707
20	$6 \times 6$	3	5	18	7.25	0.0019	0.1005
30	$6 \times 6$	3	5	13	6.76	0.00013	0.1639
20	$6 \times 6$	4	5	11	6.55	0.00587	0.43
30	$6 \times 6$	4	5	16	6.8	0.00588	0.3587

and a  $6 \times 6$ -intersection scenario reflecting 3 lanes for each direction. In Table 1, we present our numerical results for both scenarios with 20 and 30 cars for each direction.

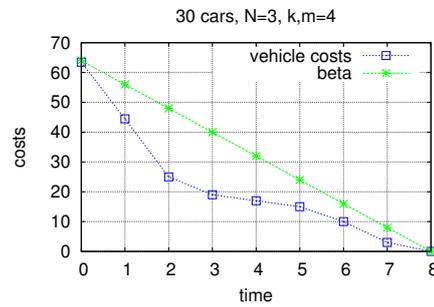
In the  $4 \times 4$ -intersection we observe a slightly larger mean path length  $\bar{p}$  for 30 cars than for 20 cars. This due to higher reservation times for the direct way from the start to the target cell of a vehicle as more cars try to reserve the direct way. With the growing reservation time values for the direct lane, the minimum of cost function is attained using the second lane. Hence, cars entering the intersection at a later time step choose the second lane. Reflecting this situation we can argue that a certain amount of cars is necessary that the cost function gives an advantage over distance for a shorter time delay. For  $\alpha$  we used the value in the last step named here mentioned  $\alpha_n$  and  $\bar{\alpha}$  as the mean over all time steps.

For the  $6 \times 6$ -scenario, we similarly observe that the mean path length  $\bar{p}$  is smaller for the configuration of 30 cars than with 20 cars. The effect is also increased with the greater intersection dimensions. Hence, we can conclude that the influence of the higher distances for the cars leads to the greater gap in  $\bar{p}$ . Additionally, we considered different horizon lengths and observed an decrease in  $\bar{p}$  for larger horizon for 20 cars. For the scenario with 30 cars  $\bar{p}$  changes only slightly. Again, the reservation of more cells in the optimization has a large impact on the difference of  $\bar{p}$  for 20 cars with greater horizon length. The time slots which are reserved by the cars reveal later times for late arriving cars. Hence, the cost function leads to an earlier change to the second lane. Yet, we also observed an improvement in the suboptimality index for both scenarios which improves greatly for the larger horizon length.

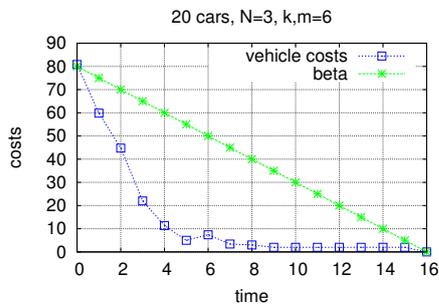
In the following figures, we give some graphs for the overall cost function according to Table 1. We included  $\beta(r, n)$  as a monotonously decreasing function representing the upper bound for our cost function. For longer horizons illustrated in Figure 6 and 7, we observe that the cost function tends to zero more slowly. In this case, the control impact is less aggressive compared to a shorter horizon, which is a known typical effect of using MPC. In the figures 4-7 a slightly increase of the costs occurs in one step. As mentioned in the 4x4-evaluation, this is caused by changing to the second lane of the later following cars. As the cost function returns lower costs due to earlier available reservation times for the second lane over the optimization horizon for one car than reserving later reservation times in the crowded cells, the applied optimal control  $u(0)$  increases the costs for the changing cars slightly. Still, the costs are decreasing and remain bounded by  $\beta(r, n)$ , which indicates that the solutions show stable behavior.



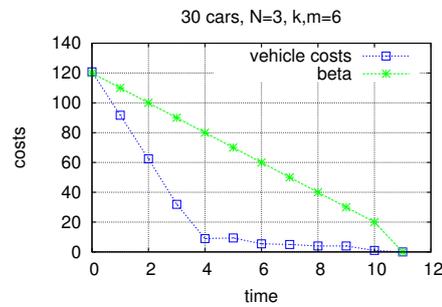
**Figure 2.** Overall costs for 20 cars,  $N=3$ ,  $k,m=4$



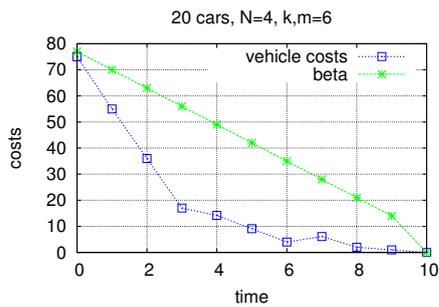
**Figure 3.** Overall costs for 30 cars,  $N=3$ ,  $k,m=4$



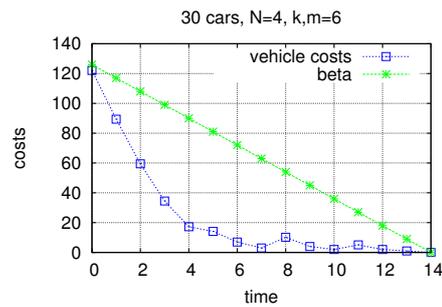
**Figure 4.** Overall costs for 20 cars,  $N=3$ ,  $k,m=6$



**Figure 5.** Overall costs for 30 cars,  $N=3$ ,  $k,m=6$



**Figure 6.** Overall costs for 20 cars,  $N=4$ ,  $k,m=6$



**Figure 7.** Overall costs for 30 cars,  $N=4$ ,  $k,m=6$

**5. Conclusion**

In this paper we used an intersection scenario as a discrete shared space grid. We formulated this scenario as a Distributed Model Predictive Control Problem in Section 2 where each car

modelled as an multi-agent optimizes its own path. In Section 3 we have proven the stability for the overall system by deriving a class  $\mathcal{KL}$  boundary function. Our numerical results showed that the suboptimality index  $\alpha$  known from relaxed Lyapunov theory remains positive. Additionally, the cumulated costs are decreasing in every time step and satisfy our computed bound, which indicates stable behavior of the system.

In further research we will consider an exponential term depending on time to privilege long waiting cars. Converting the cost function into a continuous one will also extend the suitability for continuous optimization algorithms. Additionally, we plan to incorporate the risk of deadlocks by allowing any direction for the lanes to increase the throughput for the overall traffic. Last, by including path predictions of non-communicating participants we will improve practicability of the coordination.

### Acknowledgment

We thank Prof. C. Müller-Schloer from the Leibniz University of Hannover for a fruitful discussion on this topic.

### References

- [1] Qin S J and Badgwell T A 2003 A survey of industrial model predictive control technology *Control Engineering Practice* **11** 733–764
- [2] Rawlings J B and Mayne D Q 2009 *Model Predictive Control: Theory and Design* (Nob Hill Pub.)
- [3] Grüne L and Pannek J 2011 *Nonlinear Model Predictive Control: Theory and Algorithms* Communications and Control Engineering (London: Springer)
- [4] Keerthi S S and Gilbert E G 1988 Optimal Infinite Horizon Feedback Laws for a General Class of Constrained Discrete-Time Systems: Stability and Moving Horizon Approximations *Journal of Optimization Theory and Applications* **57** 265–293
- [5] Chen H and Allgöwer F 1998 A Quasi-Infinite Horizon Nonlinear Model Predictive Control Scheme with Guaranteed Stability *Automatica* **34** 1205–1217
- [6] Grüne L 2012 NMPC without terminal constraints *IFAC Proceedings Volumes (IFAC-PapersOnline)* **4** 1–13
- [7] Richards A and How J 2003 Model predictive control of vehicle maneuvers with guaranteed completion time and robust feasibility *Proceedings of the 2003 American Control Conference, 2003.* **5** 4034–40
- [8] Chaaban Y, Hähner J and Müller-Schloer C 2009 Towards Fault-Tolerant Robust Self-Organizing Multi-agent Systems in Intersections without Traffic Lights *2009 Computation World: Future Computing, Service Computation, Cognitive, Adaptive, Content, Patterns* 467–475
- [9] Richter U and Mnif M 2006 Towards a generic observer / controller architecture for Organic Computing *GI Jahrestagung* **1** 112–119
- [10] Hart P E, Nilsson N J and Raphael B 1968 A Formal Basis for the Heuristic Determination of Minimum Cost Paths *IEEE Transactions of systems science and cybernetics* 100–107
- [11] Makarem L and Gillet D 2013 Model predictive coordination of autonomous vehicles crossing intersections *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC* 1799–04
- [12] Jiang Z P and Wang Y 2002 A converse Lyapunov theorem for discrete-time systems with disturbances *Systems and Control Letters* **45** 49–58
- [13] Nutaro J 2013 adevs (A Discrete Event Simulator) library <http://web.ornl.gov/~1qn/adevs/>
- [14] Digia Qt Framework
- [15] Powell M J D 1998 Direct search algorithms for optimization calculations *Acta Numerica* **7** 287–336
- [16] Johnson S The {NLopt} nonlinear-optimization package

## Occupancy Grid based Distributed MPC for Mobile Robots\*

Mohamed W. Mehrez<sup>1</sup>, Tobias Sprodowski<sup>2</sup>, Karl Worthmann<sup>3</sup>, George K.I. Mann<sup>1</sup>, *Member, IEEE*  
Raymond G. Gosine<sup>1</sup>, Juliana K. Sagawa<sup>2,4</sup> and Jürgen Pannek<sup>2</sup>

**Abstract**—In this paper, we introduce a novel approach of reducing the communication load in distributed model predictive control (DMPC) for mobile robots. The key idea is to project the predicted state trajectory onto a grid resulting in an occupancy grid prediction. This approach has the advantage of utilizing continuous optimization methods while only quantized information is exchanged. We consider non-holonomic mobile robots to numerically and experimentally investigate the proposed method.

### I. INTRODUCTION

In formation control, the control objective is to coordinate a group of robots to first form, and then maintain a prescribed formation. This can be done either in a cooperative or non-cooperative manner, see, e.g. [1] or [2]. In the latter case, each robot has its own reference point with respect to dynamic (e.g. collision avoidance) and static constraints (e.g. operating region restrictions). Most of formation control strategies presented in the literature do not account for actuator saturation limits of robots, see [3] for a survey.

In this paper, we utilize a model predictive control (MPC) scheme to stabilize a group of autonomous non-cooperative differential drive robots (formation stabilization) to individual reference equilibria while avoiding collisions. In MPC, we first measure the current state of the system, which serves as a basis for solving a finite horizon optimal control problem (OCP). This results in a sequence of future control values. Then, the first element of the computed sequence is applied before the process is repeated, cf. [4] for an overview on non-linear MPC. Within this setting, the OCP cost function allows us to encode a control objective and to evaluate the closed-loop performance. A key advantage of MPC is that static and dynamic constraints can be directly taken into account [5].

MPC has been used in several studies considering formation control, see, e.g. [3], [6], [7]. Here, we consider a distributed implementation of MPC (DMPC), in which each subsystem represents a single robot. The robots solve

their own (local) optimization tasks. However, the objective of collision avoidance induces a strong coupling among the robots, which may render the control task infeasible in decentralized control (no communication), see, e.g. [8] or [9], for a delineation of concepts w.r.t. decentralized, distributed, and centralized MPC. Hence, data exchange between the robots is a necessary prerequisite for feasibility. Here, the individual optimization tasks are executed in a fixed order similar to the method proposed by Richards and How [10].

Previous studies [11] were based on the exchange of the predicted state trajectories to formulate the coupling constraints within each OCP. Since our goal is to reduce the communication load, we first partition the operating region into a grid. Then, the predicted trajectories are projected onto the grid resulting in an occupancy grid prediction, which serves as a quantization of the communication data. Based on an exchange of these projections, each robot formulates suitable coupling constraints. Utilizing the occupancy grid reduces communication bandwidth utilization and congestion issues since a more compact data representation (integers instead of floating point values) is employed. To construct the constraints, we employ squireles as an approximation for a grid cell to use gradient-based algorithms for optimization. In summary, the optimization is performed in a spatial set to make use of sensitivity information while the communication is conducted in a quantized manner to reduce the communication load.

Furthermore, we derive an estimate on the minimum size of a grid cell and numerically compute the minimal horizon length such that a desired closed-loop performance is achieved. Finally, we demonstrate the real-time applicability of the proposed controller via laboratory experiments adopting two research non-holonomic mobile robots.

**Notation:**  $\mathbb{R}$ ,  $\mathbb{N}$  and  $\mathbb{Z}$  denote the real, natural and integer numbers, respectively.  $\mathbb{N}_0 := \mathbb{N} \cup \{0\}$  represents the non-negative integers and  $\mathbb{R}_{\geq 0}$  the non-negative real numbers. For a variable  $x \in \mathbb{R}$ , the floor operator  $\lfloor \cdot \rfloor$  is defined as  $\lfloor x \rfloor := \max\{m \in \mathbb{Z} : m \leq x\}$ . For given integers  $a, b \in \mathbb{N}_0$ , the set  $\{n \in \mathbb{N}_0 : a \leq n \leq b\}$  is abbreviated by  $[a : b]$ . Moreover, for a vector  $x \in \mathbb{R}^n$ ,  $n \in \mathbb{N}$ , the infinity norm  $\|\cdot\|_\infty$  is defined as  $\|x\|_\infty := \max\{|x_i| : i \in [1 : n]\}$ .

### II. PROBLEM SETTING

Firstly, we present the kinematics of the considered robots and a respective grid representation of the operating region. Thereafter, we introduce the optimal control problem (OCP) of each robot before these components are integrated in a DMPC scheme.

\*This work is supported by the Deutsche Forschungsgemeinschaft grants WO 2056/1 and WO 2056/4-1, Natural Sciences and Engineering Research Council of Canada (NSERC), the Research and Development Corporation (RDC), and C-CORE J.I. Clark Chair.

<sup>1</sup>M.W. Mehrez, G.K.I. Mann, and R.G. Gosine are with Intelligent Systems Lab, Memorial University of Newfoundland, St. John's, NL, Canada. [m.mehrez.said, gmann, rgosine]@mun.ca

<sup>2</sup>T. Sprodowski, J.K. Sagawa and J. Pannek are with Department of Production Engineering, University of Bremen and BIBA Bremer Institut für Produktion und Logistik GmbH, Bremen, Germany. [spr, sag, pan]@bibu.uni-bremen.de

<sup>3</sup>K. Worthmann is with Institute for Mathematics, Technische Universität Ilmenau, Ilmenau, Germany. karl.worthmann@tu-ilmenau.de

<sup>4</sup>J.K. Sagawa is with Department of Production Engineering, Federal University of São Carlos, Brazil. juliana@dep.ufscar.br

The control objective is to stabilize a group of  $P$  non-holonomic mobile robots,  $P \in \mathbb{N}$ , at reference equilibria while avoiding collisions. Each robot is considered as part of a multi-agent system with state  $z_p \in Z \subset \mathbb{R}^3$ ,  $p \in [1 : P]$ . For the  $p^{\text{th}}$  robot, the discrete time kinematics is given by

$$\begin{aligned} z_p(n+1) &= f(z_p(n), u_p(n)), \\ &= z_p(n) + \begin{pmatrix} \cos(\theta_p(n)) \\ \sin(\theta_p(n)) \\ 0 \end{pmatrix} v_p(n) + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \omega_p(n) \end{aligned} \quad (1)$$

with vector field  $f : \mathbb{R}^3 \times \mathbb{R}^2 \rightarrow \mathbb{R}^3$ . Here,  $n$  denotes the current time. The state  $z_p$  contains position and orientation coordinates, i.e.  $z_p = (x_p, y_p, \theta_p)^\top$  (m,m,rad) $^\top$ . The control  $u_p = (v_p, \omega_p)^\top \in U \subset \mathbb{R}^2$  (m/s,rad/s) $^\top$  consists of the linear speed  $v_p$  and the angular speed  $\omega_p$  of the robot. Each robot has an initial state  $z_p(0)$  and a reference  $z_p^*$ . State and control constraints are incorporated in

$$Z := [-\bar{x}, \bar{x}] \times [-\bar{y}, \bar{y}] \times \mathbb{R}, \quad U := [\underline{v}, \bar{v}] \times [\underline{\omega}, \bar{\omega}]$$

for  $\bar{x}, \bar{y}, \bar{v}, \bar{\omega} > 0$  and  $\underline{v}, \underline{\omega} < 0$ , i.e. the control constraints can be asymmetric.

We partition the set  $[-\bar{x}, \bar{x}] \times [-\bar{y}, \bar{y}]$  into a grid of squared cells. Each cell has a cell size  $c$ , where  $c$  satisfies  $2\bar{x} = a_{\max}c$  and  $2\bar{y} = b_{\max}c$  with  $a_{\max}, b_{\max} \in \mathbb{N}$ . Each cell in the grid has a unique index  $(a, b) \in \mathcal{G}$  where  $\mathcal{G} := [0 : a_{\max} - 1] \times [0 : b_{\max} - 1] \subset \mathbb{N}_0^2$ . Then, a state  $z_p$  (and its location  $(x_p, y_p)$ ) can be mapped to the discrete set  $\mathcal{G}$  using the quantization  $q : Z \rightarrow \mathcal{G}$  defined as

$$q(z_p) = (a_p, b_p) := \left( \min \left\{ a_{\max}, \left\lfloor \frac{x_p + \bar{x}}{c} \right\rfloor \right\}, \min \left\{ b_{\max}, \left\lfloor \frac{y_p + \bar{y}}{c} \right\rfloor \right\} \right).$$

We next present the required notation to set up the DMPC algorithm. To this end, we define our quantized trajectory, communication scheme, and stage costs to compose the optimal control problem (OCP). For a finite control sequence  $u_p = (u_p(0), u_p(1), \dots, u_p(N-1))$  and an initial value  $z_p^0$ , the predicted state trajectory of robot  $p$  over prediction horizon  $N \in \mathbb{N}$  is given by

$$z_p^u(\cdot; z_p^0) := (z_p^u(0; z_p^0), z_p^u(1; z_p^0), \dots, z_p^u(N; z_p^0)).$$

Here, we removed the subscript  $p$  from  $u_p$  in  $z_p^u(\cdot; z_p^0)$  to simplify our notation. The trajectory together with the introduced quantization  $q$  allows us to define the occupancy grid prediction  $\mathcal{I}_p(n) \in (\mathbb{N}_0 \times \mathcal{G})^{N+1}$  at time  $n$  as

$$\begin{aligned} \mathcal{I}_p(n) &:= (n+k, q(z_p^u(k; z_p^0)))_{k \in [0:N]} \\ &:= (n+k, a_p^u(k; z_p^0), b_p^u(k; z_p^0))_{k \in [0:N]}. \end{aligned}$$

where  $(a_p^u(k; z_p^0), b_p^u(k; z_p^0))$  denotes the quantized state. While each robot  $p$  sends only one such occupancy grid prediction containing the quantized prediction  $\mathcal{I}_p(n)$ , it collects all received occupancy grid predictions in  $i_p$ , i.e.

$$i_p(n) := (\mathcal{I}_1(n), \dots, \mathcal{I}_{p-1}(n), \mathcal{I}_{p+1}(n), \dots, \mathcal{I}_P(n)).$$

Later, we use this data to construct the coupling constraints to ensure collision avoidance. The main motivation of communicating the occupancy grid prediction instead of the predicted trajectories is to transmit integers instead of floating point values to reduce the communication load.

Here, we utilize a DMPC scheme, which implements a local controller for each robot  $p$ . To this end, for a given reference point  $z_p^*$ , we choose the (local) continuous stage costs  $\ell_p : Z \times U \rightarrow \mathbb{R}_{\geq 0}$ , which are supposed to be positive definite with respect to  $z_p^*$ . Then, at every time instant  $n$ , each robot  $p$  solves the following finite horizon optimal control problem for initial condition  $z_p^0 = z_p(n)$  and given data  $i_p(n)$ :

$$\min_{u_p} J_p^N(u_p; z_p^0, i_p(n)) := \sum_{k=0}^{N-1} \ell_p(z_p^u(k; z_p^0), u_p(k)) \quad (2)$$

subject to the constraints

$$\begin{aligned} z_p^u(k+1; z_p^0) &= f(z_p(k; z_p^0), u_p(k)), \quad k \in [0 : N-1], \\ u_p(k) &\in U, \quad k \in [0 : N-1], \\ G(z_p^u(k; z_p^0), i_p(n)) &\leq 0, \quad k \in [1 : N], \\ z_p^u(k; z_p^0) &\in Z, \quad k \in [1 : N], \end{aligned}$$

see Subsection III-B for details on  $G$ . As a result, an optimal control sequence

$$u_p^* = (u_p^*(0), \dots, u_p^*(N-1))$$

is obtained which minimizes the cost function (2). Existence of a minimizer is ensured since a continuous objective on a compact domain is considered. The corresponding value function  $V_p^N : Z \times (\mathbb{N}_0 \times \mathcal{G})^{(P-1)(N+1)} \rightarrow \mathbb{R}_{\geq 0}$  is defined as

$$V_p^N(z_p^0, i_p(n)) := J_p^N(u_p^*; z_p^0, i_p(n)).$$

The detailed construction of the collision avoidance constraints  $G$  will be explicated in the ensuing Section III. Using this formulation of the OCP, in analogy to Richards and How [10], [12] the DMPC algorithm is as follows:

---

**Algorithm 1** DMPC-algorithm for the overall system

---

- 1: **Communicate**  $\mathcal{I}_p(0) := (k, q(z_p^0))_{k \in [0:N]}$  among all robots for all  $p \in [1 : P]$ .
  - 2: **for** time instant  $n = 0, 1, \dots$  **do**
  - 3:   **for** robot  $p$  from 1 to  $P$  **do**
  - 4:     **Receive**  $i_p(n)$ .
  - 5:     **Solve** OCP (2) and **Apply**  $u_p^*(0)$ .
  - 6:     **Broadcast**  $\mathcal{I}_p(n)$ .
  - 7:   **end for**
  - 8: **end for**
- 

Note that, in the initialization of Algorithm 1, the quantization of the initial state of each robot is communicated among all robots. Moreover, the individual OCP's are solved sequentially following a fixed ordering (priority rule) before the control values are applied to all robots. The ordering, however, induces asymmetric information: when robot  $p$  is

solving its OCP, the received occupancy grid predictions from the previous robots  $(1, \dots, p-1)$  contain the information for the full horizon length for the current time instant  $n$ . Yet for the subsequent robots  $(p+1, \dots, P)$ , the predictions are received at the previous time instant  $n-1$  and, therefore, the last prediction state  $\mathcal{I}_q(n)(N), q \in [p+1:P]$  is missing. To resolve this issue, for  $q \in [p+1:P]$ , we extend the occupancy grid prediction by setting  $\mathcal{I}_q(n)(N) = \mathcal{I}_q(n)(N-1)$ .

In the generic case, the calculation of an admissible set of controls for all robots in the initialization of Algorithm 1 is particularly demanding. In our setting without terminal constraints, using the control  $u_p \equiv 0$  ensures initial feasibility. Moreover, by our choice of the extension of the occupancy grid predictions, for robot  $p$  we can always construct a feasible control via  $u_p := (u_p^*(1), \dots, u_p^*(N), 0)$ , which shows recursive feasibility of the problem. Then, the first robot starts to improve this initial guess during the runtime of the DMPC algorithm. We like to stress that, in contrast to [10], [12], initial feasibility is not sufficient for stability in the considered setting. Instead stability conditions similar to [11] need to be verified, which is outside the scope of this paper. For advanced techniques regarding recursive feasibility in the absence of terminal costs and constraints, we refer to [13].

### III. FORMULATION OF THE CONSTRAINTS

In this section, we derive an appropriate cell size  $c$  based on the kinematics and geometry of the system and specify a safety distance between two robots to sufficiently ensure the collision avoidance constraints in and between two sampling instants  $n$  and  $n+1$ . Thereafter, we expound a method to set up collision avoidance constraints for the OCP (2) which is based on a squirrel approximation.

To formulate the coupling constraints, we first define the backward mapping  $f_c: \mathbb{N}_0 \times \mathcal{G} \rightarrow \mathbb{R}^2$  given by

$$f_c((n, a, b)) = \underbrace{(c \cdot (a + 0.5) - \bar{x}, c \cdot (b + 0.5) - \bar{y})^\top}_{=: (x^c, y^c)_{(a,b)}}, \quad (3)$$

which transforms a cell index  $(a, b) \in \mathcal{G}$  to the corresponding cell center location  $(x^c, y^c)_{(a,b)} \in [-\bar{x}, \bar{x}] \times [-\bar{y}, \bar{y}]$ . Note that due to discretization, the backward mapping (3) may lead to an error in reconstructing the actual position  $(x_p, y_p)$ . The Euclidean norm of this error is upper bounded by  $\sqrt{2} \cdot c/2$ . This problem is treated by adding suitable safety margins in constructing the collision avoidance constraints as will be shown in the following subsections.

#### A. Grid Generation

As our kinematic model (1) represents a sampled continuous time model, we have to ensure that a cell cannot be skipped during one time step. To derive a lower bound on  $c$ , we assume that the dynamics (1) is Lipschitz continuous in the spatial coordinates  $x_p$  and  $y_p$  with respect to its second argument  $u_p$  — uniform in  $p$ , i.e. there exists  $L > 0$  such

that

$$\left| (f(z_p, u_p) - f(z_p, \hat{u}_p))_j \right| \leq L \|u_p - \hat{u}_p\|_\infty \quad \forall z_p \in \mathcal{Z}$$

holds for  $j \in \{1, 2\}$  and all  $u_p, \hat{u}_p \in U$ . Then, to avoid skipping cells, we have to ensure that  $\|q(z_p) - q(f(z_p, u_p))\|_\infty \leq 1$  holds for all  $u_p \in U$ . Due to the quantization (3), the latter inequality considers the differences in the spatial variables  $x_p$  and  $y_p$  only. Hence, inequality

$$\begin{aligned} \left| (z_p - f(z_p, \hat{u}_p))_j \right| &= \left| (f(z_p, 0) - f(z_p, \hat{u}_p))_j \right| \\ &\leq L \|\hat{u}_p\|_\infty =: \underline{c} \leq c \end{aligned}$$

is sufficient. Note that due to the structure of our model (1), we can drop the angular speed and set  $\underline{c} := L \cdot \max\{|\underline{v}|, |\bar{v}|\}$ . The Lipschitz condition holds, e.g. for control affine systems with continuous vector fields and a compact state space.

#### B. Safety Margins and Constraint Construction

Based on the actual cell size  $c \geq \underline{c}$ , in the spatial set  $[-\bar{x}, \bar{x}] \times [-\bar{y}, \bar{y}]$ , first, we aim to guarantee a minimum distance  $d_{\min}$  between each pair of robots regarding their physical dimensions. Second, we aim also to avoid overlapping of predicted trajectories, see Fig. 1 (left) for an illustration. From Fig. 1 (left) and our definition of  $\underline{c}$ , we obtain that overlapping can only occur if the distance between two robots is less than  $\underline{c}$ , i.e. if both robots are in the vicinity of the same cell boundary. Hence, keeping two consecutive time instants in mind, minimum distance  $d_{\min}$  and avoidance of overlapping can be ensured if the condition

$$\left\| \begin{pmatrix} x_p \\ y_p \end{pmatrix} - \begin{pmatrix} x_q \\ y_q \end{pmatrix} \right\|_\infty \geq \max\{d_{\min}, \underline{c} + \varepsilon\} \quad (4)$$

with  $0 < \varepsilon \ll 1$  is satisfied for  $z_p, z_q$  with  $p, q \in [1:P], p \neq q$ . Due to the transmission of quantized position information, however, we have to use  $f_c(q(z_q))$  instead of  $(x_q, y_q)$  and, thus, obtain

$$\begin{aligned} &\left\| \begin{pmatrix} x_p \\ y_p \end{pmatrix} - \begin{pmatrix} x_q \\ y_q \end{pmatrix} \right\|_\infty \\ &\geq \left\| \begin{pmatrix} x_p \\ y_p \end{pmatrix} - f_c(q(z_q)) \right\|_\infty - \underbrace{\left\| f_c(q(z_q)) - \begin{pmatrix} x_q \\ y_q \end{pmatrix} \right\|_\infty}_{\leq c/2} \end{aligned}$$

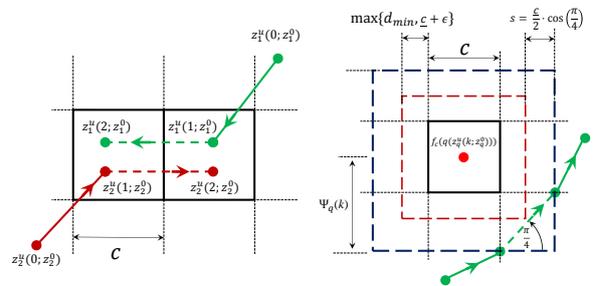


Fig. 1. Left: overlapping of prediction trajectories. Right: avoiding constraint violation due to discretization for the case  $q(z_q(k)) = q(z_q(k+1))$ .

using the triangular inequality. Hence, Condition (4) is satisfied if

$$\left\| \begin{pmatrix} x_p \\ y_p \end{pmatrix} - f_c(q(z_q)) \right\|_\infty \geq \underbrace{\max\{d_{\min}, \underline{c} + \varepsilon\}}_{=: d_{\text{quant}}} + \frac{c}{2} \quad (5)$$

holds for two consecutive time instants. Last, we have to cope with the intersample behavior of model (1), which may cause (5) to be violated in between two sampling instants  $n$  and  $n+1$ . Since no additional information regarding the continuous time dynamics is available, we assume that linear interpolation between  $z_p(n)$  and  $z_p(n+1)$  reveals the respective connecting path. To avoid violation of (5), we have to consider the cases  $q(z_q(k)) = q(z_q(k+1))$  and  $q(z_q(k)) \neq q(z_q(k+1))$ . In the latter case, robot  $p$  may move in the direction of cell  $q(z_q(k))$  causing a maximal violation  $s = \underline{c}$  of (5). In the first case, the maximal possible violation  $s$  is illustrated in Fig. 1 (right), which can be obtained by the rule of Pythagoras as  $s = \underline{c}/2 \cdot \cos(\pi/4)$ .

Now, the coupling constraints  $g_{q,k}^p$  of the  $p^{\text{th}}$  robot induced by the other robots  $q \in [1 : P] \setminus \{p\}$  can be formulated via

$$\Psi_q(k) - \left\| \begin{pmatrix} x_p(k) \\ y_p(k) \end{pmatrix} - f_c(q(z_q(k))) \right\|_\infty \quad (6)$$

for  $k \in [1 : N]$ , where

$$\Psi_q(k) = \begin{cases} d_{\text{quant}} + \underline{c}, & \text{if } q(z_q(k)) \neq q(z_q(k+1)) \\ d_{\text{quant}} + \frac{\underline{c}}{2} \cdot \cos\left(\frac{\pi}{4}\right), & \text{otherwise.} \end{cases}$$

Note that the differentiability property of (6) is not affected by possible switches in  $\Psi_q$  as these are fixed, but only depends on differentiability of model (1). To circumvent the nondifferentiability of the infinity norm in (6), we utilize squircles to approximate the cell geometry [14]. A squircle, which is a special case of a superellipsoid, is a geometric shape between a square and a circle. The polar equations of a squircle are given by

$$\begin{aligned} x(\phi) &= 2^{-3/4} h \cdot |\cos(\phi)|^{1/2} \cdot \text{sgn}(\cos(\phi)), \\ y(\phi) &= 2^{-3/4} h \cdot |\sin(\phi)|^{1/2} \cdot \text{sgn}(\sin(\phi)) \end{aligned}$$

with  $\phi \in [0, 2\pi)$ .  $h$  is the size of the square, which a squircle is approximating from the outside. Now, by setting  $h = 2 \cdot \Psi_q(k)$ , the coupling constraints  $g_{q,k}^p$  of the  $p^{\text{th}}$  robot can be formulated using the Euclidean norm as<sup>1</sup>

$$\underbrace{\frac{\Psi_q(k) \sqrt{|\cos(\beta)| + |\sin(\beta)|}}{2^{(1/4)}} - \left\| \begin{pmatrix} x_p(k) \\ y_p(k) \end{pmatrix} - f_c(q(z_q(k))) \right\|_2}_{=: g_{q,k}^p}$$

with  $k \in [1 : N]$  and  $q \in [1 : P] \setminus \{p\}$ . The angle  $\beta$  is calculated via

$$\beta = \arctan\left(\frac{y_p(k) - y_q^c(k)}{x_p(k) - x_q^c(k)}\right),$$

<sup>1</sup>The absolute function  $|\cdot|$  is approximated in simulations and experiments by the differentiable expression  $|x| \approx \sqrt{x^2 + \varepsilon}$ , for  $x \in \mathbb{R}$  and  $0 < \varepsilon \ll 1$ .

where  $(x_q^c(k), y_q^c(k))$  is obtained via the mapping (3) given the predictions  $\mathcal{I}_q(n)(k)$ .

Now, for each robot  $p$ , the coupling constraints  $g_{q,k}^p$ ,  $q \in [1 : P - 1]$  are assembled to

$$G(z_p^u(k; z_p^0), i_p(n)) := \left( g_{1,k}^p, \dots, g_{p-1,k}^p, g_{p+1,k}^p, \dots, g_{P,k}^p \right)$$

with  $k \in [1 : N]$ . Incorporating the latter in the respective OCP (2) enforces minimal distance between robots; collision avoidance; and overlapping avoidance of predicted trajectories at and in between sampling instants  $k \in [1 : N]$ .

#### IV. NUMERICAL SIMULATIONS

In this section, we explore the performance of Algorithm 1 through numerical simulations of a group of  $P = 4$  non-holonomic mobile robots. The state and control constraints are set as

$$Z := [-6, 6]^2 \times \mathbb{R}, \quad U := [-0.5, 0.5] \times [-0.5, 0.5].$$

The minimum distance to be ensured between the robots is chosen as  $d_{\min} = 0.5$  (m). Using  $L = 1$ , the minimum cell size is given by  $\underline{c} = 0.5$  (m), see Subsection III-A. We perform simulations with different cell sizes  $c \in \{0.5, 1, 1.5, 2\}$ . The initial conditions of the robots as well as their reference points can be found in Table I. Following [15], the running costs  $\ell_p$  are chosen as

$$\ell_p(z_p, u_p) := \left\| \begin{pmatrix} x_p - x_p^* \\ 5(y_p - y_p^*) \\ \theta_p - \theta_p^* \end{pmatrix} \right\|^2 + 0.2 \cdot \left\| \begin{pmatrix} v_p^2 \\ w_p^2 \end{pmatrix} \right\|^2$$

to ensure stabilization of non-holonomic robots without terminal constraints or costs. Moreover, closed-loop simulations are executed until all robots meet the condition

$$\|z_p(n) - z_p^*\| \leq 0.01. \quad (7)$$

Finally, the numerical safety margin  $\varepsilon = 10^{-6}$  is used. The total number of closed-loop iterations is denoted by  $n_\#$ .

TABLE I  
INITIAL CONDITIONS AND REFERENCE POINTS OF THE ROBOTS

Robot $i$	Initial condition $z_i^0$	Reference point $z_i^*$
1	$(4.5, 4.5, -3\pi/4)^\top$	$(-4.5, -4.5, \pi)^\top$
2	$(-4.5, 4.5, -\pi/4)^\top$	$(4.5, -4.5, 0)^\top$
3	$(4.5, -4.5, 3\pi/4)^\top$	$(-4.5, 4.5, \pi)^\top$
4	$(-4.5, -4.5, \pi/4)^\top$	$(4.5, 4.5, 0)^\top$

We investigate the performance of Algorithm 1 in terms of the number of communicated tuples as well as accumulated closed-loop costs. The number of communicated tuples is given by  $K := n_\# \cdot P \cdot N$ , i.e.  $K$  is the number of the communicated tuples among all robots over the closed-loop simulation. The accumulated closed-loop costs are given by

$$M := \sum_{n=0}^{n_\#} \sum_{p=1}^P \ell_p(z_p^{MPC}(n), u_p^{MPC}(n)),$$

where  $u_p^{MPC}(n)$  denotes the control signal applied in Step 5) of Algorithm 1 at time instant  $n$  and  $z_p^{MPC}$  is the corresponding closed-loop trajectory.

Simulations are performed using the derivative-based optimization (interior-point) method implemented in IPOPT [16] and coupled with MATLAB via CasADi toolbox [17]. Fig. 2 shows the required number of communicated tuples  $K$  and the accumulated costs  $M$ , respectively. The growth of these key figures is caused by robots, which must take detours to avoid collisions and reach their reference equilibria, cf. Fig. 3 for simulation snapshots. For increasing cell sizes also, the length of these detours as well as the simulation time increase, which leads to the observed increases in communication load and costs. We also observed that the minimal prediction horizon  $N$  required to meet condition (7) grows with increasing the cell size, cf. Table II.

The reason is that a coarser quantization (larger cell sizes) leads to worse position data on the other robots. As a result, the robots require a longer prediction horizon to observe a decrease to the reference point while circumventing occupied cells. This can also be seen from Fig. 4, which shows the evolution of the sum  $M_P(n) = \sum_{p=1}^P \ell_p(z_p^{MPC}(n), u_p^{MPC}(n))$  for each considered cell size (left) and depicts the develop-

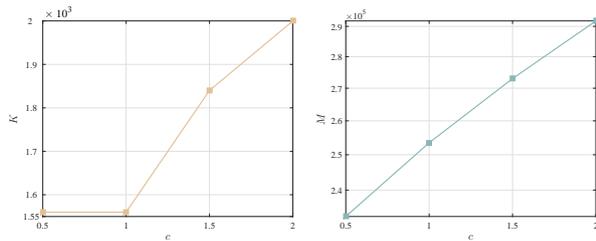


Fig. 2. Cell size  $c$  vs. number of communicated tuples  $K$  (left) and closed-loop costs  $M$  (right) for  $N = 9$  and four non-holonomic robots

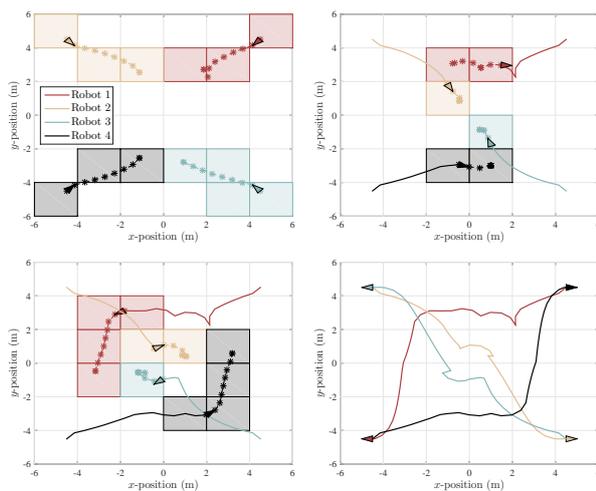


Fig. 3. Snap shots of closed-loop trajectories (continuous lines) for  $c=2$  and  $N=9$  taken at  $n=1$  (top left),  $n=10$  (top right),  $n=20$  (bottom left) and  $n=51$  (bottom right). Predictions are dashed lines with star markers.

TABLE II  
IMPACT OF CELL SIZE  $c$  ON MINIMAL  $N$  AND  $n_{\#}$

$c$	0.5	1	1.5	2
$N$	7	7	9	9
$n_{\#}$	40	45	47	51

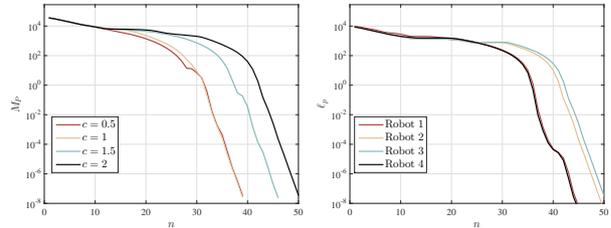


Fig. 4. Development of  $M_P$  with  $c = \{0.5, 1, 1.5, 2\}$  and  $N=9$  (left) and of  $\ell_p(z_p^{MPC}(n), u_p^{MPC}(n))$  for each robot with  $c=2$  and  $N=9$  (right).

ment of  $\ell_p(z_p^{MPC}(n), u_p^{MPC}(n))$  for each robot with  $c = 2$  (right). It can be observed from Fig. 4 (left) that the overall instantaneous costs  $M_P$  are decreasing faster for smaller cell sizes  $c$  in our simulations. Increases in the costs  $\ell_p$  occurs due to the necessity of detours for collision avoidance. The decrease in  $M_P$  can be interpreted as the overall system convergence for the chosen cell sizes. Hence, we have numerically shown that the proposed approach works.

## V. LABORATORY EXPERIMENTS

Algorithm 1 was validated experimentally using two Pioneer 3-AT<sup>®</sup> non-holonomic research platforms, see, e.g. [18], for the technical details of the used robots. In the experiments, the state and control constraints are set as

$$Z := [-3, 3]^2 \times \mathbb{R}, \quad U := [-0.125, 0.3] \times [-0.35, 0.35].$$

Based on the physical dimensions of the robots, the minimum distance to be ensured between the centers of the robots is  $d_{\min} = 0.5$  (m). Using  $L = 1$ , the minimum cell size is computed as  $\underline{c} = 0.3$  (m), see Subsection III-A. The localization of the robots was achieved using the motion capture system *OptiTrack*<sup>®</sup> with a setup of 4 cameras.

The conducted experiments adopted a remote host using a Matlab's TCP/IP interface, which is remotely communicating with *OptiTrack*'s system in order to acquire the localization data. Moreover, the same host was used to send the feedback control commands to the robots. Similar to the numerical simulations, Algorithm 1 was implemented on MATLAB using CasADi toolbox [17]. However, the running costs weights used in the experiments were set as

$$\ell_p(z_p, u_p) := \left\| \begin{pmatrix} (x_p - x_p^*)^2 \\ 5(y_p - y_p^*)^2 \\ (\theta_p - \theta_p^*)^2 \end{pmatrix} \right\|^2 + \left\| \begin{pmatrix} 2(v_p)^2 \\ 2\sqrt{2}(\omega_p)^2 \end{pmatrix} \right\|^2,$$

i.e. the control weights are chosen larger than the values used in the simulations in order to get a well damped closed-loop behavior of the robots and to compensate for the unmodeled dynamics of the used robots in the experiments.

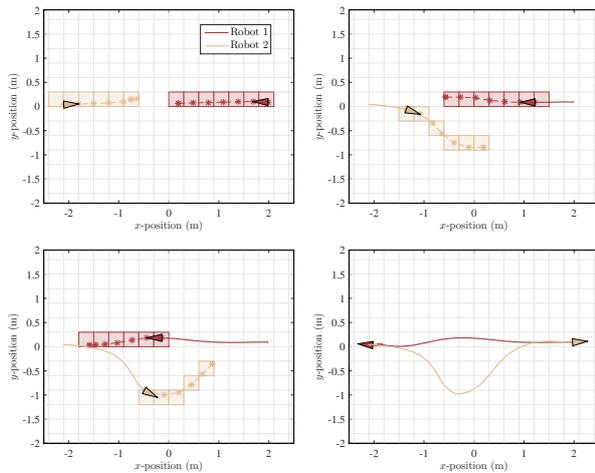


Fig. 5. Snap shots of closed-loop behavior in experiments for case 1 taken at  $n = 1$  (top left),  $n = 7$  (top right),  $n = 12$  (bottom left) and  $n = 36$  (bottom right).

The experiments were performed with cell size  $c = 0.3$  (m). Overall, four experiments were conducted. In all experiments, the two robots were commanded to exchange positions while in two of the experiments the robots were additionally commanded to also exchange orientation. The closed-loop experiments were executed until all robots met the condition  $\|z_p(n) - z_p^*\| \leq 0.1$ .

Based on a primary simulation with the parameters given above,  $N = 7$  was necessary for the convergence of Algorithm 1. Therefore, the same prediction horizon was adopted in all experiments. See the attached video for the laboratory experiments. Fig. 5 shows the closed-loop trajectories of case 1, where the robots start from two opposite sides of the spatial set  $[-\bar{x}, \bar{x}] \times [-\bar{y}, \bar{y}]$ . The robots are commanded to exchange positions while the reference orientations are equal to the initial ones. As it can be noticed, the proposed controller fulfilled the control task. The influence of the fixed priority rule appears in Fig. 5 as robot 1 takes a shorter path to its reference point than robot 2. The success of achieving the control objective was also observed for the other three cases. In summary, the applicability of the proposed controller was successfully tested.

## VI. CONCLUSION

In this paper, we proposed a distributed MPC scheme based on the communication of occupancy grid predictions instead of predicted trajectories for mobile robots. Changing the data representation of the communicated information leads to alleviated requirements on the communication means. Still, sensitivity information can be exploited as the optimization is performed in a continuous setting. We provided a possible formulation of the coupling constraints utilized by the proposed controller. This formulation is based on squircles and yielded satisfactory results both numerically and experimentally. Thus, a proof-of-concept of the proposed approach is demonstrated.

Future considerations of this study include the extension to multi-step MPC using the priority list approach from [19]. Additionally, parallel decision schemes or improvements via contract-based schemes shall be considered, see, e.g. [20]. Moreover, stability will be analyzed rigorously.

## REFERENCES

- [1] B. T. Stewart, S. J. Wright, and J. B. Rawlings, "Cooperative distributed model predictive control for nonlinear systems," *Journal of Process Control*, vol. 21, no. 5, pp. 698–704, 2011.
- [2] M. Farina and R. Scattolini, "An output feedback distributed predictive control algorithm," in *IEEE Conference on Decision and Control and European Control Conference*, December 2011, pp. 8139–8144.
- [3] M. W. Mehrez, G. K. I. Mann, and R. G. Gosine, "An optimization based approach for relative localization and relative tracking control in multi-robot systems," *Journal of Intelligent & Robotic Systems*, vol. 85, no. 2, pp. 385–408, 2017.
- [4] L. Grüne and J. Pannek, *Nonlinear Model Predictive Control: Theory and Algorithms*, 2nd ed., ser. Communications and Control Engineering. Springer International Publishing, 2017.
- [5] R. Scattolini, "Architectures for distributed and hierarchical model predictive control – a review," *Journal of Process Control*, vol. 19, no. 5, pp. 723–731, 2009.
- [6] W. B. Dunbar and R. M. Murray, "Distributed receding horizon control for multi-vehicle formation stabilization," *Automatica*, vol. 42, no. 4, pp. 549–558, 2006.
- [7] K. Hashimoto, S. Adachi, and D. V. Dimarogonas, "Distributed aperiodic model predictive control for multi-agent systems," *IET Control Theory Applications*, vol. 9, no. 1, pp. 10–20, 2015.
- [8] A. Venkat, J. Rawlings, and S. Wright, "Stability and optimality of distributed model predictive control," *Proceedings of the 44th IEEE Conference on Decision and Control*, pp. 6680–6685, 2005.
- [9] K. Worthmann, C. Kellett, P. Braun, L. Grüne, and S. Weller, "Distributed and decentralized control of residential energy systems incorporating battery storage," *IEEE Trans. Smart Grid*, vol. 6, no. 4, pp. 1914–1923, 2015.
- [10] A. Richards and J. P. How, "Robust distributed model predictive control," *International Journal of Control*, vol. 80, no. 9, pp. 1517–1531, 2007.
- [11] L. Grüne and K. Worthmann, *Distributed Decision Making and Control*. Springer Verlag, 2012, ch. A distributed NMPC scheme without stabilizing terminal constraints, pp. 261–287.
- [12] A. Richards and J. How, "A decentralized algorithm for robust constrained model predictive control," in *American Control Conference*, vol. 5, June 2004, pp. 4261–4266 vol.5.
- [13] A. Boccia, L. Grüne, and K. Worthmann, "Stability and feasibility of state constrained MPC without stabilizing terminal constraints," *Systems & Control Letters*, no. 72, pp. 14–21, 2014.
- [14] M. Fernández-Guasti, "Analytic geometry of some rectilinear figures," *Int. Jour. of Math. Ed. in Sci. & Tech.*, vol. 23, no. 6, pp. 895–901, 1992.
- [15] K. Worthmann, M. W. Mehrez, M. Zanon, G. K. I. Mann, R. G. Gosine, and M. Diehl, "Model predictive control of nonholonomic mobile robots without stabilizing constraints and costs," *IEEE Transactions on Control Systems Technology*, vol. 24, no. 4, pp. 1394–1406, 2016.
- [16] A. Wächter and T. L. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, no. 1, pp. 25–57, 2006.
- [17] J. Andersson, "A General-Purpose Software Framework for Dynamic Optimization," PhD thesis, Arenberg Doctoral School, KU Leuven, Department of Electrical Engineering (ESAT/SCD) and Optimization in Engineering Center, Kasteelpark Arenberg 10, 3001-Heverlee, Belgium, October 2013.
- [18] M. W. Mehrez, G. K. Mann, and R. G. Gosine, "Stabilizing NMPC of wheeled mobile robots using open-source real-time software," in *16th International Conference on Advanced Robotics (ICAR)*, 2013, pp. 1–6.
- [19] J. Pannek, "Parallelizing a State Exchange Strategy for Noncooperative Distributed NMPC," *Systems & Control Letters*, vol. 62, pp. 29–36, 2013.
- [20] S. Lucia, M. Kögel, and R. Findeisen, "Contract-based predictive control of distributed systems with plug and play capabilities," *IFAC-PapersOnLine*, vol. 48, no. 23, pp. 205–211, 2015.

This article is reprinted via the license agreement of IEEE for reprinting in thesis: ©2017 IEEE. Reprinted, with permission.

# Connection between Quantisation and Bandwidth Requirements of Distributed Model Predictive Control

Tobias Sprodowski \* Juliana K. Sagawa \*\* Jürgen Pannek \*\*\*

\* Faculty for Production Engineering, University of Bremen, Germany,  
(e-mail: spr@biba.uni-bremen.de)

\*\* Department of Production Engineering, Federal University of São Carlos, Brazil

\*\*\* BIBA, Bremer Institut für Produktion und Logistik, University of Bremen, Germany

---

**Abstract:** Many distributed systems rely on communication as a necessary condition to steer the overall system to a reference or target state, which may lead to a large bandwidth requirements. Here, we consider a Distributed Model Predictive Control Scheme (DMPC) where each agent predicts its own trajectory in every time step, which is then broadcasted among the agents. We aim to reduce the necessary communication and introduce the concept of prediction coherence as a degree of difference of two predictions in two successive time steps. We evaluate the influence of quantisation of communicated predicted states on the prediction coherence in a street traffic model for a quantised intersection in order to incorporate possible disturbances in communication. We numerically observe that prediction coherence reveals a bound for the minimal bandwidth requirements for such a distributed control setting.

*Keywords:* Multi-agent systems, Coordination of multiple vehicle systems, Control of constrained systems, Modelling for control optimisation

---

## 1. INTRODUCTION

In the last years, the control of autonomous vehicles developed to a large research field. While incorporating communication between the vehicles induces new problems and increases complexity, the main challenge in this setting is to reduce the communication requirements to comply with wireless bandwidth limitations.

Distributed Model Predictive Control Scheme (DMPC) is one control technique, which is applicable in distributed settings and allows for a direct incorporation of a performance index, constraints and nonlinear dynamics. It has been applied in many areas ranging from highly dynamic systems as in formation control of robots, see Farina et al. (2015); Zhang and Pathirana (2011), or slower reactive systems like chemical plants, cf. Rawlings and Stewart (2008); Venkat et al. (2005). Furthermore, the theoretical aspects regarding stability and suboptimality with and without terminal conditions were widely discussed, cf., e.g., Rawlings et al. (2017); Grüne and Worthmann (2012); Grüne and Pannek (2017).

Regarding communication, a strong coupling of the subsystems requires a reliable exchange of data to render such a system stable which increases the communication load with a higher number of subsystems. Several methods were proposed to reduce the communication effort, which can be structured in neighbourhood, active incorporating loss of communication, or quantisation methods.

Neighbourhoods are either defined as a fixed dependency graph, defining which subsystems are coupled in an upstream- and downstream graph, cf. Rawlings and Stewart (2008); Dunbar (2006), or are revised in every time instant depending on the current dependency among highly dynamic systems like mobile robots, see, e.g. Farina et al. (2015).

To react directly to loss of communicated data, one possibility is to utilise either predictions of already received packets Onat et al. (2011); Varutti et al. (2009); Grüne (2009) with globally synchronised time instants, imposing stability or terminal constraints (Wang and Ong (2010); Wang et al. (2015)), or calculate a worst-case state estimation (Hu and Ei-Farrat (2013)) based on a decreasing Lyapunov function.

Another approach for reducing communication is quantisation of data. El Mongi Ben Gaid and Çela (2006) evaluated the relation between the quantisation precision of communicated data and sampling step size. In each sampling step the quantisation precision was chosen based on a precalculated set given by the designer. Pu et al. (2015) developed a quantisation of states which is adopted in every iteration regarding the bitsize of the communication channel. Therefore, the quantisation operator and states were transmitted in every time instant. Liu et al. (2016) extended this to an event-triggered approach, where the agents follow a common set constraint. Zhu et al. (2016) applied an ADMM-scheme using an apriori chosen quantised setting for the exchange of the consensus

constraints among the agents. The final state region was defined as consensus region, which was the desired state for the overall system. By applying the quantisation on the state region, the choice of the quantisation level influenced the consensus region proportionally, leading to a faster convergence for the overall system with respect to a larger size of the consensus region. Chen et al. (2016) evaluated a scheme where a sensor network communicating with a central instance are utilizing an  $H_2/H_\infty$  estimator to incorporate unknown noise. To further reduce the communication load, they introduced a dual data compression strategy, which was based on a weighted random choice of the components of the sensors for transmitting data.

All of these models handle the disturbance in two ways: For one, stability constraints are imposed for each subsystem to keep them in a predefined state for the duration of the disturbances. Secondly, the amount of exchanged data is extended to compensate for future communication losses. Here, we do not adapt the system to incorporate disturbances, but examine the similarity of broadcasted predictions of one subsystem in two successive time instants.

We propose the concept of *prediction coherence* for quantised communication data. We utilise the intersection scenario from Sprodowski and Pannek (2015) and extend it to cover for street traffic rules. The spacial set is quantised to allow for transmitting the occupancy grid predictions defined in Mehrez et al. (2017). The idea of this approach is to locally optimise over a continuous state/control set while communication relies on integer values only. Within the concept of prediction coherence, we consider the difference between occupancy grid predictions for two successive time instants. Measuring the number of new or updated predicted states, we will derive that the cumulative prediction coherence represents a lower bound on the bandwidth requirements, which is crucial for VANET applications, Khaliq et al. (2016). Last, within our numerical experiments we examine the dependency of the prediction coherence on both the number of vehicles and the chosen quantisation. The results indicate by varying the quantisation that, there exists a minimal lower bound for the bandwidth requirements.

We first recap the quantised setting based on the occupancy grid and define the creation of the directional constraints in addition to the collision avoidance constraints in Section 2. In Section 3, we present the distributed MPC scheme and define prediction coherence. In Section 4, we investigate this concept via numerical simulations before drawing conclusions in Section 5.

## 2. PROBLEM SETTING

We consider an intersection with lanes in both directions where the spacial set is quantised, cf. Fig.1 for a schematic sketch. To match the latter to our dynamics introduced next, we recap the definition of the occupancy grid and the necessary quantisation steps. Then, we explore the composition of directional and collision avoidance constraints. Combining both dynamics and constraints and introducing a performance index we derive an Optimal Control Problem (OCP), which we will utilise in a Distributed

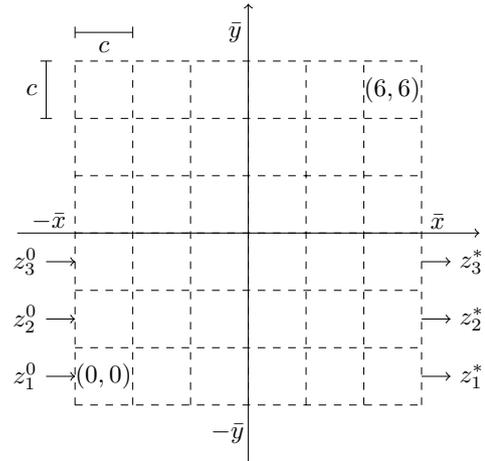


Fig. 1. Illustration of the intersection spacial set  $\mathcal{G}$  with  $a_{\max} = b_{\max} = 6$  and vehicles start ( $z_i^0$ ) and reference states ( $z_i^*$ ).

Model Predictive Control (DMPC) scheme relying on a quantised data exchange.

### 2.1 Dynamics and Quantisation

Each vehicle in the intersection is modeled as a discrete time control system

$$z_p(n+1) = f(z_p(n), u_p(n)) \quad (1)$$

which may be specified by a kinematic model or a system dynamics, where  $z_p \in Z \subset \mathbb{R}^2$  and  $f: \mathbb{R}^2 \times \mathbb{R}^2 \rightarrow \mathbb{R}^2$  is continuous. Here,  $n$  denotes the current time step whereas the state  $z_p = (x_p, y_p)^\top$  contains the spatial coordinates. The control  $u_p \in U \subset \mathbb{R}^2$  is given by  $u_p = (v_p^x, v_p^y)^\top$ , denoting the speed in both directions. State and controls are bounded by

$$Z := [-\bar{x}, \bar{x}] \times [-\bar{y}, \bar{y}], \quad U := [-\bar{v}^x, \bar{v}^x] \times [-\bar{v}^y, \bar{v}^y]$$

for  $\bar{x}, \bar{y}, \bar{v}^x, \bar{v}^y > 0$ .

Given a control sequence

$$u_p = (u_p(0), u_p(1), \dots, u_p(N-1)),$$

we can define the state trajectory utilising (1)

$$z_p^u(\cdot; z_p^0) := (z_p^u(0; z_p^0), z_p^u(1; z_p^0), \dots, z_p^u(N; z_p^0)) \quad (2)$$

for the initial value  $z_p^0$  over the prediction horizon  $N$ . As illustrated in Fig. 1, the spacial set  $[-\bar{x}, \bar{x}] \times [-\bar{y}, \bar{y}]$  is partitioned to create the grid based on Mehrez et al. (2017). It consists of squared cells with cell width  $c$  such that  $c$  satisfies

$$2\bar{x} = a_{\max}c \quad \text{and} \quad 2\bar{y} = b_{\max}c$$

with  $a_{\max}, b_{\max} \in \mathbb{N}$ . The cells are indexed by  $(a, b) \in \mathcal{G}$  where  $\mathcal{G} := [0 : a_{\max} - 1] \times [0 : b_{\max} - 1] \subset \mathbb{N}_0^2$ . The state of a vehicle  $z_p$  can be mapped from the spacial set  $Z$  to the discrete set  $\mathcal{G}$  via the quantisation function  $q: Z \rightarrow \mathcal{G}$

$$q(z_p) = (a_p, b_p) := \left( \left\lfloor \frac{x_p + \bar{x}}{c} \right\rfloor, \left\lfloor \frac{y_p + \bar{y}}{c} \right\rfloor \right). \quad (3)$$

This quantisation is used to map the continuous predicted state trajectory (2) to the discrete occupancy grid

$$\begin{aligned} \mathcal{I}_p(n) &:= (n+k, q(z_p^u(k; z_p^0)))_{k \in [0:N]} \\ &:= (n+k, a_p^u(k; z_p^0), b_p^u(k; z_p^0))_{k \in [0:N]}, \end{aligned}$$

where  $(a_p^u(k; z_p^0), b_p^u(k; z_p^0))$  represents the quantised state of the  $p^{\text{th}}$  vehicle. Each vehicle broadcasts its occupancy grid prediction  $\mathcal{I}_p(n) \in (\mathbb{N}_0 \times \mathcal{G})^{N+1}$  and receives tuples from the other vehicles. The collected data is assembled in

$$i_p(n) := (\mathcal{I}_1(n), \dots, \mathcal{I}_{p-1}(n), \mathcal{I}_{p+1}(n), \dots, \mathcal{I}_P(n)), \quad (4)$$

and will be employed to ensure collision avoidance.

## 2.2 Formulation of the Constraints

We impose two different types of constraints for this intersection scenario. The first one enforces collision avoidance via infinity norm constraints originate from the geometry of the quantisation. The second type represents directional constraints, which depend on the heading of the vehicles to ensure avoidance of opposite lanes.

To define the collision avoidance constraints  $G_C$ , we can utilise the state trajectory of the  $p^{\text{th}}$  vehicle with  $z_p(k) \in Z$  and the transmitted quantised state trajectory of the  $q^{\text{th}}$  vehicle with  $\mathcal{I}_q(n)(k) \in \mathcal{G}$ . Here, the latter represents the  $k^{\text{th}}$  tuple in the received occupancy grid prediction. To utilise the discrete values in a continuous space, we introduce the backward mapping  $f_c : \mathbb{N}_0 \times \mathcal{G} \rightarrow \mathbb{R}^2$

$$f_c((n, a, b)) = \underbrace{(c \cdot (a + 0.5) - \bar{x}, c \cdot (b + 0.5) - \bar{y})^\top}_{=: (x^c, y^c)_{(a,b)}}, \quad (5)$$

which represents the conversion from the discrete set to the spacial set. Now, we can utilise the geometry of the quantisation to define the function  $g_{q,k}^p = g_{q,k}^p(z_p^u(k; z_p^0), \mathcal{I}_q(n)(k))$  using the infinity norm via

$$g_{q,k}^p := - \left\| \begin{pmatrix} x_p(k) \\ y_p(k) \end{pmatrix} + f_c(\mathcal{I}_q(n)(k)) \right\|_\infty + \frac{\Psi}{2} \leq 0 \quad (6)$$

for all  $k \in [1 : N]$ ,  $q \in [1 : P] \setminus \{p\}$  where  $\Psi \in \mathbb{R}$  defines a minimum distance to avoid overlapping of vehicles. Considering the collision avoidance expression  $g_{q,k}^p$ ,  $q \in [1 : P - 1]$  of vehicle  $p$  in relation to vehicle  $q \in [1 : P] \setminus \{p\}$ , this is assembled to

$$G_C(z_p^u(k; z_p^0), i_p(n)) = (g_{1,k}^p, \dots, g_{p-1,k}^p, g_{p+1,k}^p, \dots, g_{P,k}^p)$$

for all  $k \in [1 : N]$ .

The directional constraints are constructed in dependence of the start position  $z_p^0$  for each vehicle  $p$ . The main tool to ensure adherence of lanes in dependency of the entry point of a vehicle is a lock function  $f_l : \mathbb{R}^2 \rightarrow 2^{\mathcal{G}}$ , which basically rules out subsets of cells of the quantised intersection, here the opposite lanes. In the simplest case of an intersection of two streets with one lane in each direction, the quantised intersection consists of four elements, each representing an entry point into the intersection. The lock function can now be used, e.g., to exclude U-turns by locking the next clockwise element, cf. Fig. 2 for an illustration. More generally, we can utilise lock functions to impose directional constraints via

$$G_D(z, z_p^0) := - \min_{z^c \in f_l(z_p^0)} \|z - z^c\|_\infty + \frac{\Psi}{2} \leq 0,$$

where  $z$  is the current state,  $z^c$  the centre of the excluded cell, and  $\Psi$  denotes a safety margin.

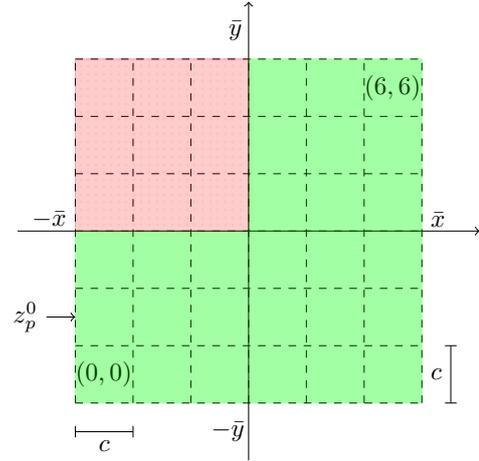


Fig. 2. Illustration of the lock function  $f_l$  for a discretised set  $\mathcal{G}$  with  $a_{\max} = b_{\max} = 3$ . The red dotted cells ( $z^c$ ) represent locked and the green cells unlocked areas (opposite lanes).

## 2.3 Optimal Control Problem and DMPC

Having defined the discrete control system and constraints, we now introduce a performance indicator function. This allows us to contemplate our setting to an optimal control problem. Here, we utilise the stage cost function  $\ell_p : Z \times U \rightarrow \mathbb{R}_{\geq 0}$ , which we assume to be positive definite with respect to a given target  $z_p^*$ . A typically definition is given by the Euclidean distance  $\ell_p(z_p(n), u_p(n)) := \|z_p(n) - z_p^*\|_2$ .

Based on the stage cost function, we can now define a finite horizon optimal control problem (OCP) which computes

$$\underset{u_p}{\operatorname{argmin}} J_p^N(u_p; z_p^0, i_p(n)) := \sum_{k=0}^{N-1} \ell_p(z_p^u(k; z_p^0), u_p(k)) \quad (7)$$

subject to the constraints

$$\begin{aligned} z_p^u(k+1; z_p^0) &= f(z_p(k; z_p^0), u_p(k)), & k \in [0 : N-1], \\ G_C(z_p^u(k; z_p^0), i_p(n)) &\leq 0, & k \in [1 : N], \\ G_D(z_p^u(k; z_p^0)) &\leq 0, & k \in [1 : N], \\ z_p^u(k; z_p^0) &\in Z, & k \in [1 : N], \\ u_p(k) &\in U, & k \in [0 : N-1]. \end{aligned}$$

By solving problem (7), we obtain an optimal control sequence

$$u_p^* = (u_p^*(0), \dots, u_p^*(N-1)),$$

which minimises  $\ell_p(z_p, u_p)$  and adheres the underlying system and the constraints. Here, we assume such a minimiser to exist in order to avoid technical difficulties, see Grüne and Pannek (2017) for details.

To obtain a feedback, which is applicable in a distributed setting, we incorporate the optimal control problem (7) in a distributed model predictive control (DMPC) setting. Apart from solving (7), the DMPC approach outlined in Algorithm 1 only requires two additional steps: To evaluate the current state of the system and to implement only a fraction of the computed open loop control before

restarting the procedure. For further details regarding

---

**Algorithm 1** DMPC Scheme

---

- (1) Measure current state  $z_p^0 := z_p(n)$  and collect  $i_p(n)$  for all  $p \in [1 : P]$
  - (2) For  $p = 1 : P$   
    Compute a minimiser  $u_p^*$  of OCP (7) and broadcast  $\mathcal{I}_p(n)$
  - (3) Implement  $u_p^*(0)$  for all  $p \in [1 : P]$ , increment  $n$  and goto Step (1).
- 

DMPC we refer to Grüne and Pannek (2017); Rawlings et al. (2017).

To simplify our setting, we follow the approach of Richards and How (2004, 2007) and consider a sequential order of vehicles. In this case, a bound on the maximal length of the trajectory is known from Sprodowski and Pannek (2015). Furthermore, sufficient conditions for stability and recursive feasibility are known from Grüne and Pannek (2017); Grüne and Worthmann (2012); Pannek (2013). These conditions exclude deadlocks and blockings of vehicles, but are outside the scope of this article.

### 3. PREDICTION COHERENCE

Utilizing quantised communication data allows us to consider incremental updates of data required to define the optimal control problem (7) within our controller. To derive bandwidth requirements for these incremental updates, we define the prediction coherence as a measure for the difference between two successive predicted state trajectories of a vehicle  $p$ . To simplify our notation, we abbreviate the prediction of vehicle  $p$  for timestep  $n - 1$  and  $n$  via

$$z_p^u(\cdot) := z_p^u(\cdot; z_p(n)) \quad \text{and} \quad \tilde{z}_p^u(\cdot) := z_p^u(\cdot; z_p(n-1)).$$

Then, the continuous prediction coherence  $r : Z^{N+1} \times Z^{N+1} \rightarrow \mathbb{R}_{\geq 0}$  is given by

$$r(z_p^u(\cdot), \tilde{z}_p^u(\cdot)) = \sum_{k=0}^{N-1} \|z_p^u(k) - \tilde{z}_p^u(k)\|_2.$$

Utilizing the quantisation to communicate the occupancy grid, we can accordingly define the quantised prediction coherence  $r_q : (\mathcal{G})^{N+1} \times (\mathcal{G})^{N+1} \rightarrow \mathbb{N}_0$  as

$$r_q(\mathcal{I}_p(n), \mathcal{I}_p(n-1)) = r(q(z_p^u(\cdot)), q(\tilde{z}_p^u(\cdot))) \quad (8)$$

for the difference of two successive occupancy grids of a vehicle  $p$  in the discrete set  $\mathcal{G}$ .

Assembling the quantised prediction coherence values of all vehicles we call

$$r_c(n) := \sum_{p=1}^P r_q(\mathcal{I}_p(n), \mathcal{I}_p(n-1))$$

the cumulated prediction coherence. Given the occupancy grids  $i_p(n-1)$ , the latter can be interpreted as the required incremental updates of the discrete occupancy grids  $i_p(n)$  to define the optimal control problem (7) at time instant  $n$  within Algorithm 1. Hence, if computed rigorously, it may serve as a lower bound on the bandwidth requirement for the control scheme in such a scenario.

### 4. NUMERICAL EXPERIMENTS

Within our simulations, we consider holonomic vehicles at an intersection  $Z := [-6, 6]^2$  with two lanes in both

directions. We suppose that each lane is oriented as in right hand traffic and contains one entry and one exit point corresponding to initial and target states. Here, we limit the setting to the intersection itself and exclude subsequent or prior road segments. This allows us to examine the dependence of prediction coherence on the quantisation, i.e. when the vehicles have to take detours. Ensuing unified entries into the intersection, we assume the initial speed for newly arriving vehicles with  $v^{\bar{x}}, v^{\bar{y}} = 0$  (m/s). The kinematic model for each vehicle is defined by

$$\begin{pmatrix} x_p \\ y_p \end{pmatrix}^+ = \begin{pmatrix} x_p \\ y_p \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \end{pmatrix} v_p^x + \begin{pmatrix} 0 \\ 1 \end{pmatrix} v_p^y \quad (9)$$

whereas the control constraints are defined via  $v^{\bar{x}}, v^{\bar{y}} = -v^x, -v^y := 0.5$  (m/s). Within the prediction horizon, we suppose the stage cost function

$$\ell_p(z_p, u_p) := \left\| \begin{pmatrix} x_p - x_p^* \\ y_p - y_p^* \end{pmatrix} \right\|^2 + 0.2 \|u_p\|^2$$

to be minimised. Last, we suppose the entry and exit point of the intersection to be given by Table 1 where we utilised the abbreviation  $s := \lfloor \frac{i}{4} \rfloor$ . As we are simulating with a number of vehicles with multiplier of four, the index  $i$  is utilised as a parameter for initial and exit points.

Table 1. Entry and exit point of the scenario

System ( $i$ )	Entry ( $z_i^0$ )	Exit ( $z_i^*$ )
$i \bmod 4 == 0$	$(-\bar{x} + 0.5, -\bar{y} + 0.5 + s)^\top$	$(\bar{x} - 0.5 - s, -\bar{y} + 0.5)^\top$
$i \bmod 4 == 1$	$(-\bar{x} - 0.5 - s, \bar{y} + 0.5)^\top$	$(\bar{x} - 0.5 - s, \bar{y} - 0.5)^\top$
$i \bmod 4 == 2$	$(\bar{x} - 0.5, \bar{y} - 0.5 - s)^\top$	$(-\bar{x} + 0.5, \bar{y} - 0.5 - s)^\top$
$i \bmod 4 == 3$	$(-\bar{x} + 0.5 + s, \bar{y} - 0.5)^\top$	$(-\bar{x} + 0.5 + s, \bar{y} + 0.5)^\top$

The optimal control problem for each vehicle is solved in the fixed order of entry of vehicles into the intersection. As the infinity norm is utilised for the constraint construction, we use the derivative-free optimisation method COBYLA (Constrained Optimisation by Linear Approximations, Powell (1998)) in the NLOpt package (Johnson (2004)) and implemented the entire scenario in C++. The closed-loop simulations are executed until the condition

$$\|z_p(n) - z_p^*\| < 0.01$$

is met. We conducted simulations with four and eight vehicles, sampling time of  $T = 0.5$  (s) and prediction horizon length of  $N = 4$ .

Considering  $P = 4$  vehicles, the quantised prediction coherence based on (8) are illustrated in Fig. 3 for cell sizes  $c = \{0.5, 2.0\}$  and the resulting closed loop stage costs are depicted in Fig. 4. We observe that the quantised prediction coherence  $r_q$  decreases as the cell size is increased. This relation is due to the geometry of the problem, i.e. vehicles require more time to traverse a cell and therefore the probability of a repeated transmission of a cell index is rising. However, as larger cell sizes induce longer detours, also higher times-to-target will occur. Here, with four vehicles the increase of time is small as the influence of the increased cell size is too little with respect to the number of vehicles. This can be observed in Fig. 4 showing a slightly longer time-to-target for  $c = 2.0$ .

For the setting with  $P = 8$  vehicles, the quantised prediction coherence is depicted in Fig. 5 and the resulting

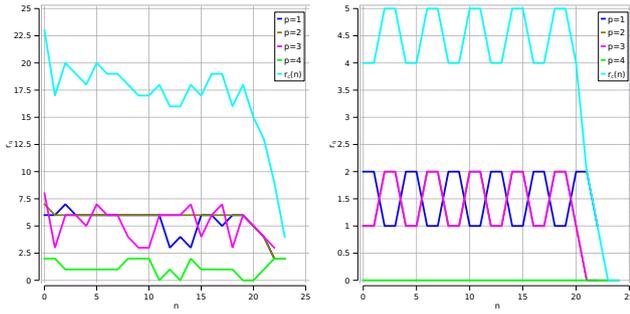


Fig. 3. Quantised prediction coherence  $r_q$  for  $P = 4$  vehicles and horizon  $N = 4$  with cell size  $c = 0.5$  (left),  $c = 2.0$  (right)

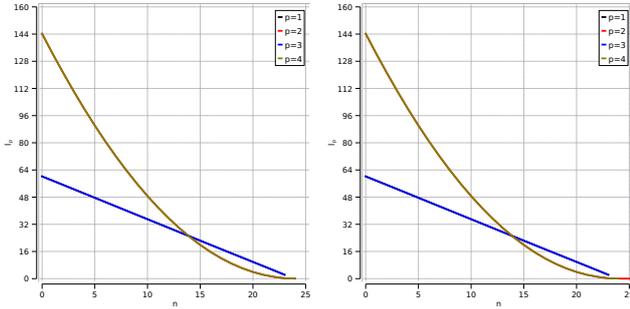


Fig. 4. Closed loop stage costs  $\ell(z_p)$  for  $P = 4$  vehicles and horizon  $N = 4$  with cell size  $c = 0.5$  (left),  $c = 2.0$  (right)

closed-loop stage costs in Fig. 6. From Fig. 5, we also

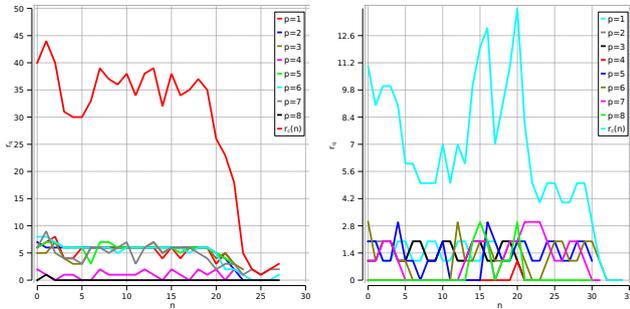


Fig. 5. Quantised prediction coherence  $r_q$  for  $P = 8$  vehicles and horizon  $N = 4$  with cell size  $c = 0.5$  (left),  $c = 2.0$  (right)

observe that the quantised prediction coherence decreases for larger cell sizes. As the impact of a higher number of vehicles results in a drastic increase in the constraints, the decrease in  $r_q$  is much stronger than for  $P = 4$  vehicles. This is also reflected in detours and time-to-target, cf. Fig. 6, where the impact of the necessary detours affects closed-loop stage costs more than with four vehicles, cf. Fig. 4.

In order to highlight properties of the intended bandwidth bound induced by the cumulated quantised prediction coherence, recall that from Figs. 3 and 5, we observed a dependency of  $r_q$  on both number of vehicles and cell size. Considering the cumulated quantised prediction

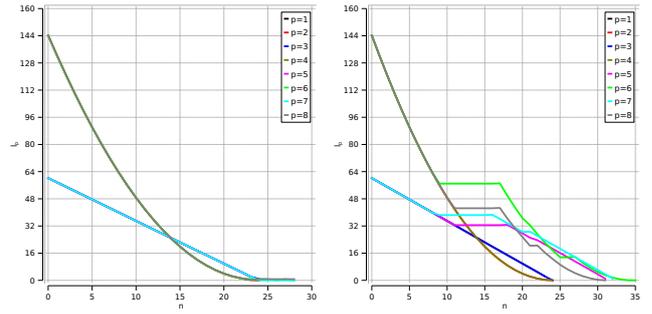


Fig. 6. Closed loop stage costs  $\ell(z_p)$  for  $P = 8$  vehicles and horizon  $N = 4$  with cell size  $c = 0.5$  (left),  $c = 2.0$  (right)

coherence for  $P = 4$  and  $P = 8$  vehicles, Fig. 7 shows the respective development for cell sizes  $c \in \{0.5, 1, 1.5, 2\}$ . From Fig. 7, we observe that the reduction rate of the

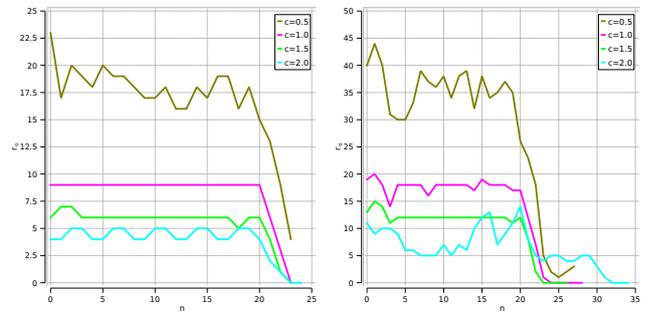


Fig. 7. Cumulated quantised predictions coherence  $r_q$  for  $P = 4$  (left) and  $P = 8$  (right) vehicles

cumulated quantised prediction coherence for increased cell sizes is not decreasing proportionally for larger cell sizes. Additionally, our experiments for  $P = 8$  show that the maximum cumulated quantised prediction coherence for  $c = 2$  is larger than for  $c = 1.5$  (for  $n = 20$ ). This indicates that there exist a lower bound to the cumulated quantised prediction coherence and thereby also to the required bandwidth. In practical terms, even with large cell sizes, a sufficient amount of information exchange among the agents is necessary.

## 5. CONCLUSION

In this paper, we utilised a prediction coherence function to examine the variation of the open loop solutions in successive time steps. The comparison of the prediction coherence for quantised states allowed us conclude that the cumulated prediction coherence decreases for increased cell sizes and appears to be lower bounded. Computed rigorously, the latter property may reveal a bound for the required bandwidth of a DMPC scheme in the considered intersection scenario, which is subject to future research. Furthermore, an analytical examination of coherence properties in a general nonlinear setting and of the requirements for convergence of the overall system will be done. Practical aspects will utilise larger test scenarios, adaptive quantisation, and further reduction of communication by introducing neighbourhood sets.

REFERENCES

- Chen, B., Hu, G., Zhang, W.a., and Yu, L. (2016). Distributed Mixed H2/H Fusion Estimation With Limited Communication Capacity. *IEEE Transactions on Automatic Control*, 61(3), 805–810. doi:10.1109/TAC.2015.2450271.
- Dunbar, W.B. (2006). Distributed receding horizon control of coupled nonlinear oscillators: Theory and application. In *Proceedings of the IEEE Conference on Decision and Control (CDC)*, 4854–4860. San Diego, CA, USA. doi:10.1109/Cdc.2006.376959.
- El Mongi Ben Gaid, M. and Çela, A. (2006). Trading Quantization Precision for Sampling Rates in Networked Systems with Limited Communication. In *Proceedings of the 45th IEEE Conference on Decision and Control*, 1135–1140. IEEE, San Diego, CA, USA. doi:10.1109/CDC.2006.377761.
- Farina, M., Perizzato, A., and Scattolini, R. (2015). Application of distributed predictive control to motion and coordination problems for unicycle autonomous robots. *Robotics and Autonomous Systems*, 72, 248–260. doi:10.1016/j.robot.2015.06.007.
- Grüne, L. and Pannek, J. (2017). *Nonlinear Model Predictive Control: Theory and Algorithms*. Communications and Control Engineering. Springer, London. doi:10.1007/978-3-319-46024-6.
- Grüne, L. and Worthmann, K. (2012). A distributed NMPC scheme without stabilizing terminal constraints. In *Distributed Decision Making and Control*, 261–287. Springer, London. doi:10.1007/978-1-4471-2265-4\_12.
- Grüne, L. (2009). Analysis and design of unconstrained nonlinear mpc schemes for finite and infinite dimensional systems. *SIAM Journal on Control and Optimization*, 48(2), 1206–1228. doi:10.1137/070707853.
- Hu, Y. and Ei-Farrat, N.H. (2013). Quasi-Decentralized Output Feedback Model Predictive Control of Networked Process Systems with Forecast-Triggered Communication. In *Proceedings of the American Control Conference (ACC)*, 2612–2617. Washington, DC, USA.
- Johnson, S.G. (2004). The NLOpt nonlinear-optimization package.
- Khaliq, K.A., Qayyum, A., and Pannek, J. (2016). Synergies of Advanced Technologies and Role of VANET in Logistics and Transportation. *International Journal of Advanced Computer Science and Applications(IJACSA)*, 7(11), 359–369. doi:10.14569/IJACSA.2016.071148.
- Liu, S., Xie, L., and Quevedo, D.E. (2016). Event-triggered Quantized Communication Based Distributed Convex Optimization. *IEEE Transactions on Control of Network Systems*, 5870(99), 1–11. doi:10.1109/TCNS.2016.2585305.
- Mehrez, M.W., Sprodownski, T., Worthmann, K., Mann, G.K.I., Gosine, R.G., Sagawa, J.K., and Pannek, J. (2017). Occupancy Grid based Distributed Model Predictive Control of Mobile Robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 4842–4847. IEEE, Vancouver, Canada. doi:10.1109/IROS.2017.8206360.
- Onat, A., Naskali, T., Parlakay, E., and Mutluer, O. (2011). Control over imperfect networks: Model-based predictive networked control systems. *IEEE Transactions on Industrial Electronics*, 58(3), 905–913. doi:10.1109/TIE.2010.2051932.
- Pannek, J. (2013). Parallelizing a state exchange strategy for noncooperative distributed NMPC. *Systems and Control Letters*, 62(1), 29–36. doi:10.1016/j.sysconle.2012.10.015.
- Powell, M.J.D. (1998). Direct search algorithms for optimization calculations. *Acta Numerica*, 7, 287–336.
- Pu, Y., Zeilinger, M.N., and Jones, C.N. (2015). Quantization design for distributed optimization with time-varying parameters. In *Proceedings of the 54th IEEE Conference on Decision and Control*, 2037–2042. IEEE, Osaka, Japan. doi:10.1109/CDC.2015.7402506.
- Rawlings, J.B., Mayne, D.Q., and Diehl, M. (2017). *Model Predictive Control: Theory and Design*. Nob Hill Publishing, 2nd edition.
- Rawlings, J.B. and Stewart, B.T. (2008). Coordinating multiple optimization-based controllers: New opportunities and challenges. *Journal of Process Control*, 18(9), 839–845. doi:10.1016/j.jprocont.2008.06.005.
- Richards, A. and How, J. (2004). A decentralized algorithm for robust constrained model predictive control. In *Proceedings of the American Control Conference*, 4261–4266. IEEE, Boston, MA, USA. doi:10.23919/ACC.2004.1383977.
- Richards, A. and How, J.P. (2007). Robust distributed model predictive control. *International Journal of Control*, 80(9), 1517–1531. doi:10.1080/00207170701491070.
- Sprodownski, T. and Pannek, J. (2015). Stability of distributed MPC in an intersection scenario. *Journal of Physics: Conference Series*, 659(1), 12049. doi:10.1088/1742-6596/659/1/012049.
- Varutti, P., Kern, B., Faulwasser, T., and Findeisen, R. (2009). Event-based model predictive control for Networked Control Systems. In *Proceedings of the 48th IEEE Conference on Decision and Control*, 567–572. Shanghai, China. doi:10.1109/CDC.2009.5400921.
- Venkat, A., Rawlings, J., and Wright, S. (2005). Stability and optimality of distributed model predictive control. In *Proceedings of the 44th IEEE Conference on Decision and Control*, 6680–6685. IEEE, Seville, Spain. doi:10.1109/CDC.2005.1583235.
- Wang, C. and Ong, C.J. (2010). Distributed model predictive control of dynamically decoupled systems with coupled cost. *Automatica*, 46(12), 2053–2058. doi:10.1016/j.automatica.2010.09.002.
- Wang, Q., Zou, Y., and Niu, Y. (2015). Event-triggered Model Predictive Control for Wireless Networked Control Systems with Packet Losses. In *5th Annual IEEE International Conference on Cyber Technology in Automation, Control and Intelligent Systems*, 1281–1286. Shenyang, China. doi:10.1109/CYBER.2015.7288128.
- Zhang, H. and Pathirana, P.N. (2011). Formation control of weak autonomous robots. In *50th IEEE Conference on Decision and Control (CDC) and European Control Conference (ECC)*, 4220–4225. IEEE, Orlando. doi:10.1109/CDC.2011.6160878.
- Zhu, S., Hong, M., and Chen, B. (2016). Quantized consensus ADMM for multi-agent distributed optimization. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 4134–4138. Shanghai, China. doi:10.1109/ICASSP.2016.7472455.

This article is printed under terms of Creative Commons CC-BY-NC-ND  
<https://creativecommons.org/licenses/by-nc-nd/4.0/> with all rights ©IFAC.  
Original publication is available via <https://doi.org/10.1016/j.ifacol.2017.08.1666>.

## Differential Communication with Distributed MPC based on Occupancy Grid <sup>☆</sup>

Tobias Sprodowski<sup>a,\*</sup>, Mohamed W. Mehrez<sup>c</sup>, Karl Worthmann<sup>d</sup>, George K.I. Mann<sup>c</sup>, Raymond G. Gosine<sup>c</sup>, Juliana K. Sagawa<sup>b,e</sup>, Jürgen Pannek<sup>a,b</sup>

<sup>a</sup>*University of Bremen, Department of Production Engineering, Hochschulring 20, 28359 Bremen, Germany*

<sup>b</sup>*BIBA - Bremer Institut für Produktion und Logistik GmbH, Hochschulring 20, 28359 Bremen*

<sup>c</sup>*Intelligent Systems Lab, Memorial University of Newfoundland, St. John's, NL, Canada*

<sup>d</sup>*Institute for Mathematics, Technische Universität Ilmenau, Ilmenau, Germany*

<sup>e</sup>*Department of Production Engineering, Federal University of São Carlos, Brazil*

---

### Abstract

We introduce a novel Distributed Model Predictive Control (DMPC) algorithm, which is based on projecting predicted trajectories on a quantised spatial set to reduce the communication load. The scheme exploits advantages of continuous optimisation methods while only quantised data is broadcasted. Further, we set up a differential communication scheme, in which only altered cells instead of the full prediction are broadcasted. While the quantisation reduces the communication effort for the overall system, the differential communication further reduces the effort depending on the chosen cell size. The approach is evaluated in simulations using groups of holonomic and non-holonomic mobile robots.

*Keywords:* Distributed Model Predictive Control (DMPC), Non-holonomic and holonomic robots, Multi-agent system, Quantisation, Collision avoidance

---

### 1. Introduction

Control problems of mobile robots have gained a considerable interest in the robotics community. This development is driven by technological advances, e.g., in servo engines and computationally-powerful processors, which led to a new generation of autonomous platforms capable of conducting sensitive tasks [1].

---

<sup>☆</sup>This work is supported by the Deutsche Forschungsgemeinschaft grants WO 2056/1-1 and WO 2056/4-1. The material in this paper was partially presented at 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2017) from September 24th-28th in Vancouver, Canada.

\*Corresponding author

*Email addresses:* [spr@biba.uni-bremen.de](mailto:spr@biba.uni-bremen.de) (Tobias Sprodowski), [m.mehrez.said@mun.ca](mailto:m.mehrez.said@mun.ca) (Mohamed W. Mehrez), [karl.worthmann@tu-ilmenau.de](mailto:karl.worthmann@tu-ilmenau.de) (Karl Worthmann), [gmann@mun.ca](mailto:gmann@mun.ca) (George K.I. Mann), [rgosine@mun.ca](mailto:rgosine@mun.ca) (Raymond G. Gosine), [sag@biba.uni-bremen.de](mailto:sag@biba.uni-bremen.de), [juliana@dep.ufscar.br](mailto:juliana@dep.ufscar.br) (Juliana K. Sagawa), [pank@biba.uni-bremen.de](mailto:pank@biba.uni-bremen.de) (Jürgen Pannek)

This interest also includes formation control [2], which is suitable for distributed applications, e.g. search and rescue operations [3] or exploration of unknown environment [4]. In such applications, given tasks may not be feasible for a single working robot, or these tasks may be executed more efficiently by adopting multiple robots. In formation control, robots not only fulfil a task or goal, but also have to obey constraints due to interaction of robots with the environment as well as with each other, see, e.g., [5]. The former leads to static constraints, e.g., map margins and/or stationary obstacles, while the latter induces dynamic constraints, e.g., collision avoidance. Formation control can be achieved using centralised, decentralised, or distributed control schemes. Centralised control generally leads to superior performance, but its complexity renders this approach to be computationally infeasible for large scale formations. Similarly, if systems with strong coupling are considered, constraint violations may lead decentralised control to be inapplicable. Distributed control schemes can be used for large-scale systems with coupled constraints. However, exchange of information between the subsystems is necessary, cf. e.g. [6, 7], for details.

In this paper, we consider formation control of a group of mobile robots, in which each robot has to be driven to a predefined target state (non-cooperative control). To this end, we use a Distributed Model Predictive Control (DMPC) scheme. The main advantage of model predictive control is its applicability to nonlinear and constrained systems, see e.g. [8], for details on practical applications. In DMPC, based on the most recent location-measurement, each robot solves a local optimal control problem on a finite prediction horizon to calculate an optimal sequence of control values. The first element of the computed control sequence is applied by the corresponding robot before the overall process is repeated at the successive time instant. To prevent collisions among the robots, these conditions are realised via coupling constraints incorporated in the optimal control problem, which is solved by each robot. For constructing these coupling constraints for collision avoidance, in previous works, see, e.g. [9, 10, 11], the complete predicted trajectories were exchanged among the robots. However, since the communication is established using wireless networks, we have to deal with limited network resources. Here, we emphasize that additional techniques may be necessary to mitigate effects resulting from delays and packet dropouts, see, e.g. [12, 13] or [14] and the references therein.

In [15], we introduced a method, which projects predicted trajectories onto a spatial set. Here, so called occupancy tuples encoded as a sequence of grid indices are broadcasted. A number of techniques for distributed control problems with limited communication has been also presented in the literature. In [16], the authors proposed a quantisation method for distributed control problems with a limited amount of bits for communication in each iteration. Here, the level of quantisation is adapted in each iteration. In [17], the authors utilised an encoder-decoder scheme for the linkage between local agents to minimise the quantisation error for a distributed subgradient algorithm. Within the minimisation, a global scaling function is employed to estimate the state of the neighbours. Therein, the quantisation was fixed to a minimum quantisation level for ensuring stability of the whole system. The authors of [18] extended

this approach to an event-triggered quantisation applied on a continuous time system, where the finite-level quantisation, based on time and state of a subsystem, is updated only if the state deviates from the quantised state by more than a defined threshold.

While in the mentioned approaches a distributed optimisation problem is solved, in our approach, the robots solve an optimal control problem in a non-cooperative manner following individual target states. To obey collision avoidance, the subsystems are only coupled by constraints, which are based on the quantised prediction exchange. The quantisation is constant and equidistant over the whole spatial set. This allows to communicate predictions in the form of integers instead of floating-point values, and, thus, reduces the communication effort. Hence, instead of the whole prediction, each robot broadcasts the grid cell indices of those cells (occupancy tuples), which will be occupied at future time instants. Experiments considering this quantisation method were conducted in our previous work [15]. Here, we aim to reduce the communication load further. Thus, instead of sending full predictions in each iteration, we broadcast only the altered cells via a differential communication scheme. In this framework, each robot saves the previously received occupancy tuples of other robots and, thus, is able to reconstruct the respective complete information from the newly received occupied cell indices. Furthermore, we derive a lower bound for the cell size and sufficient conditions based on the exchanged quantised information to guarantee a necessary safety margin among the robots. We use this quantisation for communication only, while the optimisation problem is conducted in the original spatial set. Since the received occupancy tuples impose a quantisation error, the (re-) construction of collision avoidance constraints leads to a constraint tightening and therefore shrinks the set of admissible control actions of each robot. Given an initially feasible starting condition for each robot (initial feasibility), we show that a feasible solution can be found at each time instant (recursive feasibility). Moreover, we investigate numerically the trade-off due to the quantisation. To this end, we examine the closed-loop performance of the simulations for different cell sizes.

The paper is organised as follows: We first formalise the problem setting, recap the grid generation and derive a suitable representation of the coupling constraints for collision avoidance in Section 2. Then, we present the utilised DMPC scheme and analyse initial and recursive feasibility in Section 3. In Section 4, we propose a differential communication scheme. Then, we investigate the proposed method by means of numerical simulations for holonomic as well as non-holonomic robots in Section 5. Last, we draw conclusions and give an outlook on future topics.

**Notation:**  $\mathbb{R}$  and  $\mathbb{N}$  denote the real and natural numbers, respectively.  $\mathbb{N}_0 := \mathbb{N} \cup \{0\}$  represents the non-negative integers and  $\mathbb{R}_{\geq 0}$  the non-negative real numbers. For integers  $a, b \in \mathbb{N}_0$  with  $a \leq b$ , the expression  $[a : b]$  denotes the set  $\{a, a + 1, \dots, b\}$ . Moreover, for a vector  $x \in \mathbb{R}^n$ ,  $n \in \mathbb{N}$ , we define the infinity norm  $\|x\|_\infty := \max_{i \in [1:n]} |x_i|$ . Furthermore, for a sequence  $C = (c_i)_{i=1}^{|C|}$ ,  $|C| \in \mathbb{N}$ , we define the operation  $\text{replace}(A, B; C) :=$

{ if  $c_i = A$ , then  $c_i := B, i = [1 : |C|]$  } to replace every element of the sequence  $C$ , which is equal to  $A$ , by  $B$ .  $\mathbf{find}(A, C) := \inf_{i \in [1:|C|]} \{i \in \mathbb{N} : c_i = A\}$  returns the position of the first appearance of element  $A$  in sequence  $C$ . If  $A$  is not contained in  $C$ ,  $\mathbf{find}$  returns  $\infty$ . To obtain a value  $n$  representing a time index from an element  $A$ , we define the function  $n := \mathbf{time}(A)$ . Finally, given two finite sequences  $A$  and  $B$ , the function  $\mathbf{append}(A, B)$  appends the elements of  $B$  to  $A$ .

## 2. Problem Setting

We consider a group of  $P$  mobile robots with  $P \in \mathbb{N}$ . Each robot  $p$  is following a discrete time model defined as a control-affine system

$$z_p^+ = f(z_p, u_p) := f_0(z_p) + \sum_{i=1}^m f_i(z_p) u_{p,i}, \quad p \in [1 : P] \quad (1)$$

with (analytic) vector fields  $f_i : \mathbb{R}^d \rightarrow \mathbb{R}^d$ ,  $i \in [0 : m]$ , with  $d \geq 2$  and  $d \in \mathbb{N}$ . Here, the vector fields describe the transition of the system component-wise, each driven by a control variable  $u_{p,i}$ . The successor state  $z_p^+$  is determined by the current state  $z_p$  and control  $u_p := (u_{p,1}, \dots, u_{p,m})$ . The first two components  $x_p$  and  $y_p$  of the state  $z_p \in \mathbb{R}^d$  represent the (planar) position of the robot. State and control constraints are modelled by

$$\begin{aligned} z_p \in Z &:= [-\bar{x}, \bar{x}] \times [-\bar{y}, \bar{y}] \times \tilde{Z} && \text{with } \tilde{Z} \subseteq \mathbb{R}^{d-2} \text{ and } \bar{x}, \bar{y} > 0, && (2) \\ u_p \in U &\subset \mathbb{R}^m, && && (3) \end{aligned}$$

respectively, where  $U$  is a compact, convex set with non-empty interior. As the robots are moving in continuous time, the connecting path of a robot  $p$  in this time discrete model between two successive time instants  $z_p$  and  $z_p^+$  is modelled by linear interpolation.

Moreover, the following assumption is needed in our analysis later on:

**Assumption 1 (Immediate Hold).** *Each robot can come to an immediate hold, i.e.*

$$\forall z_p \in Z \exists \bar{u}_p \in U : z_p = f(z_p, \bar{u}_p). \quad (4)$$

This may be difficult to ensure if actuator dynamics are taken into account. Yet, Assumption 1 is, e.g., always satisfied for driftless ( $f_0 \equiv 0$ ) systems (purely kinematic models of mobile robots), if the origin is contained in the control set  $U$ .

For a given time interval  $[0, N]$ , initial state  $z_p^0 \in Z$  and finite sequence  $\mathbf{u}_p : [0 : N - 1] \rightarrow U$  of control values denoted as

$$\mathbf{u}_p = (u_p(0), u_p(1), \dots, u_p(N - 1)),$$

the predicted state trajectory of robot  $p$  is given by

$$\mathbf{z}_p^{\mathbf{u}}(\cdot; z_p^0) := (z_p^u(0; z_p^0), z_p^u(1; z_p^0), \dots, z_p^u(N; z_p^0)),$$

where the subscript  $p$  of  $\mathbf{u}_p$  is skipped in  $\mathbf{z}_p^{\mathbf{u}}(\cdot; z_p^0)$  to simplify the notation. Initial states  $z_p^0$  and target states  $z_p^* \in Z$  are individually set for each robot.

Note that the model (1) covers (kinematic models of) both holonomic and non-holonomic robots. In Section 5, we consider both to numerically analyse our approach.

**Example 1 (Holonomic Robots).** *The kinematic model is given by*

$$\begin{pmatrix} x_p \\ y_p \end{pmatrix}^+ = \begin{pmatrix} x_p \\ y_p \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \end{pmatrix} u_{p,1} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u_{p,2} \quad (5)$$

with  $d = 2$  and  $\tilde{Z} = \emptyset$  according to the control-affine structure (1). The vector fields  $f_1 = (1 \ 0)^\top$  and  $f_2 = (0 \ 1)^\top$  describe the transition of the system imposed by the control variables. Here,  $u_{p,1}$  and  $u_{p,2}$  denote the linear speed in both dimensions. The set  $U \subset \mathbb{R}^2$  is defined as

$$U := \left\{ u_p = \begin{pmatrix} u_{p,1} \\ u_{p,2} \end{pmatrix}, \|u_p\|_2 \leq \bar{u} \right\}, \quad \bar{u} > 0, \quad (6)$$

that is the maximum distance travelled is uniformly bounded in terms of the Euclidean distance. This induces a coupling of the inputs  $u_{p,1}$  and  $u_{p,2}$ . Assumption 1 is reduced to  $U$  containing the origin and is, thus, satisfied.

**Example 2 (Non-holonomic Robot).** *The modelling of the non-holonomic robot requires an additional state dimension in order to represent the orientation. The kinematic model is described by*

$$\begin{pmatrix} x_p \\ y_p \\ \theta_p \end{pmatrix}^+ = \begin{pmatrix} x_p \\ y_p \\ \theta_p \end{pmatrix} + \begin{pmatrix} \cos(\theta_p) \\ \sin(\theta_p) \\ 0 \end{pmatrix} u_{p,1} + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} u_{p,2} \quad (7)$$

with  $d = 3$ ,  $u_p = (u_{p,1}, u_{p,2})^\top \in U \subseteq \mathbb{R}^2$ , where  $\theta_p$  denotes the heading of the robot and  $\tilde{Z} = \mathbb{R}$  the orientation. Here, the vector field  $f_1$  describes the state transition by the imposed linear speed while  $f_2$  defines the influence of the orientation control. Moreover, the linear speed  $u_{p,1}$  and angular speed  $u_{p,2}$  of the robots are bounded by box constraints

$$U := [-\bar{u}_1, \bar{u}_1] \times [-\bar{u}_2, \bar{u}_2], \quad \bar{u}_1, \bar{u}_2 > 0. \quad (8)$$

Similar to Example 1, Assumption 1 holds.

Note that the holonomic robot is fully *actuated*, as the controllability matrix has full rank. For the non-holonomic case, the number of actuators is lower than degrees of freedom as the control and the robot is not able to follow an

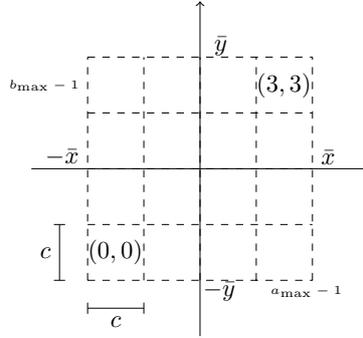


Figure 1: Example of a grid  $\mathcal{G}$  with  $a_{\max} = b_{\max} = 4$ , from [15]

arbitrary chosen trajectory due to the non-holonomic constraints. Hence, the robot is *underactuated*.

The mobile robots exchange their predicted trajectories in order to avoid collisions. In the remainder of this section, we partition our spatial set into a grid of squared cells via a quantisation. This allows to alleviate the communication requirements since only cell numbers instead of spacial positions are exchanged. However, due to less accurate information on the states of the other robots, a minimum cell size is introduced to ensure collision avoidance.

### 2.1. Occupancy Grid and Quantised Communication

We partition the spatial set  $[-\bar{x}, \bar{x}] \times [-\bar{y}, \bar{y}]$  into a grid  $\mathcal{G} := [0 : a_{\max} - 1] \times [0 : b_{\max} - 1] \subset \mathbb{N}_0^2$  of squared cells such that each cell has a width (and height)  $c$ . Hence, we have

$$2\bar{x} = a_{\max}c \quad \text{and} \quad 2\bar{y} = b_{\max}c, \quad (9)$$

where  $\bar{x}$  and  $\bar{y}$  are both multiples of  $c$ . Furthermore, we assign an index  $(a, b) \in \mathcal{G}$  to each cell (see Fig. 1 for details) and denote the boundary layer of  $\mathcal{G}$  by  $\partial\mathcal{G} := \{(a, b) \mid a \in \{0, a_{\max} - 1\} \vee b \in \{0, b_{\max} - 1\}\}$ .

Next, we define the quantisation  $q : Z \rightarrow \mathcal{G}$ ,

$$q(z_p) = (a_p, b_p) := \left( \min \left\{ a_{\max} - 1, \left\lfloor \frac{x_p + \bar{x}}{c} \right\rfloor \right\}, \min \left\{ b_{\max} - 1, \left\lfloor \frac{y_p + \bar{y}}{c} \right\rfloor \right\} \right), \quad (10)$$

to map a state  $z_p$  onto the grid  $\mathcal{G}$ . This allows us to introduce the sequence of *occupancy tuples*  $\mathcal{I}_p(n) \in (\mathbb{N}_0 \times \mathcal{G})^{N+1}$  at time  $n$  as

$$\mathcal{I}_p(n) := (n + k, q(z_p^u(k; z_p^0)))_{k=0}^N.$$

These occupancy tuples  $\mathcal{I}_p(n)$  represent the predicted occupied cells of robot  $p$ . To access the  $k$ -th tuple, we define  $\mathcal{I}_p(n)(k)$  to denote the  $k$ -th tuple in  $\mathcal{I}_p(n)$ . Often, we use the notation

$$(a_p^u(k; z_p^0), b_p^u(k; z_p^0)) := q(z_p^u(k; z_p^0)) \quad (11)$$

to have direct access to the index of an occupied cell. While each robot  $p$  broadcasts its sequence of occupancy tuples  $\mathcal{I}_p(n)$ , it also assembles all received occupancy tuples to an *occupancy grid*

$$\mathbf{i}_p(n) := (\mathcal{I}_1(n), \dots, \mathcal{I}_{p-1}(n), \mathcal{I}_{p+1}(n), \dots, \mathcal{I}_P(n)), \quad (12)$$

which contains information about cells occupied by other robots. In the following Subsection 2.2, we will utilise this information to derive coupling constraints. To this end, we require the backward mapping  $f_c : \mathbb{N}_0 \times \mathcal{G} \rightarrow \mathbb{R}^2$ , given by

$$f_c((n, a, b)) = \underbrace{((a + 0.5)c - \bar{x}, (b + 0.5)c - \bar{y})}^{\text{=:}(x^c, y^c)}, \quad (13)$$

which maps a cell index  $(a, b) \in \mathcal{G}$  to the spatial coordinates of the cell centre  $(x^c, y^c) \in [-\bar{x}, \bar{x}] \times [-\bar{y}, \bar{y}]$  ignoring the time index  $n$ .

Firstly, the main purpose of using quantised information (the occupancy grid) is reducing the amount of broadcasted data. This is achieved in two ways: For one, the robots broadcast integers instead of floating point values. And secondly, the quantisation of the spatial information allows for broadcasting differential updates similar to revision control systems (see [19, 20]), cf. Section 4 for a respective communication scheme.

## 2.2. Minimum Cell Size and Construction of the Constraints

In this subsection, we first derive a minimum cell size  $\underline{c}$  based on the aforementioned problem setting. Similar to Artstein's circles [21], in which an unnoticed constraint violation between two time instants is caused by a large sampling step, we have to ensure a lower bound to avoid skipping cells due to the discrete time setting and the quantisation. Then, we introduce a safety margin to prevent robots to cross cells occupied by other robots in between two successive instants. Thereafter, we present two approaches to construct respective coupling constraints. While the first is based on the infinity norm, the second one utilises squircles.

**Cell Size:** Our goal is to determine a lower bound  $\underline{c}$  on the cell size  $c$  to avoid skipping cells in successive time instants. Hence, the following condition has to be ensured:

$$\|f_c(0, q(z_p)) - f_c(0, q(f(z_p, u_p)))\|_\infty \leq \underline{c} \quad \forall (z_p, u_p) \in Z \times U. \quad (14)$$

As  $f_c$  is independent of the time component, cf. (13), Inequality (14) reads that between two consecutive time instants the robot should not make a cell index change of more than one in any direction. Since (14) is based on the



be estimated by (16) (left). Similarly, for  $f_i$ , using continuity and compactness of  $f_i$ ,  $i \in [1 : m]$ , this holds for (16) (right) as an upper bound for the transition of the states. Combining  $c_{f_0}$  and  $c_{f_i}$ ,  $i \in [1 : m]$ , we can guarantee (14) to hold for all  $c \geq \underline{c}$  with (15) showing the assertion.  $\square$

**Example 3 (Minimum cell size for robots).** *Reconsider the holonomic and the non-holonomic robot from Examples 1 and 2. Since both robots are modelled as driftless systems, we obtain  $c_{f_0} = 0$ . In the holonomic case, we immediately get  $c_{f_1} = c_{f_2} = 1$  (fulfilling (14)) from the system model. In the non-holonomic case, we similarly have  $c_{f_1} = 1$  as the linear speed imposes the constraint movement of the robot in both dimensions. However, since (14) relies on spatial coordinates  $j \in \{1, 2\}$  only, we now obtain  $c_{f_2} = 0$  as the vector field  $f_2$  describes the transformation of orientation of the system. Hence,  $f_2$  does not have any impact on the minimum cell size. Combined, utilising the constraint (6) and the definition (15) reveals  $\underline{c} = 2\bar{u}$  in the holonomic case. For the non-holonomic case, this leads to  $\underline{c} = \bar{u}_1$  utilising constraint (8) and condition (15).*

**Safety Margin:** We introduce a safety margin in addition to the minimum cell size to take the following issues into account: First, to regard the physical dimensions of the robots, an appropriate minimum distance  $d_{\min}$  among all of them must be ensured. Second, as the robots are moving in a continuous setting, while observed in a discrete manner (1), a possible constraint violation between two successive time instants must be avoided. Note that this excludes swapping of cells between two neighbored robots which leads to a collision (see Fig. 2 (left)). The safety margin is defined as follows:

**Definition 1 (Safety Margin).** *Consider two robots  $p$  and  $q$  with  $p, q \in [1 : P]$ ,  $p \neq q$ , a required physical minimum distance  $d_{\min}$  and a minimum cell size  $\underline{c}$ . Then we say that the robots meet the safety margin if*

$$\left\| \begin{pmatrix} x_p \\ y_p \end{pmatrix} - \begin{pmatrix} x_q \\ y_q \end{pmatrix} \right\|_{\infty} \geq \max\{d_{\min}, \underline{c}\} \quad (17)$$

holds, where  $x_i, y_i \in Z$ ,  $i \in \{p, q\}$  denote the spatial coordinates of the robots.

Based on Definition 1, we have to ensure that the robots incorporate the safety margin in their predicted state trajectories using the *states* of the other robots:

**Definition 2 (Prediction Safety Margin).** *Consider two robots  $p$  and  $q$  with dynamics (1) as well as state (2) and control constraints (3), where  $p, q \in [1 : P]$  with  $p \neq q$ . Moreover, suppose a required physical minimum distance  $d_{\min}$ , a minimum cell size  $\underline{c}$  from Proposition 1 and a time interval  $[0 : N]$  to be given. Then, robots  $p$  and  $q$  are said to obey the prediction safety margin if*

$$\left\| \begin{pmatrix} x_p^u(k; x_p^0) \\ y_p^u(k; y_p^0) \end{pmatrix} - \begin{pmatrix} x_q^u(k; x_q^0) \\ y_q^u(k; y_q^0) \end{pmatrix} \right\|_{\infty} \geq \max\{d_{\min}, \underline{c}\}, \quad k \in [0 : N], \quad (18)$$

holds where  $x_i^u(k; x_i^0)$  and  $y_i^u(k; y_i^0) \in Z$  denote the spatial coordinates of the robots over the time interval  $[0 : N]$  for  $i \in \{p, q\}$ .

Unfortunately, (18) is not applicable in the distributed setting with communication of quantised data. Hence, we need to strengthen our requirements such that (18) holds by utilising the occupancy grid  $\mathbf{i}_p(n)$  only.

**Proposition 2 (Quantised Safety Margin).** *Consider a group of  $P$  robots with dynamics (1) as well as state (2) and control constraints (3) to be given and the quantisation (10) to be defined for  $c \geq \underline{c}$  with  $\underline{c}$  from Proposition 1. If for any two robots  $p, q \in P$  with  $p \neq q$  inequality*

$$\left\| \begin{pmatrix} x_p^u(k; x_p^0) \\ y_p^u(k; y_p^0) \end{pmatrix} - f_c(\mathcal{I}_q(n)(k)) \right\|_{\infty} \geq \Psi := \max\{d_{\min}, \underline{c}\} + \frac{c}{2}, \quad k \in [0 : N] \quad (19)$$

holds using local and received quantised state information  $\mathcal{I}_q(n)$ , then all robots in  $P$  satisfy the prediction safety margin (18).

**Proof:** As robot  $p$  only receives the cell index of robot  $q$ , the exact position of robot  $q$  as used in Definition 2 is not known to robot  $p$ . Utilising the backward mapping  $f_c$ , an approximation of the position of robot  $q$  can be obtained. This leads to

$$\left\| \begin{pmatrix} x_p^u(k; x_p^0) \\ y_p^u(k; y_p^0) \end{pmatrix} - \begin{pmatrix} x_q^u(k; x_q^0) \\ y_q^u(k; y_q^0) \end{pmatrix} \right\|_{\infty} \geq \left\| \begin{pmatrix} x_p^u(k; x_p^0) \\ y_p^u(k; y_p^0) \end{pmatrix} - f_c(\mathcal{I}_q(n)(k)) \right\|_{\infty} - \underbrace{\left\| \begin{pmatrix} x_q^u(k; x_q^0) \\ y_q^u(k; y_q^0) \end{pmatrix} - f_c(\mathcal{I}_q(n)(k)) \right\|_{\infty}}_{\leq \frac{c}{2}}$$

with  $k \in [0 : N]$ , where the last term describes the maximal distance between a true and a corresponding quantised position within a cell. Hence, if (19) holds, then we can conclude condition (18) to be satisfied, showing the assertion.  $\square$

While Proposition 2 solves the first issue of collision avoidance regarding quantised communication at the time instants  $k \in [0 : N]$ , the issue of collision avoidance between these discrete time instants remains open. The latter is due to the time discretisation and may occur as crossing an occupied cell of another robot cannot be recognised between two successive time instants, but may cause a violation of (19) (see Fig. 2; right). As additional information regarding the connecting path between two successive states  $z_p(n)$  and  $z_p(n+1)$  is not available, we consider linear interpolation between these states. Here, we have to distinguish between two cases: the neighbouring robot  $q$  changes its cell ( $\mathcal{I}_q(n)(k) \neq \mathcal{I}_q(n)(k+1)$ ) or remains in its cell ( $\mathcal{I}_q(n)(k) = \mathcal{I}_q(n)(k+1)$ ). This violation of (19) in the second case is depicted in Fig. 2; right. To compensate for intermediate violations of the safety margin, we further strengthen the quantised safety margin (19).

**Proposition 3 (Intermediate Safety Margin).** *Let the assumptions of Proposition 2 hold with feasible initial conditions, and  $t_k$  denote the continuous time*

instants, which correspond to  $k \in [0 : N]$ . If additionally

$$\left\| \begin{pmatrix} x_p^u(k; x_p^0) \\ y_p^u(k; y_p^0) \end{pmatrix} - f_c(\mathcal{I}_q(n)(k)) \right\|_\infty \geq \Psi_q(\mathcal{I}_q(n)(k), \mathcal{I}_q(n)(k-1)) \quad (20)$$

holds for all  $k \in [1 : N]$  with

$$\Psi_q(\mathcal{I}_q(n)(k), \mathcal{I}_q(n)(k-1)) = \begin{cases} \bar{\Psi} := \Psi + \underline{c}, & \text{if } \mathcal{I}_q(n)(k) \neq \mathcal{I}_q(n)(k-1) \\ \underline{\Psi} := \Psi + \frac{\underline{c}}{2} \cdot \cos\left(\frac{\pi}{4}\right), & \text{otherwise,} \end{cases} \quad (21)$$

then Condition (17) holds for all time instants in  $[t_0, t_N] \subset \mathbb{R}$ .

**Proof:** We consider the two cases  $\mathcal{I}_q(n)(k) \neq \mathcal{I}_q(n)(k-1)$  and  $\mathcal{I}_q(n)(k) = \mathcal{I}_q(n)(k-1)$  for the time instants  $k-1, k$ .

Regarding the first case, we assume, that the robots have to ensure a safety margin  $\Psi$  among each other. Regarding the safety margin (17), a violation may occur for a time instant  $t \in [t_{k-1}, t_k] \in \mathbb{R}$ : Robot  $p$  is located at minimum distance (19) to the cell  $\mathcal{I}_q(n)(k-1)$  occupied by robot  $q$  and moves directly towards it, whereas robot  $q$  leaves cell  $\mathcal{I}_q(n)(k-1)$  right before time instant  $k$ . As no information for  $p$  is available between  $t_{k-1}$  and  $t_k$ , we utilize the bound given by Equation (15) regarding the imposed control bounded by condition (14) on robot  $p$ . This reveals that the violation of the safety margin (17) for a time instant in  $[t_{k-1}, t_k] \in \mathbb{R}$  caused by robot  $p$  is given by  $s = \underline{c}$  and revealing (21) for the first case.

Regarding the second case, i.e.  $q$  remains in cell  $\mathcal{I}_q(n)(k)$ , the worst case is given by the following (Fig. 2 (right)): At time instants  $k-1$  and  $k$ , Robot  $p$  is located at minimum distance (19) to the cell  $\mathcal{I}_q(n)(k)$  and crosses the cell defined by the quantised safety margin (19) with an angle of  $\pi/4$ . As crossing an entire cell is impossible due to the definition of  $\underline{c}$  in (15), the crossing trajectory leads to a right triangle. Due to our control constraints, (15) reveals that the length of the hypotenuse is at most  $\underline{c}$ . Since we utilise the infinity norm in (19), the violation is symmetric in both spatial coordinates. Hence, we may consider any coordinate and can split the violation triangle equally with length of hypotenuse  $\frac{\underline{c}}{2}$ . Accordingly, the maximal violation of (17) for a time instant in  $[t_{k-1}, t_k] \in \mathbb{R}$  caused by robot  $p$  is given by  $s = \frac{\underline{c}}{2} \cos\left(\frac{\pi}{4}\right)$ .

Combining both cases reveals condition (20) showing the assertion for  $[t_1, t_N]$ . Since the case  $\mathcal{I}_q(n)(k) \neq \mathcal{I}_q(n)(k-1)$  reveals a more conservative bound, Eq.(21) extends the interval to  $[t_0, t_N]$ , which completes the proof.  $\square$

Hence, we can conclude, that Proposition 3 reveals sufficient conditions to guarantee the prediction safety margin (17) at and in between time instants  $[0 : N]$  based on quantised information exchange.

**Derivation of the Coupling Constraints:** Each robot  $p$  is supposed to generate its predicted trajectory  $\mathbf{z}_p^u(\cdot; z_p^0)$  with respect to the received occupancy grid  $\mathbf{i}_p(n)$  to guarantee the intermediate the safety margin. Here, we present two different approaches for modelling such constraints. The first is based on the infinity norm while the second relies on squirecles.

If the distance is measured via the infinity norm, that is w.r.t. to a box with centre  $f_c(\mathcal{I}_q(n)(k))$  and width  $\Psi_q(\mathcal{I}_q(n)(k), \mathcal{I}_q(n)(k-1))$ , this corresponds to

$$\begin{aligned} g_{q,k}^p &:= g(z_p^u(k; z_p^0), \mathcal{I}_q(n)(k), \mathcal{I}_q(n)(k-1)) \\ &= \left\| \begin{pmatrix} x_p^u(k; x_p^0) \\ y_p^u(k; y_p^0) \end{pmatrix} - f_c(\mathcal{I}_q(n)(k)) \right\|_\infty - \Psi_q(\mathcal{I}_q(n)(k), \mathcal{I}_q(n)(k-1)) \geq 0 \end{aligned}$$

for all  $k \in [1 : N]$ . Note that since the infinity-norm is not differentiable, a gradient-free optimisation method should be used, which is, in general, more time consuming than gradient-based methods, see, e.g. [22].

In the second approach, a squircle [23], a special case of a superellipsoid, is used as an outer but differentiable approximation of the box with centre  $f_c(\mathcal{I}_q(n)(k))$  and width  $\Psi_q(\mathcal{I}_q(n)(k), \mathcal{I}_q(n)(k-1))$ . The polar coordinates of the squircle are given by

$$\begin{aligned} x(\phi) &= 2^{-3/4}h \cdot \text{sgn}(\cos(\phi)) \cdot |\cos(\phi)|^{\frac{1}{2}} \\ y(\phi) &= 2^{-3/4}h \cdot \text{sgn}(\sin(\phi)) \cdot |\sin(\phi)|^{\frac{1}{2}} \end{aligned}$$

with  $\phi \in [0, 2\pi)$ , where  $h = 2 \cdot \Psi_q(\mathcal{I}_q(n)(k), \mathcal{I}_q(n)(k-1))$  as the width of the square. Then, with  $\text{sgn}(x)^2 = 1$  and  $\left(|x|^{\frac{1}{2}}\right)^2 = |x|$ , we can use the euclidean distance  $\sqrt{x^2(\phi) + y^2(\phi)} = 2^{-1/4} \sqrt{|\cos(\beta)| + |\sin(\beta)|}$  to construct the coupling constraints via

$$\begin{aligned} g_{q,k}^p &:= g(z_p^u(k; z_p^0), \mathcal{I}_q(n)(k), \mathcal{I}_q(n)(k-1)) = \left\| \begin{pmatrix} x_p^u(k; x_p^0) \\ y_p^u(k; y_p^0) \end{pmatrix} - f_c(\mathcal{I}_q(n)(k)) \right\|_2 \\ &\quad - \frac{\sqrt{|\cos(\beta)| + |\sin(\beta)| + |\varepsilon_1|}}{2^{(1/4)}} \Psi_q(\mathcal{I}_q(n)(k), \mathcal{I}_q(n)(k-1)) \geq 0. \end{aligned}$$

The coefficient of the coordinates describes the additional distance by the approximation of the squircle. The latter depends on the direction of the position of robot  $p$  in relation to robot  $q$  expressed by angle  $\beta$  with

$$\beta = \arctan\left(\frac{y_p^u(k) - y^c(k)}{x_p^u(k) - x^c(k)}\right),$$

where  $(x^c(k), y^c(k))$  is obtained from  $\mathcal{I}_q(n)(k)$  according to (13).

In summary, we have constructed the coupling constraint function

$$G(z_p^u(k; z_p^0), \mathbf{i}_p(n)(k), \mathbf{i}_p(n)(k-1)) = \left(g_{1,k}^p, \dots, g_{p-1,k}^p, g_{p+1,k}^p, \dots, g_{P,k}^p\right) \geq 0 \quad (22)$$

for  $k \in [1 : N]$ . If squircles are used, the constraint functions are differentiable. If these constraints are satisfied, the intermediate safety margin from Proposition 3 based on the information  $\mathbf{i}_p(n)$  is ensured.

### 3. Distributed Model Predictive Control

Each robot  $p$  with  $p \in [1 : P]$  is equipped with a local controller, which aims to steer it to its individual target states  $z_p^*$  maintaining constraint satisfaction. Here, we consider a model predictive controller, which is a three step procedure: First, the current state of the robot is obtained. Thereafter, a finite horizon optimal control problem is solved. In the last step, only the first element of the computed optimal control sequence is applied to the robot. Shifting the horizon then allows to repeat this scheme iteratively.

To define the optimal control problem to be solved in the second step, we introduce stage costs  $\ell_p : Z \times U \rightarrow \mathbb{R}_{\geq 0}$ , for which there exists  $u_p^* \in U$  such that

$$\ell_p(z_p^*, u_p^*) = 0 \quad \text{and} \quad \inf_{u_p \in U} \ell_p(z_p, u_p) > 0 \quad \forall z_p \in Z \setminus \{z_p^*\}.$$

Given the current state  $z_p^0 = z_p(n)$  and received information  $\mathbf{i}_p^0 = \mathbf{i}_p(n)$ , minimising the stage costs over a finite horizon of length  $N$  reveals the optimal control problem

$$\min_{\mathbf{u}_p} J_p^N(\mathbf{u}_p; z_p^0, \mathbf{i}_p^0) := \sum_{k=0}^{N-1} \ell_p(z_p^u(k; z_p^0), u_p(k)) \quad (23)$$

subject to

$$\begin{aligned} z_p^u(k+1; z_p^0) &= f(z_p^u(k; z_p^0), u_p(k)), & k \in [0 : N-1], \\ u_p(k) &\in U, & k \in [0 : N-1], \\ G(z_p^u(k; z_p^0), \mathbf{i}_p^0(k), \mathbf{i}_p^0(k-1)) &\geq 0, & k \in [1 : N], \\ z_p^u(k; z_p^0) &\in Z, & k \in [1 : N]. \end{aligned}$$

To avoid technical difficulties, existence of a minimizer of (23) is assumed if the optimal control problem is feasible. As a result, we obtain an optimal control sequence

$$\mathbf{u}_p^* = (u_p^*(0), \dots, u_p^*(N-1)) \quad (24)$$

and introduce the corresponding value function  $V_p^N : Z \times (\mathbb{N}_0 \times \mathcal{G})^{(P-1)(N)}$  via

$$V_p^N(z_p^0, \mathbf{i}_p^0) = J_p^N(\mathbf{u}_p^*; z_p^0, \mathbf{i}_p^0). \quad (25)$$

The DMPC scheme — in analogy to Richards and How [24, 25] — based on this optimal control problem is presented in Algorithm 1. With respect to the sequential order of optimisation, we have to face an asymmetry of available information: At time instant  $n$ , robot  $p$  has information  $\mathcal{I}_q(n)$  on the preceding robots  $q \in [1 : p-1]$ . For successive robots  $q \in [p+1 : P]$ , however, only  $\mathcal{I}_q(n-1)$  is available. Utilizing Assumption 1, we can define an information shift without additional communication among the robots by assuming that robots  $q \in [p+1 : P]$  hold their positions and we set  $\mathcal{I}_q(n)(N) :=$

---

**Algorithm 1** DMPC-Algorithm for the overall system

---

```

1: Given admissible, initial states  $z_p^0$  for all  $p \in [1 : P]$ 
2: for  $p = 1$  to  $P$  do
3:   for  $k = 1$  to  $N$  do
4:     Set  $\mathcal{I}_p(0)(k) := (k, q(z_p^0))$ 
5:   end for
6:   Broadcast  $\mathcal{I}_p(0)$ 
7: end for
8: for  $n = 0, 1, \dots$  do
9:   for  $p = 1$  to  $P$  do
10:    Measure  $z_p(n)$ 
11:    if  $n = 0$  then
12:      Receive  $\mathcal{I}_q(n)$  for  $q \in [1 : P] \setminus \{p\}$ 
13:    else
14:      Receive  $\mathcal{I}_q(n)$  for  $q \in [1 : p - 1]$  and  $\mathcal{I}_q(n - 1)$  for  $q \in [p + 1 : P]$ 
15:      Assemble  $\mathcal{I}_q(n)$ ,  $q \in [p + 1 : P]$ , using Algorithm 2
16:    end if
17:    Set  $\mathbf{i}_p(n)$  according to (12)
18:    Solve optimal control problem (23) and Apply  $u_p^*(0)$ 
19:    Broadcast  $\mathcal{I}_p(n)$ 
20:  end for
21: end for

```

---



---

**Algorithm 2** Resolving asymmetry of communication data for robot  $p$ 

---

```

1: Given robot  $p$  and communication data  $\mathcal{I}_q(n)$  for  $q \in [1 : p - 1]$  and  $\mathcal{I}_q(n - 1)$  for  $q \in [p + 1 : P]$ 
2: for  $q = p + 1$  to  $P$  do
3:   for  $k = 1$  to  $N - 1$  do
4:     Set  $\mathcal{I}_q(n)(k) := \mathcal{I}_q(n - 1)(k + 1)$ 
5:   end for
6:   Set  $\mathcal{I}_q(n)(N) := \mathcal{I}_q(n - 1)(N)$ 
7: end for

```

---

$\mathcal{I}_q(n - 1)(N)$ . The detailed construction of this shift in the constraints for robot  $p$  is given in Algorithm 2.

In general, the calculation of an admissible set of controls for all robots in the initialisation of Algorithm 1 is conducted in a centralised way [25]. Here, we do not impose terminal condition to ensure stability as imposed in [25, 26]. Our approach follows [9, 27], which allows us to employ the system without terminal conditions.

Due to the absence of terminal conditions, we need to ensure that for each time instant  $n \in \mathbb{N}_0$  and each robot  $p \in [1 : P]$  there always exists a solution to the optimal control problem (23). If the latter holds, then Algorithm 1 is executable and does not terminate unexpectedly. This particularly holds true for

the initialisation. To show initial feasibility, we need the following assumption:

**Assumption 2.** *Given a set of  $P$  robots, we have that for  $z_p^0 \in Z$  and for all  $p, q \in [1 : P]$  with  $p \neq q$ , condition  $g_{q,0}^p(z_p^0, q(z_q^0), q(z_q^0)) \geq 0$  holds.*

Assumption 2 states that the scenario is well-defined regarding the state and coupling constraints. We show that there exists a solution to the initial optimal control problem (initial feasibility) and for all the subsequent optimal control problems arising in Algorithm 1. In the context of MPC, the latter property of Algorithm 1 is called recursive feasibility.

**Theorem 1 (Initial and Recursive Feasibility).** *Consider a set of  $P$  robots with underlying model (1) and constraints (2), (3) satisfying Assumptions 1 and 2 and the condition, that for each time instant  $n \in \mathbb{N}$ , a minimiser of (23) exists if the feasible set is non-empty. If Algorithm 1 is applied, then the problem is recursively feasible, i.e. for all  $n \in \mathbb{N}_0$  and all  $p \in [1 : P]$  there exists a solution to optimal control problem (23).*

**Proof:** We proof the assertion by induction. For  $n = 0$ , consider  $p \in [1 : P]$  to be arbitrary but fixed. Due to Assumption 2, the coupling constraint (22) holds for  $p \in [1 : P]$  and  $k = 0$ . Utilising Assumption 1, there exist  $\bar{u}_p \in U$  such that  $z_p^0 = f(z_p^0, \bar{u})$  for all  $p \in [1 : P]$ . Hence, for this choice,  $z_p^u(k; z_p^0) \equiv z_p^0 \in Z$  and (22) hold for all  $k \in \mathbb{N}_0$  and all  $p \in [1 : P]$ . Accordingly,  $\bar{u}_p$  is a feasible solution of optimal control problem (23). As  $p$  was chosen arbitrarily, this hold for all  $p \in [1 : P]$ .

Regarding the induction step from  $n - 1$  to  $n$  suppose there exists a solution for  $n - 1 \geq 0$ . Denote the respective optimal controls by  $\bar{u}_p(\cdot)$ . Again suppose  $p \in [1 : P]$  to be arbitrarily fixed. By definition of  $\mathbf{i}_p^0 = \mathbf{i}_p(n)$  and the fixed order of robots, the choice  $u_p(k) := \bar{u}_p(k + 1)$  is feasible for  $k = [0 : N - 2]$ . By Assumption 1 we obtain that for  $z_p^u(N - 1)$  there exists  $\bar{u} \in U$  such that  $z_p^u(N) = z_p^u(N - 1)$  holds. Utilising the construction of  $\mathbf{i}_p(n)(N)$  in Algorithm 2, we then obtain that  $u_p(N - 1) := \bar{u}$  is feasible. Hence, there exists a solution to the optimal control problem (23) for the chosen robot  $p \in [1 : P]$ . Since the latter was chosen arbitrarily, the assertion follows.  $\square$

Having shown initial and recursive feasibility, we can ensure that Algorithm 1 will not terminate unexpectedly. Note that, in contrast to [24, 25], initial or recursive feasibility here is not sufficient for stability, i.e. robots may not converge to their target states. To obtain the latter, stability conditions similar to [9] need to be verified, which is outside the scope of this article and we refer to respective ideas in the outlook.

#### 4. Differential Communication

Considering Algorithm 1 in our quantised setting, we now present a communication scheme, which utilizes differential updates to further reduce the communication load. Each occupancy tuple  $\mathcal{I}_p$  is composed of a sequence of

time stamps and cell indices. The fundamental idea of differential communication is that each robot  $p$  keeps track of the most recent broadcasted occupancy tuples in the storage variable  $\mathcal{I}_p^-$  to create the differential update  $\mathcal{I}_p^c$  of tuples to be broadcasted. If the quantised prediction contains tuples, which are not

---

**Algorithm 3** Preparation of communication data (robot  $p$ )
 

---

```

1: if  $n = 0$  then
2:   Set  $\mathcal{I}_p^c = \mathcal{I}_p^- := \emptyset$ 
3: end if
4: Compute  $\mathcal{I}_p(n)$ 
5: for  $k = 0 : N$  do
6:   if  $\mathcal{I}_p(n)(k) \notin \mathcal{I}_p^-$  then
7:     Set  $\mathcal{I}_p^c := \text{append}(\mathcal{I}_p^c, \mathcal{I}_p(n)(k))$ 
8:   end if
9: end for
10: Broadcast  $\mathcal{I}_p^c$ 
11: Set  $\mathcal{I}_p^- := \mathcal{I}_p(n) \setminus \mathcal{I}_p(n)(0)$ ,  $\mathcal{I}_p^c := \emptyset$ 

```

---

already stored in the memory  $\mathcal{I}_p^-$ , they are appended to the differential update  $\mathcal{I}_p^c$ , which is then broadcasted. For each time shift, the oldest tuple  $\mathcal{I}_p(n)(0)$  then can be deleted as this is no longer required.

Although only altered tuples are broadcasted by Algorithm 3, this reduced information scheme suffices to assemble the complete occupancy grid to formulate the coupling constraints (22) as shown in Algorithm 4.

---

**Algorithm 4** Assembly of  $\mathbf{i}_p$  by robot  $p$ 


---

```

1: Receive  $\mathcal{I}_q^c$  for all  $q \in [1 : P] \setminus \{p\}$ 
2: if  $n = 0$  then
3:   Set  $\mathcal{I}_q^- = \mathcal{I}_q^c$  for all  $q \in [1 : P] \setminus \{p\}$ 
4: else
5:   for  $q \in [1 : P] \setminus \{p\}$  do
6:     for  $j = 0 : |\mathcal{I}_q^c| - 1$  do
7:       if  $\mathcal{I}_q^c(j) \notin \mathcal{I}_q^-$  then
8:          $i := \text{find}(\text{time}(\mathcal{I}_q^c(j)), \mathcal{I}_q^-)$ 
9:         Set  $\mathcal{I}_q^- := \text{replace}(\mathcal{I}_q^-(i), \mathcal{I}_q^c(j); \mathcal{I}_q^-)$ 
10:       end if
11:     end for
12:      $\mathcal{I}_q^- := \text{append}(\mathcal{I}_q^-, \mathcal{I}_q^c(|\mathcal{I}_q^c|))$ 
13:   end for
14: end if
15: Set  $\mathbf{i}_p(n) := (\mathcal{I}_q^-)_{q \in [1:P] \setminus \{p\}}$  and  $\mathcal{I}_q^- := \mathcal{I}_q^- \setminus \mathcal{I}_q^c(n)(0)$  for all  $q \in [1 : P] \setminus \{p\}$ 

```

---

Each robot examines the received update  $\mathcal{I}_q^c$  from robot  $q$  whether the current memory  $\mathcal{I}_q^-$  has to be updated. If the memory is outdated, the tuples with

the corresponding time instant are found and replaced.

As a result, Algorithm 1 can be reformulated based on the differential communication. To this end, in Algorithm 1 we replace lines 14 and 15 by

**Receive**  $\mathcal{I}_q^c$  for all  $q \in [1 : P] \setminus \{p\}$

**Assemble**  $\mathbf{i}_p(n)$  using Algorithm 4

and line 19 by

**Broadcast**  $\mathcal{I}_p^c$  using Algorithm 3.

Thus, utilising Algorithms 3 and 4, the communication load reduces to differential updates instead of full communication, thereby reducing necessary resources in wireless networks. On the other hand, this scheme requires an additional memory component. In this case, as at maximum  $N \cdot P$  tuples have to be stored by each robot, the size of the latter is directly proportional to the prediction horizon.

## 5. Numerical Simulations

In this section, we explore the performance of Algorithm 1 and the improvements resulting from the differential communication (see Algorithms 3 and 4) for holonomic and non-holonomic robots by simulations.

### 5.1. Setup of Holonomic and Non-Holonomic Robots

Within the general setting, we considered a group of  $P = 4$  mobile robots for both, holonomic and non-holonomic setting, considered in Example 1 and 2. We set the state constraints  $Z := [-6(m), 6(m)]^2 \times \tilde{Z}$ . Regarding the controls, we set the bounds  $\sqrt{(\bar{u}_1)^2 + (\bar{u}_2)^2} \leq 0.5$  (m/s),  $\bar{u}_1 \leq 0.5$  (m/s), and  $\bar{u}_2 \leq 0.5$ . Moreover, we used the minimum distance  $d_{\min} = 0.5 + \varepsilon$  (m) between the robots where  $0 < \varepsilon \ll 1$  denotes a numerical margin. We obtained the minimum cell size  $\underline{c} = \max\{\bar{u}_1, \bar{u}_2 + \varepsilon\} = 0.5 + \varepsilon$  (m) via (15) and referring to Example 3. To investigate the effect of the quantisation, we considered the fixed initial and target states of the robots displayed in Table 1 and varied the cell size  $c \in \{0.5, 1.0, 1.5, 2.0\}$ .

Table 1: Initial and target states of the robots

Robot ( $p$ )	Initial states ( $z_p^0$ )	Target states ( $z_p^*$ )
1	$(4.5, 4.5, -3\pi/4)^\top$	$(-4.5, -4.5, \pi)^\top$
2	$(-4.5, 4.5, -\pi/4)^\top$	$(4.5, -4.5, 0)^\top$
3	$(4.5, -4.5, 3\pi/4)^\top$	$(-4.5, 4.5, \pi)^\top$
4	$(-4.5, -4.5, \pi/4)^\top$	$(4.5, 4.5, 0)^\top$

Last, we imposed

$$\|z_p(n) - z_p^*\| \leq 0.01, \quad \text{with } \forall p \in [1 : P] \quad (26)$$

as termination condition of the simulation and denoted the total number of closed-loop iterations by  $n_\#$ .

Simulations for the holonomic robots were conducted utilising the infinity norm in constructing the coupling constraints as shown in Section 2.2. Here, the derivative-free optimisation method COBYLA (Constrained Optimisation by Linear Approximations) [28] was employed in the NLOpt package [29]. The simulations were implemented in C++. For the non-holonomic robots, simulations were performed utilising squircles to compose the coupling constraints presented in Section 2.2. Thus, the (derivative-based) interior-point method IPOPT [30] was used via CasADi toolbox [31] running in MATLAB.

We investigated the performance of Algorithm 1 in terms of the following criteria:

- The average number of broadcasted tuples, which is given by

$$K := \frac{1}{n_\#} \sum_{n=0}^{n_\#} \sum_{p=1}^P \#\mathcal{I}_p^c|_n, \quad (27)$$

where  $\#\mathcal{I}_p^c|_n$  represents the broadcasted tuples of robot  $p$  at time instant  $n$ .

- The accumulated closed-loop stage costs for all robots for one time step  $n$ , which are defined as

$$M_P(n) := \sum_{p=1}^P \ell_p(z_p^{\text{MPC}}(n), u_p^{\text{MPC}}(n)) \quad (28)$$

and the accumulated closed-loop costs over the whole simulation time  $n_\#$  given by

$$M_P^{n_\#} := \sum_{n=0}^{n_\#} M_P(n). \quad (29)$$

The closed-loop costs illustrate the current stage costs by the applied control  $u_p^*(n)$ . From that, the spatial movements of the set of robots can be anticipated. Interpreting the performance criteria in practical terms, a lower communication effort would reduce bandwidth utilisation, while shorter simulation times imply faster convergence of the robots to their target states.

For the holonomic case, the kinematic model is given by Example 1 and the stage costs  $\ell_p$  were chosen as

$$\ell_p(z_p, u_p) := \left\| \begin{pmatrix} (x_p - x_p^*)^2 \\ 5(y_p - y_p^*) \end{pmatrix} \right\|^2 + 0.2 \|u_p\|^2.$$

For the non-holonomic case, the kinematic model is outlined in Example 2. Following [32, 33], the stage costs  $\ell_p$  were chosen as

$$\ell_p(z_p, u_p) := \left\| \begin{pmatrix} (x_p - x_p^*)^2 \\ 5(y_p - y_p^*)^2 \\ (\theta_p - \theta_p^*)^2 \end{pmatrix} \right\|^2 + 0.2 \|u_p^2\|^2$$

to ensure convergence of the MPC closed loop without terminal constraints or costs based on the orientation and position of the vehicle. We stress that purely quadratic costs may not be sufficient, see [34]. Note that, in contrast to the holonomic case, the costs  $\ell_p$  depend on the position of the robot and on its orientation.

For both, holonomic and non-holonomic setting, we considered the coefficients and exponents derived in [32, 33] to allow for comparability of the results.

### 5.2. Simulations

For the **holonomic case**, Fig. 3 (left) illustrates the required number of broadcasted tuples  $K$  computed via (27) and Fig. 3 (right) the accumulated costs (Eq. 29).

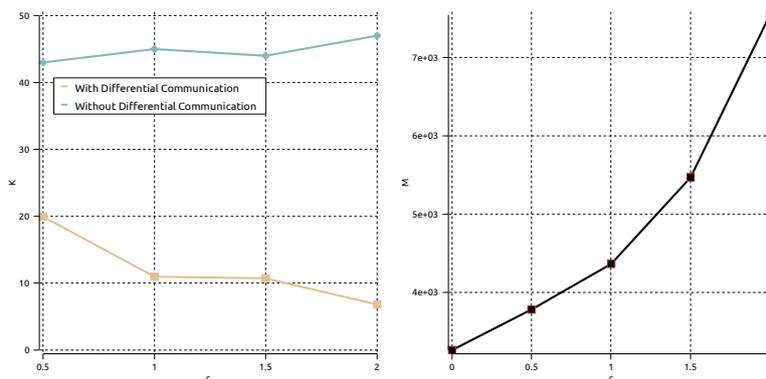


Figure 3: Four holonomic robots ( $N = 12$ ): cell size  $c$  vs. communication load  $K$  (left) and closed-loop costs  $M_p^{n_{\#}}$  vs. cell size  $c$  (right).

By using the differential communication method, we observed that the communication rate was reduced by at least 66% as compared to full communication. Moreover, with the increase of the cell size, we found a decrease in the communication load (Fig. 3 (left)). Of particular interest was the observation that, with larger cell sizes, the occupancy grid was essentially updated by adding one cell for the next time step. It can be noticed that the full communication effort  $K$  increases with larger cell sizes as the simulation time  $n_{\#}$  also increases, which has a direct impact on the amount of broadcasted tuples.

Considering the accumulated closed-loop costs (Fig. 3 (right)), we observed that the costs are directly proportional to the cell size. This connection is due to robots, which have to circumvent other robots resulting in higher costs in case of larger grid cells, see Fig. 4 for an example of the closed-loop trajectories of the robots.

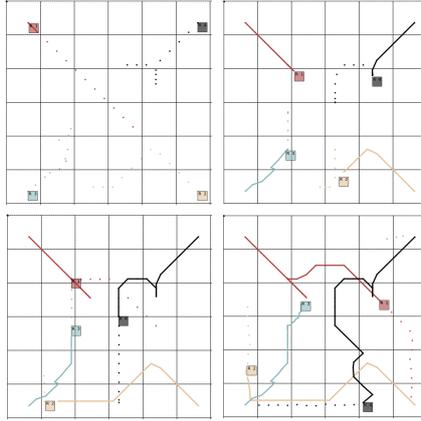


Figure 4: Four holonomic robots: Snap shots of closed-loop behaviour for  $c = 2$  and  $N = 12$  taken at  $n = 9$  (left),  $n = 20$  (middle) and  $n = 31$  (right). Trajectories are presented as continuous lines and predictions as dotted lines.

Additionally, we observed that — at least for the considered setting — in order to achieve convergence increasing the cell size leads to longer prediction horizons, see Table 2 for a summary of this observation. The observation can be explained by the size of blocked areas, which increases with the cell size. To overcome deadlocks, a prediction horizon is needed, which allows to obtain a decrease in the cost. As the latter is equivalent to approaching the target, larger blocked areas require longer prediction horizons to achieve such a reduction.

Table 2: Comparison of cell size  $c$  and prediction horizon  $N$  required to observe convergence, number of closed-loop iterations  $n_{\#}$ , and communication reduction

cell size $c$	$N$	$n_{\#}$	Communication Reduction (%) using differential comm.
0.5	9	33	42.03
1.0	10	34	72.53
1.5	11	41	73.16
2.0	12	46	85.10

As a consequence, with longer prediction horizon and longer simulation

times, the communication effort increased further.

Concerning the costs, the graphs of the accumulated closed-loop costs for all robots, i.e.  $M_P$  (see Eq. (28)), and the graphs of the stage costs for each robot  $\ell_p(z_p^{\text{MPC}}(n), u_p^{\text{MPC}}(n))$ , with  $c = 2.0$ , are shown in Fig. 5. Here, we utilized the classical DMPC approach without quantisation, cf. [9], as a benchmark. The results are depicted in the graph for  $c = 0.0$  in Fig. 5 (left). As it can be noticed, with smaller cell sizes, the performance of the proposed approach becomes closer to the performance of the classical DMPC implementation.

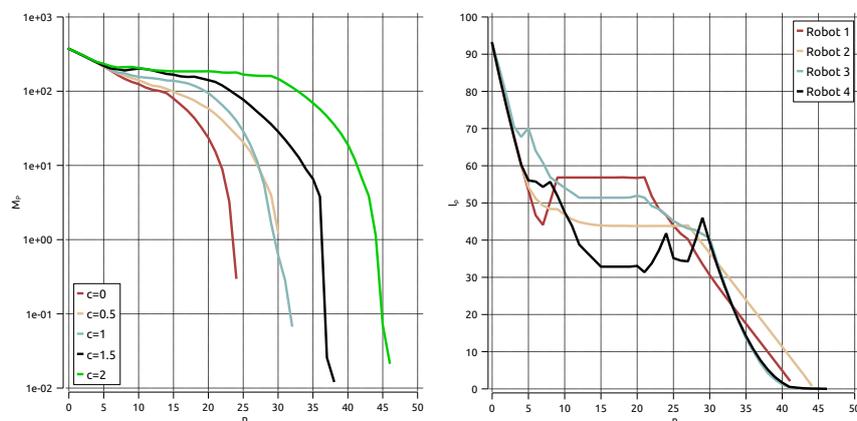


Figure 5: Four holonomic robots: evolution of  $M_P$  with  $c = \{0.0, 0.5, 1.0, 1.5, 2.0\}$  and  $N = 12$  (left). Evolution of  $\ell_p(z_p^{\text{MPC}}(n), u_p^{\text{MPC}}(n))$  for each robot with  $c = 2.0$  and  $N = 12$  (right).

From Fig. 5 (left), we observed that for increased cell size  $c$ , the simulation time  $n_{\#}$  rose and the costs decreased slower due to longer detours around blocked cells. A large difference regarding  $n_{\#}$  occurred for  $c \in \{1.0, 1.5, 2.0\}$ . In contrast to that, for  $c \in \{0.5, 1.0\}$  the development of the accumulated sum and simulation time was much more similar. We like to note that large cell sizes may lead to longer detours, which in turn may result in temporary increases of the stage costs, cf. Fig. 5 (right).

For the **non-holonomic case**, the communication effort (Fig. 6 (left)) increased similarly to the holonomic setting (Fig. 3 (left)) along with the increase in cell size, if we account for the absolute number of communicated tuples.

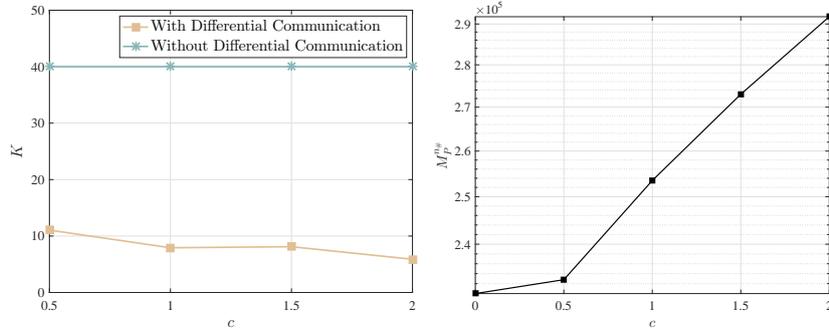


Figure 6: Four non-holonomic robots: cell size  $c$  vs. communication load  $K$  (left) and closed-loop costs  $M_P^{\#}$  (right) for  $N = 9$ .

Here, we observed that the communication effort for small cell sizes  $c \in \{0.5, 1.0\}$  in the full scheme is almost identical. The reason for this similarity is found in Fig. 8 showing an almost identical decrease in the accumulated closed-loop costs  $M_P$  and identical simulation time  $n_{\#}$ . The latter resulted in an analogue number of broadcasted tuples. Similar to the holonomic case, for larger cell sizes we observed that the number of broadcasted tuples increased due to longer simulation times. Considering the differential communication scheme, the number of broadcasted tuples was reduced for a suitable cell size. This may be connected to the dynamics and constraints for the robot as they require more time to cross a cell, which leads to a smaller number of altered cells. In detail, the closed-loop trajectories are illustrated in Fig. 7.

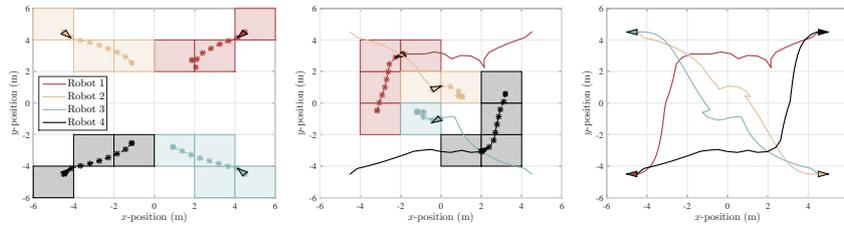


Figure 7: Four non-holonomic robots: Snapshots of closed-loop behaviour for  $c = 2.0$ , and  $N = 9$ , taken at  $n = 8$  (left),  $n = 17$  (middle), and  $n = 41$  (right).

The development of the closed-loop costs are depicted in Fig. 8.

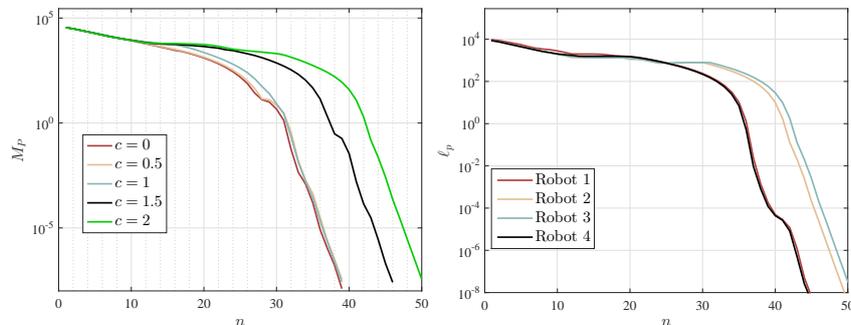


Figure 8: Four non-holonomic robots: development of  $M_P$  with  $c = \{0.0, 0.5, 1.0, 1.5, 2.0\}$  and  $N = 9$  (left) and of  $l_p(z_p^{\text{MPC}}(n), u_p^{\text{MPC}}(n))$  for each robot with  $c = 2.0$  and  $N = 9$  (right).

These results correspond to the ones shown in Fig. 5 for the holonomic robots. Since the non-holonomic kinematic model has less flexibility in changing direction or angular orientation, the simulation time  $n_{\#}$  for the non-holonomic case was longer as for the holonomic case. From the costs (Fig. 8 (right)) of the individual robots, we also observed that stage costs of the robots, which are crossing each other diagonally — here robot 1, 4 and 2, 3 — decreased similarly.

The accumulated costs  $M_P^{n_{\#}}$  (Fig. 6 (right)) rose proportionally with the increase in the cell size  $c$ , which is similar to the holonomic case (Fig. 3 (right)). The non-holonomic equivalent of Table 2 regarding required prediction horizon length to meet Ineq. (26) is presented in Table 3.

Table 3: Comparison of cell size  $c$  and prediction horizon  $N$  required to observe convergence, number of closed-loop iterations  $n_{\#}$ , and communication reduction

cell size $c$	$N$	$n_{\#}$	Communication Reduction (%) using differential comm.
0.5	7	40	70.35
1.0	7	45	78.90
1.5	9	47	79.73
2.0	9	51	85.44

Note that the comparably short prediction horizon lengths are directly connected to the used optimisation method. At least in these numerical simulations, gradient based methods showed better results, but were based on the more conservative definition of coupling constraints via squircles.

### 5.3. Conclusion of the Simulations

The reduced communication load due to quantisation can be highlighted as a relevant advantage of the proposed approach in relation to previous schemes based on the communication of predicted trajectories [9]. Secondly, the quantisation of communication allows us to introduce the differential communication scheme, which reduces the communication requirements depending on the chosen cell size.

As our simulations have shown, increasing cell sizes cause robots to take longer detours, resulting in larger accumulated costs. As this leads to longer simulation times, i.e. slower convergence, the number of broadcasted occupancy tuples increases. Regarding the full communication scheme, this was observed for both holonomic and non-holonomic robots, cf. Fig. 3 and Fig. 6. In contrast to that, the usage of the differential communication scheme reduces the number of occupancy tuples even for longer simulation times. In fact, we observed that the number of broadcasted tuples remains constant for the chosen cell sizes in both settings.

From the individual costs functions, see Fig. 5 and Fig. 8, we can conclude that the decrease of the cost function depends on the optimisation order. The first one is related to the smaller solution spaces of robots, which have to accept occupancy grid predictions of other robots. For the holonomic case, this can be directly seen in Fig. 5 (right), where robot 1 decreased its costs faster than the following robots. In the non-holonomic case (Fig. 8 (right)), we faced the problem of blocking. Here, all robots steered towards the centre of the spatial set. Then, robot 1, which optimised first, blocked robot 2. Additionally, robot 4 had to let robot 3 pass, which then got blocked in the centre of the spatial set as it had to wait for robot 2. Therefore, robots 1 and 4 blocked the inner robots 2 and 3 and reached their target states first.

## 6. Conclusion

In this paper, we proposed a distributed MPC scheme based on communication of occupancy grid predictions for mobile robots. We projected the continuous predicted trajectories onto a grid to obtain occupancy grid predictions described by grid cell indices. These occupancy grid predictions are broadcasted via a differential communication method, which considers only altered cells. Furthermore, we derived sufficient conditions to ensure collision avoidance among the robots utilising the quantised communication. In numerical simulations for both holonomic and non-holonomic robots, we concluded that the differential communication scheme reduces the communication effort depending on the chosen cell size.

To contemplate recursive feasibility by a respective stability proof, we will consider an idea of parking lots similar to tower of Hanoi technique for a worst case analysis. Introducing conditions on target states, this will allow us to generate freeways for robots. Apart from the stability aspect, future considerations are the dynamic variation of the occupancy grid cell size and a dynamic priority rule in accordance to [27].

- [1] M. Hanses, C. Walter, A. Lüder, [Operating articulated objects with force sensitive mobile manipulators](#), in: 20th IEEE Conference on Emerging Technologies Factory Automation, Luxembourg, 2015, pp. 1–4. doi:[10.1109/ETFA.2015.7301579](#).
- [2] H. Zhang, P.N. Pathirana, [Formation control of weak autonomous robots](#), in: Proceedings of the 50th IEEE Conference on Decision and Control (CDC) and European Control Conference (ECC), Orlando, 2011, pp. 4220–4225. doi:[10.1109/CDC.2011.6160878](#).
- [3] M. Bernard, K. Kondak, I. Maza, A. Ollero, [Autonomous transportation and deployment with aerial robots for search and rescue missions](#), *J. Field Robotics* 28 (6) (2011) 914–931. doi:[10.1002/rob.20401](#).
- [4] J. Espinoza L, A. Sánchez L, M. Osorio, [Exploring unknown environments with mobile robots using SRT-Radial](#), in: IEEE International Conference on Intelligent Robots and Systems, IEEE, San Diego, CA, USA, 2007, pp. 2089–2094. doi:[10.1109/IRoS.2007.4399021](#).
- [5] M.W. Mehrez, G.K.I. Mann, R.G. Gosine, [An Optimization Based Approach for Relative Localization and Relative Tracking Control in Multi-Robot Systems](#), *J. Intell. & Robotic Syst.* 85 (2) (2017) 385–408. doi:[10.1007/s10846-016-0408-2](#).
- [6] A. Venkat, J. Rawlings, S. Wright, [Stability and optimality of distributed model predictive control](#), in: Proceedings of the 44th IEEE Conference on Decision and Control, IEEE, Seville, Spain, 2005, pp. 6680–6685. doi:[10.1109/CDC.2005.1583235](#).
- [7] K. Worthmann, C.M. Kellett, P. Braun, L. Grüne, S.R. Weller, [Distributed and Decentralized Control of Residential Energy Systems Incorporating Battery Storage](#), *IEEE Transactions on Smart Grid* 6 (4) (2015) 1914–1923. doi:[10.1109/TSG.2015.2392081](#).
- [8] S.J. Qin, T.A. Badgwell, [A survey of industrial model predictive control technology](#), *Control. Eng. Pract.* 11 (7) (2003) 733–764. doi:[10.1016/S0967-0661\(02\)00186-7](#).
- [9] L. Grüne, K. Worthmann, [A distributed NMPC scheme without stabilizing terminal constraints](#), in: Distributed Decision Making and Control, Springer, London, 2012, pp. 261–287. doi:[10.1007/978-1-4471-2265-4\\_12](#).
- [10] M. Farina, A. Perizzato, R. Scattolini, [Application of distributed predictive control to motion and coordination problems for unicycle autonomous robots](#), *Robotics Auton. Syst.* 72 (2015) 248–260. doi:[10.1016/j.robot.2015.06.007](#).

- [11] W.B. Dunbar, D.S. Caveney, [Distributed receding horizon control of vehicle platoons: Stability and string stability](#), *IEEE Transactions on Autom. Control.* 57 (3) (2012) 620–633. doi:[10.1109/TAC.2011.2159651](https://doi.org/10.1109/TAC.2011.2159651).
- [12] L. Grüne, J. Pannek, K. Worthmann, [A networked unconstrained nonlinear MPC scheme](#), in: Proceedings of the IEEE European Control Conference (ECC), IEEE, Budapest, Hungary, 2009, pp. 371–376.
- [13] L. Grüne, J. Pannek, K. Worthmann, [A prediction based control scheme for networked systems with delays and packet dropouts](#), in: Proceedings of the 48th IEEE Conference on Decision and Control (CDC), IEEE, Shanghai, China, 2009, pp. 537–542. doi:[10.1109/CDC.2009.5399922](https://doi.org/10.1109/CDC.2009.5399922).
- [14] J. Lunze, [Control theory of digitally networked dynamic systems](#), Springer International Publishing, Cham, Switzerland, 2014. doi:[10.1007/978-3-319-01131-8](https://doi.org/10.1007/978-3-319-01131-8).
- [15] M.W. Mehrez, T. Sprodowski, K. Worthmann, G.K.I. Mann, R.G. Gosine, J.K. Sagawa, J. Pannek, [Occupancy Grid based Distributed Model Predictive Control of Mobile Robots](#), in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, Vancouver, Canada, 2017, pp. 4842–4847.
- [16] Y. Pu, M.N. Zeilinger, C.N. Jones, [Quantization design for distributed optimization with time-varying parameters](#), in: Proceedings of the 54th IEEE Conference on Decision and Control (CDC), Osaka, Japan, 2015, pp. 2037–2042. arXiv:[1504.02317v1](https://arxiv.org/abs/1504.02317v1), doi:[10.1109/CDC.2015.7402506](https://doi.org/10.1109/CDC.2015.7402506).
- [17] P. Yi, Y. Hong, [Quantized Sub-gradient Algorithm and Data Rate Analysis for Distributed Optimization](#), *IEEE Transactions on Control. Netw. Syst.* 1 (4) (2014) 380–392. doi:[10.1109/TCNS.2014.2357513](https://doi.org/10.1109/TCNS.2014.2357513).
- [18] S. Liu, L. Xie, D.E. Quevedo, [Event-triggered Quantized Communication Based Distributed Convex Optimization](#), *IEEE Transactions on Control. Netw. Syst.* 5870 (99) (2016) 1–11. doi:[10.1109/TCNS.2016.2585305](https://doi.org/10.1109/TCNS.2016.2585305).
- [19] W.F. Tichy, [Design, Implementation, and Evaluation of a Revision Control System](#), in: Proceedings of the 6th International Conference on Software Engineering, ICSE '82, IEEE Computer Society Press, Los Alamitos, CA, USA, 1982, pp. 58–67.
- [20] B. Collins-Sussman, B.W. Fitzpatrick, M.C. Pilato, [Version Control with Subversion](#), O'Reilly Media, 2004.
- [21] Z. Artstein, [Stabilization with relaxed controls](#), *Nonlinear Analysis: Theory, Methods & Appl.* 7 (11) (1983) 1163–1173. doi:[http://dx.doi.org/10.1016/0362-546X\(83\)90049-4](http://dx.doi.org/10.1016/0362-546X(83)90049-4).

- [22] S. Körkel, H. Qu, G. Rücker, S. Sager, [Derivative Based vs. Derivative Free Optimization Methods for Nonlinear Optimum Experimental Design](#), in: *Current Trends in High Performance Computing and Its Applications: Proceedings of the International Conference on High Performance Computing and Applications*, Springer, Heidelberg, 2005, pp. 339–344. doi:10.1007/3-540-27912-1\_41.
- [23] M. Fernández-Guasti, Analytic geometry of some rectilinear figures, *Int. Jour. Math. Ed. Sci. & Tech.* 23 (6) (1992) 895–901.
- [24] A. Richards, J. How, A decentralized algorithm for robust constrained model predictive control, in: *Proceedings of the American Control Conference (ACC)*, IEEE, Boston, MA, USA, 2004, pp. 4261–4266.
- [25] A. Richards, J.P. How, [Robust distributed model predictive control](#), *Int. J. Control.* 80 (9) (2007) 1517–1531. doi:10.1080/00207170701491070.
- [26] D. Mayne, J. Rawlings, C. Rao, P. Scokaert, Constrained model predictive control: Stability and optimality, *Autom.* 36 (6) (2000) 789–814. doi:10.1016/S0005-1098(99)00214-9.
- [27] J. Pannek, [Parallelizing a state exchange strategy for noncooperative distributed NMPC](#), *Syst. Control. Lett.* 62 (1) (2013) 29–36. doi:10.1016/j.sysconle.2012.10.015.
- [28] M.J.D. Powell, Direct search algorithms for optimization calculations, *Acta Numer.* 7 (1998) 287–336.
- [29] S. Johnson, [The NLOpt nonlinear-optimization package](#) (2004).
- [30] A. Wächter, T.L. Biegler, [On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming](#), *Math. Program.* 106 (1) (2006) 25–57. doi:10.1007/s10107-004-0559-y.
- [31] J. Andersson, A General-Purpose Software Framework for Dynamic Optimization, Phd thesis, Arenberg Doctoral School, KU Leuven (2013).
- [32] K. Worthmann, M.W. Mehrez, M. Zanon, G.K.I. Mann, R.G. Gosine, M. Diehl, [Model Predictive Control of Nonholonomic Mobile Robots Without Stabilizing Constraints and Costs](#), *IEEE Transactions on Control. Syst. Technol.* 24 (4) (2016) 1394–1406. doi:10.1109/TCST.2015.2488589.
- [33] K. Worthmann, M.W. Mehrez, M. Zanon, G.K.I. Mann, R.G. Gosine, M. Diehl, [Regulation of Differential Drive Robots using Continuous Time MPC without Stabilizing Constraints or Costs](#), *IFAC-PapersOnLine* 48 (23) (2015) 129–135. doi:10.1016/j.ifacol.2015.11.272.
- [34] M.A. Müller, K. Worthmann, [Quadratic costs do not always work in MPC](#), *Autom.* 82 (2017) 269–277. doi:10.1016/j.automatica.2017.04.058.

This article is printed under terms of Creative Commons CC-BY-NC-ND  
<https://creativecommons.org/licenses/by-nc-nd/4.0/> with all rights ©Elsevier.  
Original publication is available via <https://doi.org/10.1016/j.ins.2018.04.034>.

## Interval Superposition Arithmetic Inspired Communication for Distributed Model Predictive Control

Tobias Sprodowski<sup>1</sup>, Yanlin Zha<sup>2</sup>, and Jürgen Pannek<sup>1</sup>

<sup>1</sup> University of Bremen, Department of Production Engineering and BIBA Bremer Institut für Produktion und Logistik GmbH, Hochschulring 20, 28359 Bremen, Germany, [sprlpan@biba.uni-bremen.de](mailto:sprlpan@biba.uni-bremen.de)

<sup>2</sup> Shanghai Tech University, School of Information Science and Technology, Shanghai, China, [zhayl@shanghaitech.edu.cn](mailto:zhayl@shanghaitech.edu.cn)

**Abstract.** In this paper, we combine an quantised communication approach for a distributed system consisting of holonomic robots with the set characterization to further reduce the communication load. To ensure collision avoidance among the robots, the trajectory is quantised and incorporated into a distributed model predictive control scheme. Combining this quantised approach with the set characterization to communicate only the lower and upper bound, the communication load is independent of the necessary horizon length while numerical results still show that target states are reached.

### 1 Introduction

Today, robots are suitable to fulfil many tasks as the computational abilities are capable to do also complex calculations online. For many applications, robots either have to collaborate, e.g. formation control [5], or, in non-cooperative scenarios, at least have to avoid collisions. As handling this problem in a centralised manner may be infeasible [9], a distributed control algorithm can be applied to divide the problem into subproblems, which could then be solved by each robot. Distributed Model Predictive Scheme (DMPC) turned out to be an appropriate method, which is able to steer a distributed system to a defined equilibrium [10], see also [1] for a detailed overview of the method. However, if a strong coupling among the subsystems exists, then communication is required [9].

Considering the communication burden in many robotic or automotive applications, wireless communication is used, see, e.g., [4]. Hence, the resources for bandwidth and throughput are limited. To incorporate the latter, we proposed a quantised communication scheme in [6], where optimisation is based on a continuous spatial set but communication is shifted to a quantised space. Here, the spatial set is quantised into a grid of cells and indexes of the occupied cells are used to exchange state information. This scheme allows for an extension to transmit only the altered cells instead of the whole prediction [8].

In this paper, we present an idea, which incorporates the set characterization from existing set-valued arithmetics [11] into the communication scheme. Instead

of using the full prediction of quantised or altered cells, we transmit the lower and upper bound of the prediction of the robot. This leads to a fixed number of exchanged information as we have to transmit only two cells indexes for each time instant. We explore this approach numerically and compare the results to previously established schemes.

The remaining paper is organised as follows: First, we introduce the problem setting presenting the robotic model and give a short definition of the occupancy grid construction in Section 2. In Section 3, we present the set characterization adaptation for the communication among the robots and in Section 4 formulate the adapted DMPC algorithm which is executed by each robot. Before concluding our paper, we will present numerical results by utilising holonomic robots in Section 5.

## 2 Problem Setting

Here, we consider  $P$  mobile robots following a time discrete model

$$z_p^+ = f(z_p, u_p) := f_0(z_p) + \sum_{i=1}^m f_i(z_p) u_{p,i}, \quad p \in [1 : P] \quad (1)$$

with  $z_p = (x_p, y_p)^\top$  as the current state of system  $p$  in the 2D-space,  $f_i$  denoting continuous and smooth vector fields  $f_i : \mathbb{R}^d \rightarrow \mathbb{R}^d$ ,  $i \in \{1, \dots, m\}$  where each vector field is driven by a control variable  $u_{p,i}$  and  $m \in \mathbb{N}$ . Here, we specialise this to a kinematic holonomic model

$$\begin{pmatrix} x_p \\ y_p \end{pmatrix}^+ = \begin{pmatrix} x_p \\ y_p \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \end{pmatrix} v_p^x + \begin{pmatrix} 0 \\ 1 \end{pmatrix} v_p^y, \quad (2)$$

where each vector field is driven by a control variable  $v_p^x$  and  $v_p^y$ . The constraints are bounded by a convex set  $\mathbb{U} \subset \mathbb{R}^2$  containing the origin,

$$\mathbb{U} := \left\{ u_p = \begin{pmatrix} v_p^x \\ v_p^y \end{pmatrix}, \|u_p\| \leq \bar{v} \right\}, \quad \bar{v} > 0 \quad (3)$$

For a given prediction horizon  $N$ , we define a control sequence

$$\mathbf{u}_p = (u_p(0), u_p(1), \dots, u_p(N-1)),$$

which is utilised to describe the predicted state trajectory of a robot via

$$\mathbf{z}_p^{\mathbf{u}}(\cdot; z_p^0) := (z_p^u(0; z_p^0), z_p^u(1; z_p^0), \dots, z_p^u(N; z_p^0)).$$

To establish the quantised communication approach, we project the spatial set  $[-\bar{x}, \bar{x}] \times [-\bar{y}, \bar{y}]$  into a grid  $\mathcal{G} := [0 : a_{\max} - 1] \times [0 : b_{\max} - 1] \subset \mathbb{N}_0^2$  of squared cells such that each cell has a width (and height)  $c$ . Hence, we have

$$2\bar{x} = a_{\max}c \quad \text{and} \quad 2\bar{y} = b_{\max}c, \quad (4)$$

where  $\bar{x}$  and  $\bar{y}$  are both multiples of  $c$ . To quantise the position of a robot, we utilise the following function:

$$q(z_p) = (a_p, b_p) := \left( \left\lfloor \frac{x_p + \bar{x}}{c} \right\rfloor, \left\lfloor \frac{y_p + \bar{y}}{c} \right\rfloor \right), \quad (5)$$

The obtained quantised position can then be used to assemble a predicted trajectory in  $\mathbb{R}^d$  over a horizon length  $N$  into a sequence of occupied cells via

$$\mathcal{I}_p(n) := (n+k, q(z_p^u(k; z_p^0)))_{k=0}^N. \quad (6)$$

If a robot receives all sequences of occupied cells, it can construct the so called *occupancy grid*

$$i_p(n) := (\mathcal{I}_1(n), \dots, \mathcal{I}_{p-1}(n), \mathcal{I}_{p+1}(n), \dots, \mathcal{I}_P(n)). \quad (7)$$

If the states are mapped back to  $\mathbb{R}^d$ , the middle point of a cell is used as approximation for the true position of the robot

$$f_c((n, a, b)) = \underbrace{((a+0.5)c - \bar{x}, (b+0.5)c - \bar{y})^\top}_{=: (x^c, y^c)}. \quad (8)$$

To derive collision avoidance constraints, we define a function  $g_{q,k}^p$  to measure the necessary distance via the infinity norm

$$\begin{aligned} g_{q,k}^p &:= g(z_p^u(k; z_p^0), \mathcal{I}_q(n)(k), \mathcal{I}_q(n)(k-1)) \\ &= \left\| \begin{pmatrix} x_p^u(k; x_p^0) \\ y_p^u(k; y_p^0) \end{pmatrix} - f_c(\mathcal{I}_q(n)(k)) \right\|_\infty \geq 0 \end{aligned} \quad (9)$$

Hence, the coupling constraints are given by

$$G(z_p^u(k; z_p^0), i_p(n)(k), i_p(n)(k-1)) = (g_{1,k}^p, \dots, g_{p-1,k}^p, g_{p+1,k}^p, \dots, g_{P,k}^p) \geq 0. \quad (10)$$

### 3 Set Characterization

Instead of transmitting a sequence of cells (6), we now characterize the set of the positions of the robots as an interval rectangle, which are denoted by  $X_p$  for a robot  $p$ . Then, for the quantised set, we obtain

$$q(X_p) = \{ \min \mathbf{z}_p^u(\cdot; z_p^0), \max \mathbf{z}_p^u(\cdot; z_p^0) \} \quad (11)$$

by utilising the structure of the infinity norm in (9), cf. Figure 1 for a sketch. This allows to reduce communication as only the minimum and maximum cells for each robot are required. To reconstruct then the occupancy grid, Algorithm 1 can be applied.

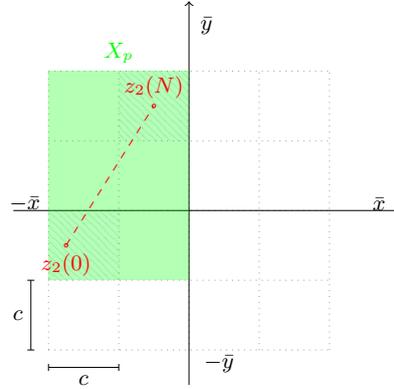


Fig. 1: Example with an occupied set  $X_p$  for robot  $p$   $a_{\max} = b_{\max} = 4$  with dashed cells for quantisation (11) and reconstruction by Alg. 1.

---

**Algorithm 1** Occupancy grid construction for (robot  $p$ ) by received occupancy tuples from robots  $q \in [1 : P] \setminus p$

---

- 1: **Input**  $q(X_q)$
  - 2: **Set**  $a_{\min} = \min_a \{q(X_q)\}$ ,  $a_{\max} = \max_a \{q(X_q)\}$ ,  $b_{\min} = \min_b \{q(X_q)\}$ ,  $b_{\max} = \max_b \{q(X_q)\}$
  - 3: **for**  $a = a_{\min}$  to  $a_{\max}$  **do**
  - 4:   **for**  $b = b_{\min}$  to  $b_{\max}$  **do**
  - 5:     **for**  $k = n$  to  $n + N$  **do**
  - 6:       **Append**  $\mathcal{I}_q(n) := \mathcal{I}_q(n) \cup (k, a, b)$
  - 7:     **end for**
  - 8:   **end for**
  - 9: **end for**
  - 10: **Set**  $i_p(n)$  according to (7)
- 

#### 4 Distributed MPC

Each robot  $p \in [1 : P]$  is equipped with a controller executing the DMPC algorithm as mentioned in [6]. Following the DMPC scheme, each robot solves in every time instant an optimal control problem over a finite horizon  $N$  based on the current state  $z_p^0 = z_p(n)$  and received information  $i_p^0 := i_p(n)$  by minimising a cost function  $\ell_p : Z \times \mathbb{U} \rightarrow \mathbb{R}_{\geq 0}$ , regarding given individual targets  $z_p^*$  and coupling constraints based on (9). This problem reads as follows:

$$\min_{\mathbf{u}_p} J_p^N(\mathbf{u}_p; z_p^0, i_p^0) := \sum_{k=0}^{N-1} \ell_p(z_p^u(k; z_p^0), u_p(k)) \quad (12)$$

subject to

$$\begin{aligned}
z_p^u(k+1; z_p^0) &= f(z_p^u(k; z_p^0), u_p(k)), & k \in [0 : N-1], \\
u_p(k) &\in U, & k \in [0 : N-1], \\
G(z_p^u(k; z_p^0), i_p^0(k), i_p^0(k-1)) &\geq 0, & k \in [1 : N], \\
z_p^u(k; z_p^0) &\in Z, & k \in [1 : N].
\end{aligned}$$

Here, the modifications are concerning the communication. Therefore, the DMPC algorithm itself is given by Algorithm 2.

---

**Algorithm 2** DMPC-algorithm for the overall system

---

```

1: Given feasible initial states  $z_p^0$  for all  $p \in [1 : P]$ 
2: for  $p = 1$  to  $P$  do
3:   for  $k = 1$  to  $N$  do
4:     Construct  $\mathcal{I}_p(n)(k)$  based on  $z_p^0$  by (7)
5:   end for
6:   Construct  $q(X_p)$  by (11)
7: end for
8: Communicate  $q(X_p)$  among all robots for all  $p \in [1 : P]$ .
9: for time instant  $n = 0, 1, \dots$  do
10:  for robot  $p$  from 1 to  $P$  do
11:    Receive  $q(X_q)$  and construct  $\mathcal{I}_q(n)$  based on Alg. 1
12:    Solve OCP (12) and Apply  $u_p^*(0)$ 
13:    Broadcast  $q(X_p)$  based on (11)
14:  end for
15: end for

```

---

Here, we are not using terminal conditions, as this would require a sufficient long prediction horizon to match them, which would be not feasible in a large distributed system. Here, our approach is avoiding terminal conditions based on [2,1]. Regarding feasibility to ensure that the algorithm is not terminating unexpectedly, this is shown in [8].

## 5 Numerical Simulations

For the simulations, we consider a group of  $P = 3$  mobile holonomic robots and set the state constraints  $Z := [-6(m), 6(m)]^2$ . Regarding the controls, we set the bounds as  $\sqrt{(\bar{v}^x)^2 + (\bar{v}^y)^2} \leq 0.5$  (m/s), and  $\bar{v} \leq 0.5$  (m/s) and  $\bar{\omega} \leq 0.5$  (rad/s). Moreover, the minimum distance between the robots  $d_{\min} = 0.5 + \varepsilon$  (m) was used, where  $0 < \varepsilon \ll 1$  denotes a numerical safety margin. We obtained the minimum cell size  $\underline{c} = 0.5$  (m) with  $\underline{c} = \max\{\bar{v}^x, \bar{v}^y\} + \varepsilon = \max\{\bar{v}, \bar{\omega}\} + \varepsilon = 0.5(m) + \varepsilon$  based on (15) in [8]. The cell size was chosen as  $c \in \{0.5, 1.0, 1.5, 2.0\}$ . Simulations were conducted in C++ by utilising the NLOpt-Solver [3] with the gradient-free

COBYLA-Algorithm [7]. The closed-loop costs were defined as

$$\ell_p(z_p, u_p) := \left\| \begin{pmatrix} (x_p - x_p^*)^2 \\ 5(y_p - y_p^*) \end{pmatrix} \right\|^2 + 0.2 \|u_p\|^2 \quad (13)$$

and initial and target positions were chosen as in [8, Table 1]. We investigated the performance of Algorithm 2 in terms of the cumulated closed-loop costs, which we defined for all robots for one time instant  $n$  via

$$M_P(n) := \sum_{p=1}^P \ell_p(z_p^{\text{MPC}}(n), u_p^{\text{MPC}}(n)) \quad (14)$$

where  $u_p^{\text{MPC}}(n)$  the the applied control signal in Step 5) of Algorithm 2 at time instant  $n$  and  $z_p^{\text{MPC}}$  the resulting closed-loop trajectory. Fig. 2 shows the numerical results of both communication schemes. We observe that although

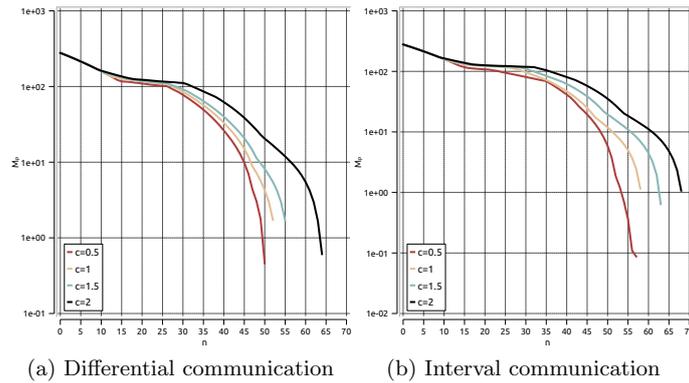


Fig. 2: Development of  $M_P$  with  $c = \{0.5, 1.0, 1.5, 2.0\}$  and  $N = 8$

the limited communication induces less accuracy, the robots are still capable to arrive at their targets. For larger cell sizes, the difference of the simulation times decreases as larger cells attenuate the difference between the interval scheme and the communication of the prediction.

The comparison for the closed-loop costs for both schemes according to (13) is displayed in Fig. 3. The figure shows that the robots are following a straight line as the transmitted cells constitute box constraints. Nevertheless, for robot 1 the arrival time is still the same, although the box constraints lead to larger detours. Therefore, the order of optimisation plays an important role for the arrival time of each robot.

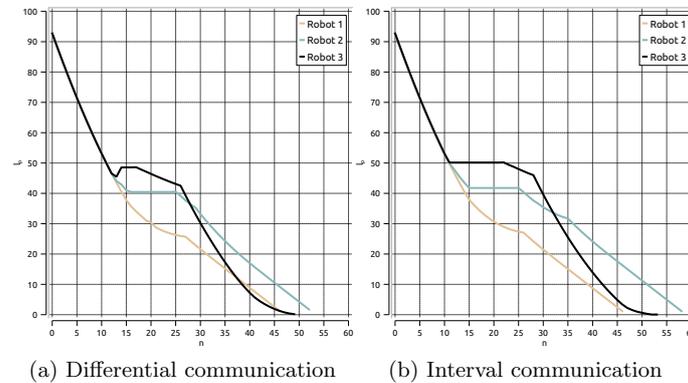


Fig. 3: Development of  $\ell_p(z_p, u_p)$  for  $c = 1.0$  and  $N = 8$

## 6 Conclusions

In this paper, we used a DMPC scheme to steer distributed controlled robots to individual targets while the communication is inspired by an set characterization based quantisation. Here, only the minimum and maximum quantisation cells are transmitted. The numerical results show that the set characterization based quantisation and communication has only slight impact on the simulation time while for all investigated cell sizes the system still converges while the communication load is reduced drastically.

This approach should be validated by utilising a variety of feasible initial and target positions to provide a statistical analysis. Further research should investigate theoretical properties, priority rules and adaptations to traffic scenarios.

## References

1. Grüne, L., Pannek, J.: Nonlinear Model Predictive Control: Theory and Algorithms. Communications and Control Engineering, Springer, London (2017), <http://link.springer.com/book/10.1007/978-3-319-46024-6>
2. Grüne, L., Worthmann, K.: A distributed NMPC scheme without stabilizing terminal constraints. In: Distributed Decision Making and Control, pp. 261–287. Springer, London (2012), [http://dx.doi.org/10.1007/978-1-4471-2265-4\\_12](http://dx.doi.org/10.1007/978-1-4471-2265-4_12)
3. Johnson, S.: The NLOpt nonlinear-optimization package (2004), <http://ab-initio.mit.edu/nlopt>
4. Khaliq, K.A., Qayyum, A., Pannek, J.: Performance Analysis of Proposed Congestion Avoiding Protocol for IEEE 802.11s. International Journal of Advanced Computer Science and Applications 8(2), 356–369 (2017), <http://dx.doi.org/10.14569/IJACSA.2017.080246>
5. Lim, H., Kang, Y., Kim, J., Kim, C.: Formation control of leader following unmanned ground vehicles using nonlinear model predictive control. In: International

- Conference on Advanced Intelligent Mechatronics. pp. 945–950. Singapore, Singapore (2009), <https://doi.org/10.1109/AIM.2009.5229887>
6. Mehrez, M.W., Sprodowski, T., Worthmann, K., Mann, G.K.I., Gosine, R.G., Sagawa, J.K., Pannek, J.: Occupancy Grid based Distributed Model Predictive Control of Mobile Robots. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 4842–4847. IEEE, Vancouver, Canada (2017)
  7. Powell, M.J.D.: Direct search algorithms for optimization calculations. *Acta Numerica* 7, 287–336 (1998)
  8. Sprodowski, T., Mehrez, M.W., Worthmann, K., Mann, G.K.I., Gosine, R.G., Sagawa, J.K., Pannek, J.: Differential Communication with Distributed Model Predictive Control of Mobile Robots based on an Occupancy Grid. submitted (2017)
  9. Venkat, A., Rawlings, J., Wright, S.: Stability and optimality of distributed model predictive control. In: Proceedings of the 44th IEEE Conference on Decision and Control. pp. 6680–6685. IEEE, Seville, Spain (2005), <http://dx.doi.org/10.1109/CDC.2005.1583235>
  10. Xi, Y.G., Li, D.W., Lin, S.: Model Predictive Control — Status and Challenges. *Acta Automatica Sinica* 39(3), 222–236 (2013), [http://dx.doi.org/10.1016/S1874-1029\(13\)60024-5](http://dx.doi.org/10.1016/S1874-1029(13)60024-5)
  11. Zha, Y., Houska, B.: Interval Superposition Arithmetic (October) (2016), <http://arxiv.org/abs/1610.05862>

Reprinted by permission from Springer: Nature, Lecture Notes in Logistics, Interval Superposition Arithmetic Inspired Communication for Distributed Model, Predictive Control, T. Sprodowski, Y. Zha, and J. Pannek, ©Springer 2018

# Relaxed collision constraints based on interval superposition principle in a DMPC scheme

1<sup>st</sup> Tobias Sprodowski

Faculty for Production Engineering  
University of Bremen  
28359 Bremen, Germany  
spr@biba.uni-bremen.de

2<sup>nd</sup> Jürgen Pannek

BIBA, Bremer Institut für Produktion und Logistik  
University of Bremen  
28359 Bremen, Germany  
pan@biba.uni-bremen.de

**Abstract**—Within this paper, we consider a group of mobile robots in a shared operation space controlled by a distributed model predictive control scheme, where each robot is assigned with individual start and target positions. Each robot solves an Optimal Control Problem, based on information about the predictions of the other robots, to calculate an optimal trajectory over a fixed horizon length. The information exchange among the robots is quantised by projecting the trajectory onto a grid. To reduce the communication overhead, the interval superposition principle is used by transmitting only the minimum and maximum coordinates of the trajectory, yielding a static interval of reserved cells. To attenuate the effect of many reserved cells, the dynamics and directional movement of the robot is incorporated to determine, which of the reserved cells are not needed by this robot anymore. This additional information can be transmitted without necessary overhead. Furthermore, the time instant can be excluded from the communicated cells. To this end, numerical simulations based on holonomic dynamics of mobile robots are presented and compared with the conservative approach.

**Index Terms**—Distributed Model Predictive Control, Robotics, Non-cooperative control

Today, most coordination problems of mobile robots or vehicles are established as a distributed system to achieve real-time availability of solutions or to decrease complexity. The problem is solved by each robot with respect to a given objective [1], e.g. measuring the distance to the given target, and given constraints, e.g. ensuring collision avoidance constraints or obey model restrictions as control limits. Since no information about the global goal or costs from the other robots is available to one individual robot, the control of such a distributed system can be classified as non-cooperative [2]. As [3] and [4] point out, an information exchange among the robots is necessary to steer the overall system to a desired equilibrium, i.e. for the robots to reach their target. As most distributed systems need to construct a future plan to establish constraints for other subsystems and a feedback to cope with model uncertainties, the Distributed Model Predictive Control (DMPC) scheme has received attention in the field of mobile robots by [5] or simulated vehicles by [6]. In the first step of this scheme each robot solves an optimal control problem based on a current state measurement and received information of the other robots over a finite horizon to obtain an optimal control sequence. Within this problem, coupling constraints can be derived from the received information of the other robots. Then, the first element of the control sequence is

applied, and the predicted trajectory is broadcasted to the other robots. After shifting the prediction horizon, the scheme is applied iteratively until each robot achieves its individual given target. As in the robotic setting mostly wireless networks are considered for information exchange, communication delays, packet loss or channel fading have to be considered. One possibility to mitigate these issues is the reduction of communicational burden. Therefore, event-triggered systems are proposed by [7] or quantisation approaches by [8] to reduce either the size of the communicated data (i.e. bit size of words) or to establish a quantised measurement for the state, which is suitable for systems incorporating a slow-changing dynamics, cf. [9]. Other robust-related approaches are inducing intervals to incorporate model uncertainties [10] or use contracts among the subsystem to guarantee that a system stays in a self-defined state interval [11]. To circumvent packet drops or delays, some approaches use the availability of the optimal control sequence over the finite horizon and the prediction consistency (see [12] and [13]), i.e. the prediction of two subsequent time instants are mostly similar. If parts of the sequence are lost during transmission, the last received control sequence can be utilised instead.

In our scenario, we focus on mobile robots using a shared space and information exchange to avoid collisions between each other. Application scenarios could be also considered in autonomous connected vehicles crossing in intersections or robots exploring a shared environment. Here, we focus on the information exchange and the communication burden, which is typically necessary to keep such a distributed system running, i.e. that all robots are able to reach their target. For simplicity, the states are only constrained by the given 2D-plane. Further restrictions, induced by recognised obstacles may be incorporated into the constraints, but are not considered explicitly. To reduce the communication burden, in previous work we applied a quantisation scheme to project trajectories onto a quantised grid with equidistant cells (see [14]), shared among all robots. The robots, solving the OCP in a sequential manner, map the prediction to a sequence of cell indexes, which consists of tuples incorporating the time instant and the cell indexes (*occupancy tuples*). The other robots are able to derive collision avoidance constraints from the received occupancy tuples. In comparison to the plain broadcast of the

exact predicted locations, which require floating point word length, only integer word length is required here. Moreover, instead of communicating all occupancy tuples, the Interval Superposition Arithmetic principle [10] was used to communicate the minimum and maximum cell by considering the 2D-plane coordinates of the assigned trajectory only [15]. Other approaches were cellular automata using Voronoi diagrams to incorporate the time evolution of cell automata [16]. To attenuate the localisation error of the robot inside the cells, the authors of [17] used an online correction step to calibrate the robot after a number of taken steps with regard to orientation and position separately. Other heuristic searches use the A\*-algorithm [18]. All these discrete algorithms do not include the kinematic of the robots in the movement and most implementations rely on full available information among all robots. The contribution in this paper is twofold: First, we focus on reducing communication based on the interval superposition principle and dropping the time stamps. The idea is to keep the received information from the other robots as long as no new information has been received to update the information. Secondly, as the cell indexes are positive, we can use the leading sign to indicate start and end positions of a robot inside the interval, here a rectangle, spanning over the whole prediction horizon. This information allows to reduce the number of blocked cells as the robots have to move according to their planned trajectory starting from the initial position at time instant  $n$  to the end position at  $n + N$ . The movement of each robot can be estimated due to the known dynamics and equal sampling steps. Here, packet losses or delays due to unreliable wireless communication are not considered but may be integrated using the immediate hold property [14, Assumption 1] and methods using the available information [12]. The paper is structured as follows: First, we recap the definition of the problem setting including the quantisation and define the information exchange including the evaluation of the directions of the robots to introduce the relaxation of the intervals. Then, we compose the Optimal Control Problem, which is then resembled in the DMPC algorithm. Simulation results are presented, before conclusion and future aspects are drawn.

**Notation:**  $\mathbb{R}$  and  $\mathbb{N}$  denote the real and natural numbers, respectively.  $\mathbb{N}_0 := \mathbb{N} \cup \{0\}$  represents the non-negative integers and  $\mathbb{R}_{\geq 0}$  the non-negative real numbers. We denote for integers  $a, b \in \mathbb{N}_0$  with  $a \leq b$  the expression  $[a : b]$ , abbreviating the set  $\{a, a + 1, \dots, b\}$ . Sequences of tuples are denoted as  $(C)_{k=0}^N$  for  $C := c_0, \dots, c_N$ .

## I. PROBLEM SETTING

We consider a number of  $P$  mobile robots defined by a constrained state  $z_p \in Z := [-x_{max}, x_{max}] \times [-y_{max}, y_{max}] \times \tilde{Z} \in \mathbb{R}^{d-2}$  with  $d \in \mathbb{N}$  and  $d \geq 2$ , allowing for holonomic or non-holonomic robots. Each robot applies the control  $u_{p,i}$  with  $i \in [1 : m]$ , which is bounded by  $u_p \in U \subset \mathbb{R}^m$ . The movement of the robots is described by the abstract dynamics

$f$ , where

$$z_p^+ = f(z_p, u_p) := f_0(z_p) + \sum_{i=1}^m f_i(z_p) u_{p,i}, \quad p \in [1 : P] \quad (1)$$

belongs to the class of time-discrete control affine systems [19]. Given a feasible initial point  $z_p^0 \in Z$ , we can define a discrete trajectory for a robot  $p$  over a finite horizon  $N$  with

$$z_p^u(\cdot; z_p^0) := (z_p^u(0; z_p^0), z_p^u(1; z_p^0), \dots, z_p^u(N; z_p^0)), \quad (2)$$

where the imposed control is defined as a finite control sequence

$$u_p = (u_p(0), u_p(1), \dots, u_p(N-1)).$$

We consider a quantisation of the state space to be given by  $\mathcal{G} := [1, x_{max}] \times [1, y_{max}]$ , where each cell is equidistant with size  $c$  and

$$2x_{max} = a_{max}c \quad \text{and} \quad 2y_{max} = b_{max}c. \quad (3)$$

To quantise a given trajectory, the mapping function

$$q(z_p) = (a_p, b_p) := \left( \left\lfloor \frac{x_p + x_{max}}{c} \right\rfloor, \left\lfloor \frac{y_p + y_{max}}{c} \right\rfloor \right), \quad (4)$$

allows to convert that trajectory into a sequence of cell indexes. In [14], we used this quantisation to generate *occupancy tuples*, which resemble the received information from other robots. Here, the aim was to reduce the word size of the transmitted information (integer instead of doubles). The transmitted information can be reduced further by incorporating the interval superposition principle from [10], where only the interval of the minimum and maximum cells are sent. As a consequence, a rectangle of cells can be reconstructed by the receiving robot, which we denote by  $X_p$  and its quantisation is given by

$$X_p = \left\{ \min_{(a_p, b_p)} \left\{ q(z_p^u(\cdot; z_p^k))_{k=0}^N \right\}, \max_{(a_p, b_p)} \left\{ q(z_p^u(\cdot; z_p^0))_{k=0}^N \right\} \right\}. \quad (5)$$

Here, we assume that both the prediction horizon  $N$  and sampling time  $T$  are identical for all robots. As the optimisation is carried out in the continuous state space, the received tuple is mapped back accordingly using

$$f_c((a, b)) = \underbrace{((a + 0.5)c - x_{max}, (b + 0.5)c - y_{max})}_{=:(x^c, y^c)}^\top. \quad (6)$$

While in [15, Algorithm 1], the interval was blocked for the full horizon length  $N$ , cf. Fig. 1, this requirement can be relaxed by additional information on the direction of the robots without inducing additional communication. Since cell indexes are defined for  $\mathbb{N}_{>0}$ , we may utilise positive or negative

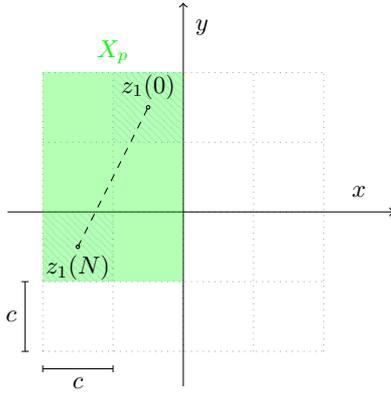


Fig. 1. Reconstruction of reserved cells (green cells) by Alg. 1 from [15] based on received  $X_p$  (dashed cells).

leading signs for the transmitted cell indexes to encode the direction of the robot via

$$X_p = \left\{ \underbrace{(\pm a_1, \pm b_1)}_{\text{direction}}, (a_2, b_2) \right\}.$$

Here, the combinations  $(+, +)$ ,  $(-, -)$ ,  $(+, -)$ ,  $(-, +)$  refer to direction "right top", "left bottom", "right bottom" and "left top" respectively, cf. Fig. 2 for an illustration.

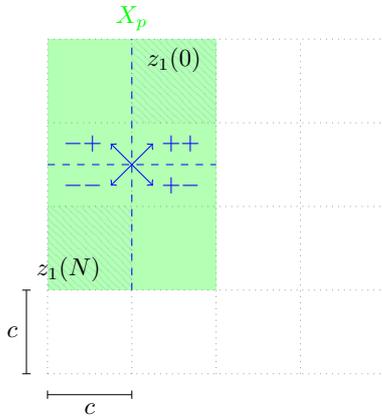


Fig. 2. Illustration of usage of signs for  $(a, b)$ , depending on the direction, e.g. here  $((-a_1, -b_1), (a_2, b_2))$

As a robot claims these cells in accordance with its calculated and feasible solution of the local optimal control problem, we can conclude that the robot is able to reach its target point at  $n+N$ . Hence, from the received tuple  $X_q$  robot  $p$  is able to construct the initial and target points  $\tilde{z}_q(1)$  and  $\tilde{z}_q(2)$  via Algorithm 1.

Note that within Algorithm 1 we utilised the  $sgn(\cdot)$  function componentwise and have to compensate for the directional usage of the signs in the first tuple to obtain correct cells.

**Algorithm 1** Direction based construction of start and target position of robot  $q$

---

**Require:**  $X_q$

- 1: **if**  $sgn(X_q(1)) = (1, 1)^\top$  **then**
- 2:      $\tilde{z}_q := X_q$
- 3: **else if**  $sgn(X_q(1)) = (-1, -1)^\top$  **then**
- 4:      $\tilde{z}_q(1) := X_q(2)$
- 5:      $\tilde{z}_q(2) := -X_q(1)$
- 6: **else if**  $sgn(X_q(1)) = (1, -1)^\top$  **then**
- 7:      $\tilde{z}_q(1) := (X_q(1)(1), X_q(2)(2))$
- 8:      $\tilde{z}_q(2) := (X_q(2)(1), -X_q(1)(2))$
- 9: **else if**  $sgn(X_q(1)) = (-1, 1)^\top$  **then**
- 10:      $\tilde{z}_q(1) := (X_q(2)(1), X_q(1)(2))$
- 11:      $\tilde{z}_q(2) := (-X_q(1)(1), X_q(2)(2))$
- 12: **end if**
- 13: **return**  $\tilde{z}_q$

---

The knowledge of the initial  $\tilde{z}_q(n) = f_c(\tilde{z}_q(1))$  and predicted target point  $\tilde{z}_q(n+N) = f_c(\tilde{z}_q(2))$  is not sufficient to relax the blockage of cells imposed by  $X_q$ . Nevertheless, the information allows to compute the minimum time  $k_{\min}$  for robot  $q$ , i.e. the robot needs to reach its target point  $\tilde{z}_q(n+N)$  from any arbitrary cell  $(i, j)$  inside its reserved set, i.e.  $X_q(1) \leq (i, j) \leq X_q(2)$ . This holds for the assumption that all robots have a common knowledge about the dynamics and the possible maximum speed. The computation of  $k_{\min}$  may be solved as a linear problem for a holonomic model as used in [14]. To this end, for a given occupancy tuple  $(k, i, j)$  with  $k \in [n : n+N]$ ,  $i \in [i_{\min} : i_{\max}]$  and  $j \in [j_{\min} : j_{\max}]$ , if the condition

$$\begin{aligned} \exists \tilde{z}_q \text{ with } (i, j) = q(\tilde{z}_q) : \exists u \in U : \\ \tilde{z}_q(2) := q(\tilde{z}_q^u(k + k_{\min}; \tilde{z}_q)) \end{aligned} \quad (7)$$

is satisfied, robot  $p$  may still be in cell  $(k, i, j)$  or will traverse it. Therefore, using  $k_{\min}$ , calculating  $k_l$  with

$$k_l = N - k_{\min} \quad (8)$$

represents the latest time instant at which the robot has to start moving from cell  $(i, j)$  in order to reach the predicted target point at time instant  $n+N$ . Hence, for each cell satisfying (7) where a  $k_l$  exists with  $n \leq k_l \leq n+N$  has to be blocked. With this approach the blockage of cells induced by the occupancy tuples may be relaxed with Algorithm 2 using (8) to evaluate a latest start time for each examined cell when the robot has to leave to reach the target point.

In Algorithm 2 the occupancy grid tuples are utilizing (7) depending on the time instant at which robot  $q$  has to leave the initial cell under consideration to reach the end cell  $\tilde{z}_q(2)$  at time instant  $n+N$ . The still occupied cells are denoted as a *reachable set* in the given time horizon. If no connecting trajectory exists, then the constraint can be relaxed, respectively, cf. Fig. 3 and 4 for an illustration.

---

**Algorithm 2** Occupancy grid construction for (robot  $p$ ) by received interval tuples  $X_q$  from robots  $q \in [1 : P] \setminus p$ 


---

```

1: Input  $X_q, n$  and  $N$ 
2: Compute  $\tilde{\mathcal{I}}_q$  from Alg. 1 and  $k_l$ 
3: Set  $i_{\min} := \text{sgn}(X_q(1)(1))X_q(1)(1), i_{\max} := X_q(2)(1)$ 
4: Set  $j_{\min} := \text{sgn}(X_q(1)(2))X_q(1)(2), j_{\max} := X_q(2)(2)$ 
5: for  $k = n$  to  $n + N$  do
6:   for  $i = i_{\min}$  to  $i_{\max}$  do
7:     for  $j = j_{\min}$  to  $j_{\max}$  do
8:       if condition (7) holds then
9:         Append  $\mathcal{I}_q(n) := \mathcal{I}_q(n) \cup (k, i, j)$ 
10:      end if
11:    end for
12:  end for
13: end for
14: return  $\mathcal{I}_q(n)$ 

```

---

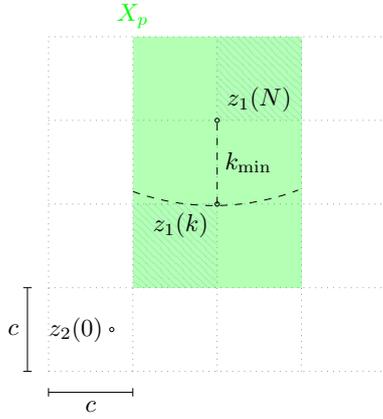


Fig. 3. Case  $k < k_l$ : robot 1 has not left its start position yet, hence, robot 2 has to avoid all green cells

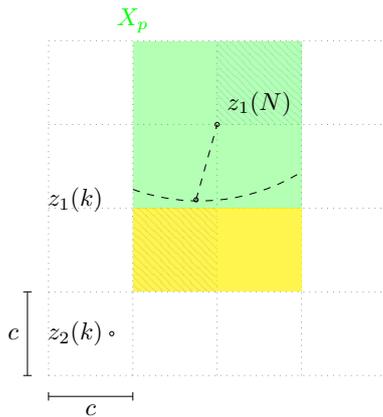


Fig. 4. Case  $k > k_l$ : robot 1 has left its start position. Yellow cells may be occupied by other robots, e.g. by 2.

The coupling constraints are then constructed via the calculated occupancy tuples with

$$\begin{aligned}
g_{q,k}^p &:= g(z_p^u(k; z_p^0), \mathcal{I}_q(n)(k), \mathcal{I}_q(n)(k-1)) \\
&= \left\| \begin{pmatrix} x_p^u(k; x_p^0) \\ y_p^u(k; y_p^0) \end{pmatrix} - f_c(\mathcal{I}_q(n)(k)) \right\|_{\infty} - d_{\min} \geq 0, \quad (9)
\end{aligned}$$

where  $\mathcal{I}_q(n)(k)$  denotes the  $k$ -th tuple for  $\mathcal{I}_q(n)$  and  $d_{\min} \in \mathbb{R}$  a minimum safety margin among the robots as each robot requires a certain diameter to obey collision avoidance constraints. If the robot receives the broadcasted information from  $q \in [1 : P] \setminus p$ , the occupancy tuples are assembled to the *occupancy grid*

$$i_p(n) := (\mathcal{I}_1(n), \dots, \mathcal{I}_{p-1}(n), \mathcal{I}_{p+1}(n), \dots, \mathcal{I}_P(n)). \quad (10)$$

The coupled constraints based on the received occupancy grid and defined in (9) are then summarised by

$$\begin{aligned}
G(z_p^u(k; z_p^0), i_p(n)(k), i_p(n)(k-1)) = \\
\left( g_{1,k}^p, \dots, g_{p-1,k}^p, g_{p+1,k}^p, \dots, g_{P,k}^p \right) \geq 0. \quad (11)
\end{aligned}$$

## II. DMPC WITH MOVING CONSTRAINTS

Based on the constructed constraints, we construct the local optimal control problems (OCP) as a non-cooperative control problem for each robot  $p \in [1 : P]$ . Given an individual start  $z_p^0$  and target position  $z_p^*$ , such that an objective function  $\ell_p : Z \times U \rightarrow \mathbb{R}_{\geq 0}$   $z_p^*$  is minimised over a finite horizon  $N$ :

$$\min_{\mathbf{u}_p} J_p^N(\mathbf{u}_p; z_p^0, i_p^0) := \sum_{k=0}^{N-1} \ell_p(z_p^u(k; z_p^0), u_p(k))$$

subject to

$$\begin{aligned}
z_p^u(k+1; z_p^0) &= f(z_p^u(k; z_p^0), u_p(k)), \\
u_p(k) &\in U, \quad k \in [0 : N-1], \\
G(z_p^u(k; z_p^0), i_p^0(k), i_p^0(k-1)) &\geq 0, \\
z_p^u(k; z_p^0) &\in Z, \quad k \in [1 : N]. \quad (12)
\end{aligned}$$

Each robot takes into account the received intervals from the other robots to calculate a feasible control sequence  $\mathbf{u}_p^* := (u_p^*(0), \dots, u_p^*(N-1))$  regarding the collision avoidance constraints. The DMPC algorithm, which is then executed for each individual robot separately, is formulated in Algorithm 3. The DMPC algorithm starts with feasible states for each robot. Upon initialization, all robots hold their respective start position over the full horizon length  $N$ . Then, the interval is constructed and broadcasted to all other robots. After the initialisation, the robots receive the interval and construct the relaxed coupling constraints according to Algorithm 2. Each robot is then executing the procedure sequentially from line 7, until the target condition is matched. Now, recursive feasibility of Algorithm 3, i.e. that the algorithm finds a feasible solution in any time instant and does not terminate unexpectedly, follows by the same arguments as in [14, Theorem 1].

**Algorithm 3** DMPC-algorithm for one robot  $p \in [1 : P]$ 

- 
- 1: **Given** feasible initial state  $z_p^0$  for  $p \in [1 : P]$ ,  $n := 0$
  - 2: **for**  $k = 1$  to  $N$  **do**
  - 3:     **Set**  $\mathcal{I}_p(0)(k) := (k, q(z_p^0))$
  - 4:     **Construct**  $q(X_p)$  by (5)
  - 5: **end for**
  - 6: **Broadcast**  $q(X_p)$  among all robots for all  $p \in [1 : P]$ .
  - 7: **while**  $\|z_p(n) - z_p^*\| > \delta$  **do**
  - 8:     **Receive**  $q(X_q)$  and construct  $\mathcal{I}_q(n)$  via Alg. 2
  - 9:     **Solve** OCP (12) and **apply**  $u_p^*(0)$
  - 10:    **Broadcast**  $q(X_p)$  based on (5)
  - 11:    **Set**  $n := n + 1$
  - 12: **end while**
- 

## III. SIMULATIONS

For comparison reasons, we used an identical setup as in [15] with a group of  $P = 3$  robots following the kinematic model  $\begin{pmatrix} x_p \\ y_p \end{pmatrix}^+ = \begin{pmatrix} x_p \\ y_p \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \end{pmatrix} v_p^x + \begin{pmatrix} 0 \\ 1 \end{pmatrix} v_p^y$ . State constraints were set as  $Z := [-6(m), 6(m)]^2$  and the control is bounded by  $\sqrt{(v^x)^2 + (v^y)^2} \leq 0.5(m/s)$  with  $-0.5 \leq v \leq 0.5(m/s)$ . A minimum safety distance among the robots was imposed with  $d_{\min} = 0.5 + \varepsilon(m)$ , where  $\varepsilon$  is denoted as a safety margin with  $0 < \varepsilon \ll 1$ . The minimum cell size  $\underline{c}$  was obtained according to [14] with  $\underline{c} = \max\{v^x, v^y, d_{\min}\} = 0.5$  and we utilised the cell sizes  $c = \{0.5, 1.0, 1.5, 2.0\}$ . The simulations were carried out in C++ using the NLOpt-Solver [20] with the gradient-free Algorithm COBYLA based on [21]. The performance criteria were based firstly on the closed-loop costs

$$\ell_p(z_p, u_p) := \left\| \begin{pmatrix} (x_p - x_p^*)^2 \\ 5(y_p - y_p^*) \end{pmatrix} \right\|^2 + 0.2 \|u_p\|^2, \quad (13)$$

where  $z_p^* := (x_p^*, y_p^*)^\top$  denotes the target condition. Secondly, we evaluated the cumulated closed-loop costs over all robots for each time instant  $n$

$$M_P(n) := \sum_{p=1}^P \ell_p(z_p^{\text{MPC}}(n), u_p^{\text{MPC}}(n)), \quad (14)$$

where  $z^{\text{MPC}}$  and  $u^{\text{MPC}}$  denote the closed-loop trajectory and the applied control respectively. We compare the presented approach with moving interval constraints with the conservative one using (5).

To this end, we compare the consumed space of both approaches in terms of assigned cells of all robot over each time instant. For the conservative approach the cells are blocked over the full prediction horizon. As this yields a rectangle,

this is defined by

$$c_c(n) := \sum_{q=1}^P \left( \frac{\overbrace{|X_q(2)(1) - X_q(1)(1)|}^{a_2} + c}{c} \right) \left( \frac{\overbrace{|X_q(2)(2) - X_q(1)(2)|}^{b_2} + c}{c} \right) N$$

giving the number of reserved cells adopting the mapping function (4) and regarding the blocked cells over the full prediction horizon length. In the introduced approach the number of cells evaluated by Alg. 2 is varying due to the relaxed constraints given by  $|\mathcal{I}_q(n)|$ , hence, the difference is calculated by

$$c_l(n) = c_c(n) - \sum_{q=1}^P |\mathcal{I}_q(n)|. \quad (15)$$

In Fig. 5 the cumulated closed-loop costs  $M_P$  are depicted for cell sizes  $c = \{0.5, 1.0, 1.5, 2.0\}$  using the introduced approach. The closed-loop costs for cell size  $c = 2.0$  are

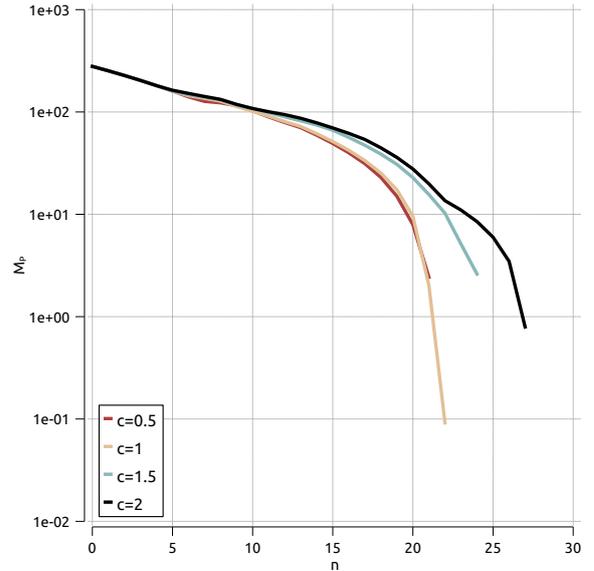


Fig. 5. Time evolution of  $M_P$  with  $c = \{0.5, 1.0, 1.5, 2.0\}$  and  $N = 8$

presented in Fig. 6. Comparing these results with the previous, more conservative approach using superposition without relaxation, see [15, Figure 3], also given in summary table below (see Table I), shows that the convergence time decreases significantly for all cell sizes. Moreover, the robots have to take less detours and reach their targets earlier, which is reflected by the reduction of costs. Considering that with the decreasing intervals the blockages are reduced further, the robots have to take less detours and in turn also do not block big regions for other robots. In detail, in Fig. 6 this is illustrated in comparison

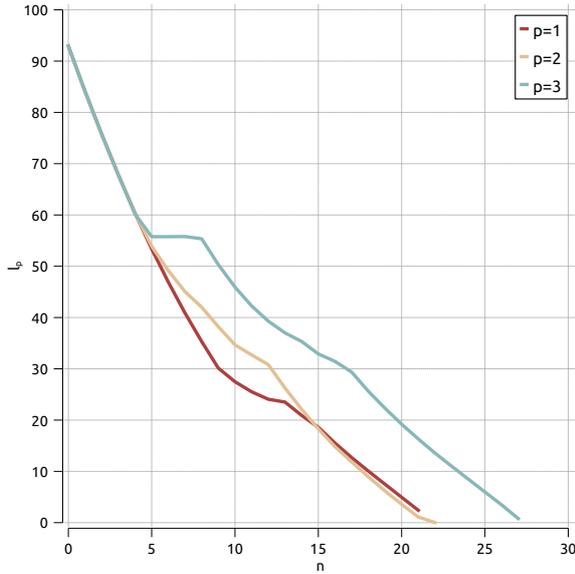


Fig. 6. Development of  $\ell_p(z_p^{\text{MPC}}(n), w_p^{\text{MPC}}(n))$  evolution of the closed-loop costs for  $c = 2.0$ .

TABLE I  
CONVERGENCE TIMES OF THE CONSERVATIVE APPROACH

Cell size ( $M_P(n)$ )	Convergence time
0.5	57
1.0	58
1.5	62
2.0	67

to [15, Fig.3(b)], where the taken detours of the individual robots are longer, resulting in longer periods of waiting times.

Fig. 7 illustrates the development of the maximum leaving time  $k_l$  for cell size  $c = 2.0$  for each robot. For each time instant, over the full horizon length  $N = 8$ , the graph shows for each robot, when it has to leave to reach the considered intermediate target cell. E.g. in time instant 0, for robot 1 full speed is imposed to reach its predicted target point as no collision avoidance constraints are active from other robots, therefore the maximum leaving time is  $k_l = 0$ , which means, that it has to leave immediately. Robot 2 and 3 have to take into account the collision avoidance constraints of robot 1 and have to take detours or slow down. Hence, their maximum leaving time is  $k_l = 2$  for robot 2 and  $k_l = 5$  for robot 3, respectively. At the end of the simulation, as the robots are reaching the target, the maximum leaving time is higher. Considering a non-holonomic robotic model approach, which was used in [14], may lead to earlier leave times as orientational adjustments have to be taken into account and therefore extend the necessary movement. Nevertheless, the holonomic model covers the lowest bound, as the leave times are always later then with any other non-holonomic model to match condition (7). The leave times, which have not to obey

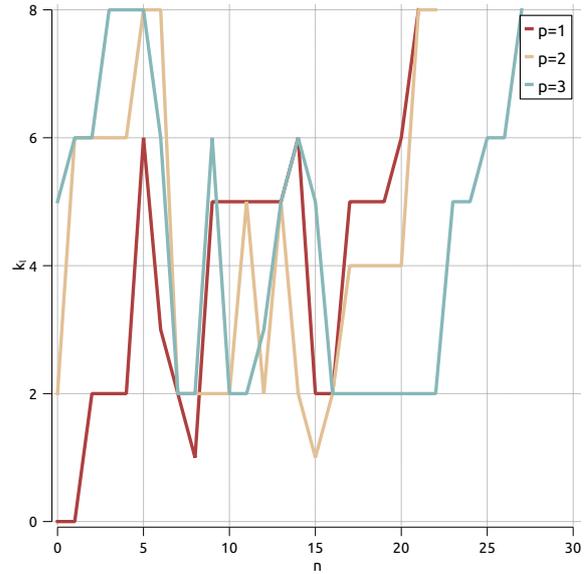


Fig. 7. Maximum leaving time  $k_l$  for  $c = 2.0$

orientation or directional restricted movements like cars, may be more conservative.

To this end, to compare the space consumption of the conservative and moving constraints approach (15), the results are presented in Fig. 8. In the beginning, as the difference is

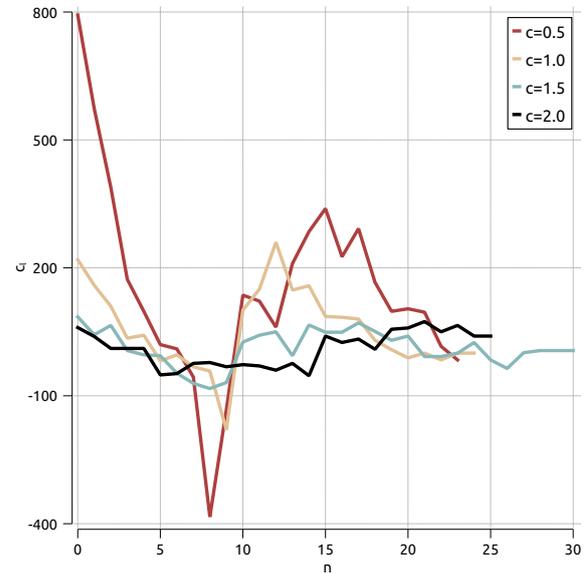


Fig. 8. Time evolution of the space consumption illustrated by the difference  $c_l$  over all cell sizes

positive especially for small cell sizes, the moving constraints approach have an advantage over the conservative one. As in the first time instants the robots are able to move as fast as possible because no collision avoidance constraints are active,

the cells are unblocked quickly. When the robots meet in the centre of the operational set and have to face detours or waiting times due to collision avoidance constraints, the conservative approach gains counterintuitively an advantage over the moving constraint approach. As robot 2 and 3 are not able to reserve reasonable cells, this leads to a waiting period until robot 1 has passed (Fig. 9) and therefore to less consumed space. Considering the moving constraint approach,

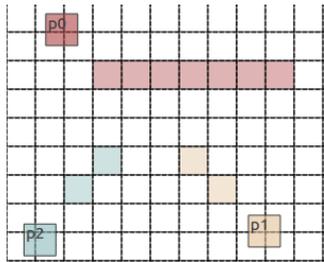


Fig. 9. Screenshot of reserved cells by time instant  $n = 7$  for the conservative approach, robot 1 (red), 2 and 3 in beige and blue, respectively.

the number of available cells is higher due to earlier releases of blocked cells. Therefore, robot 2 and 3 are able to reserve more cells. As with decreasing cell sizes the robots have less cells available to reserve in both approaches, the difference between the two approaches shrinks. The simulations show that the moving constraints approach leads to shorter convergence times, although in some configurations more space is consumed.

#### IV. CONCLUSIONS

In this paper, we demonstrated the usage of reduced communication inside a DMPC scheme for a distributed system of mobile robots where each robot solves an OCP over a finite time horizon. While the optimisation is conducted in the continuous space, the communication is utilising cell indexes by projecting the space onto a grid yielding equidistant cells. The communication is further reduced by utilising the Interval Superposition Principle and to communicate the minimum and maximum cell only. Yet, as this approach was conservative with regards to the blocked regions in state space over the full prediction horizon length, we attenuate the high number of blocked cells by utilising information regarding the maximum speed and the dynamics of the robots to release blocked cells not further needed by the claiming robot. Our numerical results show that the convergence time is significantly reduced and therefore combines the advantages of reduction of communication and less used space in comparison to the previous full blockage approach using interval communication principle. Put all in past tense.

Further consideration will address relaxations considering more complex models (tricycle, car) and furthermore evaluate theoretical properties considering stability aspects.

#### REFERENCES

- [1] M. W. Mehrez, G. K. I. Mann, and R. G. Gosine, "An Optimization Based Approach for Relative Localization and Relative Tracking Control in Multi-Robot Systems," *Journal of Intelligent & Robotic Systems*, vol. 85, no. 2, pp. 385–408, 2017. [Online]. Available: <http://dx.doi.org/10.1007/s10846-016-0408-2>
- [2] A. Venkat, J. Rawlings, and S. Wright, "Stability and optimality of distributed model predictive control," in *44th IEEE Conference on Decision and Control*. Seville, Spain: IEEE, 2005, pp. 6680–6685. [Online]. Available: <http://dx.doi.org/10.1109/CDC.2005.1583235>
- [3] B. T. Stewart, A. N. Venkat, J. B. Rawlings, S. J. Wright, and G. Pannocchia, "Cooperative distributed model predictive control," *Systems & Control Letters*, vol. 59, no. 8, pp. 460–469, 2010. [Online]. Available: <http://linkinghub.elsevier.com/retrieve/pii/S0167691110000691>
- [4] L. Grüne and J. Pannek, *Nonlinear Model Predictive Control: Theory and Algorithms*, ser. Communications and Control Engineering. London: Springer, 2017. [Online]. Available: <http://link.springer.com/book/10.1007/978-3-319-46024-6>
- [5] M. Farina and R. Scattolini, "An output feedback distributed predictive control algorithm," in *IEEE Conference on Decision and Control and European Control Conference*, Orlando, FL, USA, 2011, pp. 8139–8144.
- [6] L. Makarek and D. Gillet, "Model predictive coordination of autonomous vehicles crossing intersections," in *IEEE Conference on Intelligent Transportation Systems, Proceedings*. The Hague, Netherlands: IEEE, 2013, pp. 1799–04.
- [7] P. Varutti, T. Faulwasser, B. Kern, M. Kögel, and R. Findeisen, "Event-based reduced-attention predictive control for nonlinear uncertain systems," *IEEE International Symposium on Computer-Aided Control System Design (CACSD), part of Multi-Conference on Systems and Control (MSC)*, pp. 1085–1090, 2010.
- [8] Y. Pu, M. N. Zeilinger, and C. N. Jones, "Quantization design for distributed optimization with time-varying parameters," in *54th IEEE Conference on Decision and Control (CDC)*, Osaka, Japan, 2015, pp. 2037–2042. [Online]. Available: <http://dx.doi.org/10.1109/CDC.2015.7402506>
- [9] P. D. Christofides, R. Scattolini, D. Muñoz de la Peña, and J. Liu, "Distributed model predictive control: A tutorial review and future research directions," *Computers & Chemical Engineering*, vol. 51, pp. 21–41, 2013. [Online]. Available: <http://linkinghub.elsevier.com/retrieve/pii/S0098135412001573>
- [10] Y. Zha and B. Houska, "Interval Superposition Arithmetic," no. October, 2016. [Online]. Available: <http://arxiv.org/abs/1610.05862>
- [11] S. Lucia, M. Kögel, and R. Findeisen, "Contract-based Predictive Control of Distributed Systems with Plug and Play Capabilities," *IFAC-PapersOnLine*, vol. 48, no. 23, pp. 205–211, 2015. [Online]. Available: <http://dx.doi.org/10.1016/j.ifacol.2015.11.284>
- [12] L. Grüne, J. Pannek, and K. Worthmann, "A prediction based control scheme for networked systems with delays and packet dropouts," in *48th IEEE Conference on Decision and Control (CDC)*. Shanghai, China: IEEE, 2009, pp. 537–542. [Online]. Available: [http://ieeexplore.ieee.org/xpls/abs\\_all.jsp?arnumber=5399922](http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5399922)
- [13] T. Sprodowski, J. K. Sagawa, and J. Pannek, "Connection between Quantization and Bandwidth Requirements of Distributed Model Predictive Control," *IFAC-PapersOnLine*, vol. 50, no. 1, 2017.
- [14] T. Sprodowski, M. W. Mehrez, K. Worthmann, G. K. I. Mann, R. G. Gosine, J. K. Sagawa, and J. Pannek, "Differential Communication with Distributed Model Predictive Control of Mobile Robots based on an Occupancy Grid," *Information Sciences*, vol. 453, pp. 426–441, 2018. [Online]. Available: <https://doi.org/10.1016/j.ins.2018.04.034>
- [15] T. Sprodowski, Y. Zha, and J. Pannek, "Interval Superposition Arithmetic Inspired Communication for Distributed Model Predictive Control," in *Proceedings of the 6th International Conference on Dynamics in Logistics (LDIC 2018)*, M. Freitag, H. Kotzab, and J. Pannek, Eds. Bremen, Germany: Springer International Publishing, 2018, pp. 327–334. [Online]. Available: [https://doi.org/10.1007/978-3-319-74225-0\\_44](https://doi.org/10.1007/978-3-319-74225-0_44)
- [16] P. G. Tzionas, A. Thanailakis, and P. G. Tsilides, "Collision-free path planning for a diamond-shaped robot using two-dimensional cellular automata," *IEEE Transactions on Robotics and Automation*, vol. 13, no. 2, pp. 237–250, 1997.
- [17] L. G. A. Martins, R. d. P. Cândido, M. C. Escarpinati, P. A. Vargas, and G. M. B. de Oliveira, "An Improved Robot Path Planning Model Using Cellular Automata," in *Towards Autonomous Robotic Systems*,

- M. Giuliani, T. Assaf, and M. E. Giannaccini, Eds. Cham: Springer International Publishing, 2018, pp. 183–194.
- [18] C.-j. Yu, Y.-h. Chen, and C.-c. Wong, “Path Planning Method Design for Mobile Robots,” in *SICE Annual Conference (SICE), 2011*. Tokyo: IEEE, 2011, pp. 1681–1686. [Online]. Available: <https://ieeexplore.ieee.org/document/6060236/>
- [19] S. M. LaValle, *Planning Algorithms*. Cambridge, U.K.: Cambridge University Press, 2006. [Online]. Available: <http://planning.cs.uiuc.edu/>
- [20] S. Johnson, “The NLOpt nonlinear-optimization package,” pp. <http://ab-initio.mit.edu/nlopt>, 2004. [Online]. Available: <http://ab-initio.mit.edu/nlopt>
- [21] M. J. D. Powell, “Direct search algorithms for optimization calculations,” *Acta Numerica*, vol. 7, pp. 287–336, 1998.

This article is reprinted via the license agreement of IEEE for reprinting in thesis: ©2017 IEEE. Reprinted, with permission.

## Kapitel 2

# Collision avoidance for mobile robots based on an occupancy grid

**Abstract:** Communication among distributed systems under the regime of a distributed model predictive control scheme (DMPC), for example mobile robots, is in many cases a necessary ingredient to steer such a system to its defined target in reasonable time. Considering short sampling times due to the movement of mobile robots, the communication burden increases with a high density of robots in a shared continuous space. To attenuate the transmission load, many approaches consider triggering events, which transmits an update when the system state changes significantly. As in robotic scenarios with fast dynamics a constant communication exchange is necessary, quantisation approaches to attenuate the communication effort are discussed and approaches based on state quantisation are presented, which allows different communication reduction strategies based on differential updates, bounding boxes and the estimation of the trajectory of the other robots, which are discussed in this paper. We demonstrate these approaches in simulations with mobile robots aiming for non-cooperative control and derive additionally sufficient conditions to ensure the collision avoidance constraints in the DMPC scheme.

## 2.1 Introduction

Due to an advanced empowerment by *Industrie 4.0*, distributed systems are a fundamental architecture in many production related scenarios, robotic scenarios or traffic scenarios, for example extending production workstations to cyber-physical systems [17], industrial robots collaborating in manufacturing processes [21], or mobile robots fulfilling a common goal, for example explore an unknown environment most efficiently considering shortest way and lowest energy consumption, see [20].

As full knowledge about the state of the overall system is often not achievable for each subsystem due to unavailability of information or privacy issues, a global optimal solution, which is solved by a central instance is therefore difficult. The complexity in the overall system in most cases is increased by two factors, either the computational load of nonlinear models or due to scalability as many subsy-

stems (machines or vehicles) have to be considered. This led to the extension of MPC to distributed MPC, in which every subsystem equipped with a controller solves an Optimal Control Problem (OCP) in a (non-)cooperative manner [24]. The problem can be divided either by using methods as dual decomposition [28], establishing hierarchical structures [29], or, if the subsystems are naturally distributed (for example mobile robots or vehicles) and their local states are not necessarily depending on external states, a coupling could be established via coupling constraints to ensure, for example collision avoidance [30]. As the global system state is not available for all subsystems and information has to be exchanged, some coordination between the subsystems has to be established: Jia and Krogh proposed in [48] a coordinated DMPC scheme in which the controllers exchange min-max open-loop predictions between each other. The min-max predictions induced by a bounded disturbance for each controller are taken into account in the local OCP as constraints to guarantee feasibility. Other approaches focus on coupling constraints or coupling penalty functions among the subsystem to coordinate the subsystems [25, 26]. Venkat et. al. showed in [15] by comparing decentralised and distributed control that distributed systems have to incorporate communication to provide subsystems with information about the overall environment to improve the performance regarding the convergence time to achieve a stable state in comparison to the performance of a centralised system. Stewart et. al. compared the performance for setpoint stabilisation for linear and nonlinear systems in [49] and [50], respectively. Their implementation does not require any hierarchical decomposition or central coordination instance. While the communication is carried out as broadcasts that every controller receives all transmitted messages, other approaches restrict the information to neighboured subsystems or successor/predecessor communication as stated in [30] and [51]. We are not focusing on the reduction of communication by reducing the number of communication links but instead on the communication load between two controllers itself. A reason for that can be the guarantee to deliver high priority messages between controllers in a certain time delay, which mostly coincide with the requirement that the bandwidth of the communication channel is never exhausted. Hence, this paper is focusing on approaches to shrink the communication load of established communication links between controllers by retaining still a near-optimal solution. Therefore, in the following paragraph, approaches of reducing communication load on links between connected controllers are discussed. Then, an overview about methods to shrink the effort of communication in distributed systems is presented, mostly considering event-triggering and quantisation methods, before introducing the idea of the occupancy grid.

### **The burden of communication**

Depending on the type of systems, communication between subsystems is restricted to bandwidth and delay: one possibility to implement communication between mobile robots or vehicles are based on wireless protocols including ad-hoc networks, which are specified for vehicles by the IEEE as *VANET* [6, 31]. Considering high

## 2.1 Introduction

13

dynamic models, for example autonomous vehicles or mobile robots, response times have to be fast, and therefore the sampling time interval has to be short, see [4]. As MPC is implemented as a receding horizon control, the predicted plan, for example discrete points along the planned trajectory, has to be communicated in each time instant. As most mobile robots use digital wireless communication channels to exchange information, the capacity is limited due to few available channels and both, bandwidth and delay suffers from a growing number of participants as the access to communication channels has to be executed exclusively, creating a bottleneck of accessing the channels with a growing number of participants. Although mechanisms for prioritised channel access exists, packet drops and bandwidth for congested highways would be still an issue, especially when urban scenarios are considered with a high density of participants and higher speeds, see [52] and [33]. Therefore, motivated by saving energy and/or bandwidth, different approaches to improve the efficiency of communication are proposed: Aiming to save energy and subsequently bandwidth, event-triggered approaches based on nonlinear MPC use a threshold function, which computes a necessary control when the state deviates from a certain region [14], or by partitioning states in several disjunct spaces to obtain a necessary change in the input [46], or by weighing the deviation from a set point against the necessary communication effort to steer the system back [45] or using Lyapunov-based function triggers, which could be activated if a predicted trajectory leads to an increase of a cost function to minimise [35].

**Lifting communication burden with quantisation**

While these event-triggering approaches are applied to systems with slow changing dynamics [1], for system with a fast-changing dynamics (mobile robots or vehicles) event-triggering may be inefficient considering communication reduction: Therefore, the implementation of quantisation approaches allows for systems with fast-changing states to reduce the communication effort, which can be classified as uniform, dynamic and logarithmic quantifiers: In [36], the load-balancing problem is applied with fixed quantifiers, which is a specialisation of the consensus problem with theoretical extensions of upper bounds regarding the convergence time. Static quantisers were introduced in [37] for a linear optimal control problem subject to a low data rate encoding control values and examines the performance with respect to the limited data rate and compare them to the performance of a Linear Quadratic Regulator (LQR). In [38], the quantisers were applied for a state estimation of measured output values via robust Kalman filters, where the relative states were quantised. As the constant quantisation scheme may require high data rates to cover the set of output values, a dynamic quantiser is proposed using relative quantised distances. As the underlying model is linear, the quantisation region is calculated by solving the Riccati equation. In [39] a quantised communication scheme is implemented for the consensus problem based on quantised relative states. A Lyapunov function candidate was derived from the communication structure of the graph and convergence is shown for first- and second-order systems, reaching the consensus

region for uniform quantisers and the consensus point for logarithmic quantisers. For dynamic quantisers with adapting finite levels, in [40] the consensus problem was solved for linear multi-agent systems applying a zoom parameter to obtain a suitable resolution for the quantiser. A mixture of fixed and dynamic quantiser was developed in [41], which allows for switching the quantisation levels depending on a dynamic or fixed communication graph among the agents. A similar approach was conducted in [9] applying a flexible quantisation scheme for a distributed optimisation problem, where the number of bits are limited and evaluated in each iteration. By combining event-triggered and quantised communication, in [5] a distributed subgradient-based optimisation problem is addressed, utilising a finite number of levels to quantise the information about the states of agents. Each agent therein is equipped with a coder/decoder to quantise information about the predicted trajectory with regards to a local cost function subject to a common constraint set among all the agents. Another approach for handling distributed optimisation problem based on multi-agents with event-triggered communication was proposed in [42], which uses dynamic quantisers also based on a finite number of levels giving conditions for convergence bounded by the quantisation error. Extending this approach, in [43] each agent calculates the necessary quantisation levels, which depends on the deviation between all connected agents. By the requirement that a feasible control can be calculated one step ahead, the number of necessary quantisation levels is known beforehand and via a zooming technique the optimal solution can be achieved. To conclude the overview one can say that each quantiser type yields advantages and drawbacks: uniform quantisers may be insufficient to depict the whole state space and therefore a large amount of levels has to be used, which induces also larger sizes of encoded information. On the other side they are robust if the system is near the optimal state as the quantisation error is constant. But nevertheless, the constant quantisation levels may hinder the overall system to reach the optimum. Dynamic quantisers may handle this quantisation error as those quantisation levels are adaptable. But a careful parameter selection has to be conducted to avoid an infinite number of quantisation levels when the system is near the optimum, which would negate the advantages of quantisation. Logarithmic quantisers combine the advantages of appropriate quantise levels near the optimum and shorter convergence but may lack of mapping the whole state space.

### **Non-cooperative control and communication**

While most of the discussed scenarios consider a cooperative control scheme following a common objective known to all subsystems and sharing a common restricted constraint set among the agents, we consider a scenario of a non-cooperative control scheme, which consists of mobile robots with individual assigned targets for each robot and an absence of a cooperative goal, see [32]. This is also more realistic for traffic scenarios, where all vehicles have individual targets but are restricted to a rule set to manage the traffic and a global information state would be difficult to obtain. The continuous space (for example a 2D plane) is shared where the robots operate

## 2.1 Introduction

15

on and hence, dynamic constraints based on the states and open-loop predictions from the other robots have to be incorporated to avoid collisions between the robots. Each robot is equipped with a local controller to execute the MPC scheme in synchronised time instants. The optimisation is carried out in the continuous space solving an optimal control problem (OCP) over a finite horizon to obtain a feasible (sub)optimal control to predict a feasible trajectory incorporating all constraints. These predicted trajectories are exchanged between the robots utilising a quantiser with a finite number of quantisation levels. Here, the shared space is quantised into equidistant cells numbered by positive integers, where the quantisation level size is equal among all robots. Each robot quantises the optimal trajectory by mapping the predicted states onto the cells and broadcasts the quantised trajectory to the other robots. After the quantised trajectories are received from all the other robots yielding an *occupancy grid* for the local robot, the trajectories are converted back into the continuous space, induced as coupling constraints in the local OCP and the optimisation is executed in the continuous space aiming for the assigned individual target. Hence, the disadvantage of the quantisation on the solution quality can be omitted as the convergence of the robots to their target states is not depending on them. Nevertheless, an appropriate choice of the quantisation level (here the cell size) influences the speed of convergence as the cell size impacts the trajectories (that are necessary larger/smaller deviations) of the robots.

**Occupancy grid**

The usage of an occupancy grid to communicate between the robots allows for the reduction of the communication effort, using integers instead of floating point values. This was shown in numerical simulation for holonomic and non-holonomic simulations in [8]. Lower bounds for the cell size of the occupancy grid were derived in [13]. Therein, also lower bounds for the safety margin to incorporate the quantisation error were presented and the communication effort were reduced further using incremental updates instead of broadcasting the full trajectory. This was also examined in an intersection scenario with autonomous connected vehicles in [44]. In a further approach, adopting the interval arithmetic idea from [19], the predicted trajectory states (here the cell indices) are restricted by a bounding box, which allows for the reduction of communication to the largest and smallest cell spanned by the rectangle [12]. As this approach reduces the communication to two cells per time instant, much more cells are assigned for the planned trajectories by using the bounding box. Therefore, we use the idea of relaxed constraints to attenuate the effect of the much more needed space in this approach [22]. The chapter is organised as follows: In the following section, the problem setting is revisited, then the prediction coherence and incremental update idea are reviewed in Sec. 2.3 and illustrated in simulation examples with holonomic robots. In Sec. 2.4 the idea of using further information without additional communication effort is revisited (bounding boxes) and sufficient conditions are derived to ensure a sufficient distance among the robots. Then, the DMPC scheme is adapted and presented in Section 2.5 and a

16

2 Collision avoidance for mobile robots based on an occupancy grid

necessary condition on the dynamics of the robots is presented, before concluding this article.

**Notation:** Sets of non-negative real and natural numbers are denoted by  $\mathbb{R}_{\geq 0}$  and  $\mathbb{N}_0$  with  $\mathbb{N}_0 := \mathbb{N} \cup \{0\}$ . We use the shorthand  $[x : y] = \{x, x+1, \dots, y\}$  for all  $x, y \in \mathbb{N}_0$  with  $x \leq y$ . For a vector  $x \in \mathbb{R}^n$ ,  $n \in \mathbb{N}$ , we define the infinity norm  $\|x\|_{\infty} := \max_{i \in [1:m]} |x_i|$ . Tuples are denoted in round brackets, for example  $x = (a, b)$ . A sequence of tuples  $(x(1), x(2), \dots, x(N))$  is denoted as  $\mathbf{x} = (x(k))_{k=1}^N$ .

## 2.2 Problem Setting

In this setting, a group of  $P$  mobile robots is considered, which are governed by a discrete time model for each robot  $p$  with

$$x_p^+ = f(x_p, u_p) = f_0(x_p) + \sum_{i=1}^m f_i(x_p) u_{p,i}, \quad (2.1)$$

where  $f_i : \mathbb{R}^d \rightarrow \mathbb{R}^d$  describe smooth vector fields for all  $i \in \{1, \dots, m\}$  with  $d, m \in \mathbb{N}_0$  and  $f_0(\cdot)$  could be interpreted as a drift term. Here, for  $x_p = (z_p, y_p)^\top \in \mathbb{X}$  planar states are considered for the 2D plane, while for  $d > 2$  additional states (for example orientation) may be considered. As we focus on the quantisation purposes, we restrict the state w.l.o.g. to  $x_p \in \mathbb{X} \subset \mathbb{R}_{\geq 0}^2$ . State  $x_p$  and control  $u_p$  are constrained by

$$x_p \in \mathbb{X} \subset \mathbb{R}_{\geq 0}^d \text{ and } u_p \in \mathbb{U} \subset \mathbb{R}^m. \quad (2.2)$$

The control values are imposed component-wise on the vector fields with the abbreviation  $u_p := (u_{p,1}, \dots, u_{p,m})^\top$ . For each robot  $p$  a finite discrete state trajectory over a finite prediction horizon length  $N$  is defined, which starts from a feasible initial point  $x_p^0 \in \mathbb{X}$  with

$$\begin{aligned} \mathbf{x}_p^u &= (x_p^u(k; x_p^0))_{k=0}^N \\ &= (x_p^0, f(x_p^0, u_p(0)), f(x_p^0 + f(x_p^0, u_p(0)), u_p(1)), \dots) \\ &= ((z_0, y_0)^\top, (z_1, y_1)^\top, \dots, (z_N, y_N)^\top) \end{aligned} \quad (2.3)$$

while the sequence of control values is incorporated as  $\mathbf{u}_p := (u_p(0), u_p(1), \dots, u_p(N-1)) \in \mathbb{R}^{(N-1) \times m}$ . Moreover, we define that the robots are able to stop immediately, which is later needed to show feasibility:

*Eigenschaft 2.1 (Immediate Hold).* Each robot can come to an immediate hold, that is

$$\forall x_p \in \mathbb{X} \exists \bar{u}_p \in \mathbb{U} : x_p = f(x_p, \bar{u}_p). \quad (2.4)$$

## 2.2 Problem Setting

17

In practice, this may be difficult to achieve, as actuators may react with a certain delay to an applied control and drift is not taken into account. Therefore, to attenuate this assumption in practice, a larger safety margin may be used to regard the delay of sensors and actuators to incorporate the violation of this stated assumption.

The continuous space  $\mathbb{X}$  is quantised by  $\mathbb{G} := [1, \dots, a_{\max}] \times [1, \dots, b_{\max}]$  where  $a_{\max}, b_{\max} \in \mathbb{N}_{\geq 0}$  are the supremum of  $\mathbb{G}$ . Each cell is then defined by a cell size  $c$  based on the continuous space  $z, y, z_{\max}, y_{\max} \in \mathbb{X}$  and  $a_{\max}, b_{\max}$  the supremum with

$$a_{\max} = \frac{z_{\max}}{c} + 1 \quad \text{and} \quad b_{\max} = \frac{y_{\max}}{c} + 1,$$

where the grid indices are calculated via a quantise function  $q: \mathbb{X} \rightarrow \mathbb{G}$  with

$$q(x_p) = \underbrace{\left( \left\lfloor \frac{z_p}{c} \right\rfloor + 1, \left\lfloor \frac{y_p}{c} \right\rfloor + 1 \right)}_{=: (a,b) \in \mathbb{G}}.$$

To express the trajectory  $\mathbf{x}_p$  from (2.3) as a finite sequence of quantised states for robot  $p$ , this is denoted as a sequence of tuples

$$\begin{aligned} \mathbf{I}_{p,n} &:= (n+k, q(x_p^{\mathbf{u}}(k; x_p^0)))_{k=0}^N \\ &= ((n, q(x_p^{\mathbf{u}}(0; x_p^0))), (n+1, q(x_p^{\mathbf{u}}(1; x_p^0))), \dots, (n+N, q(x_p^{\mathbf{u}}(N; x_p^0)))) \end{aligned} \quad (2.5)$$

with  $k \in \mathbb{N}_0$  where each tuple consists of a time stamp and state.  $\mathbf{I}_{p,n,j}$  denotes the  $j$ -th tuple  $(n+j, q(x_p^{\mathbf{u}}(j; x_p^0)))$  in the sequence with  $k \leq j \leq N$ . As the robots broadcast the quantised trajectory and therefore all other robots receive these trajectories, they are assembled to an *occupancy grid* for each robot  $p$  as

$$\mathbf{i}_{p,n} := (\mathbf{I}_{1,n}, \dots, \mathbf{I}_{p-1,n}, \mathbf{I}_{p+1,n}, \dots, \mathbf{I}_{p,n}).$$

To incorporate the information in the optimisation problem, the cell indices are converted back to the continuous space via  $h: \mathbb{N} \times \mathbb{G} \rightarrow \mathbb{X}$  with

$$h(n, q(x_p)) = \underbrace{((a-0.5)c, (b-0.5)c)^\top}_{=: (z^c, y^c)^\top \in \mathbb{X}}, \quad q(x_p) = (a, b) \in \mathbb{G}$$

and then used to formulate the coupling constraints to evaluate the necessary margin between the 2D state  $x_p = (z_p, y_p)^\top$  of robot  $p$  and the quantised state  $\mathbf{I}_{q,n}$  of the other robot  $q$  with  $q \neq p$  via the infinity norm with

$$g_{q,k} := g(x_p^{\mathbf{u}}(k; x_p^0), \mathbf{I}_{q,n,k}) = \left\| \begin{pmatrix} z_p(k) \\ y_p(k) \end{pmatrix} - h(\mathbf{I}_{q,n,k}) \right\|_\infty - r_{\min} \geq 0, \quad k \in [1 : N] \quad (2.6)$$

where  $r_{\min} \geq 0$  denotes a numerical safety margin. To choose the lower bound for the cell size  $c$ , see [13] for deriving sufficient safety margins, which consider the quantisation error and prevent skipping cells in two successive time instants. In short,

18

2 Collision avoidance for mobile robots based on an occupancy grid

this could occur if two robots are located in opposite and the next step would be the occupied cell by the other robot, see Fig. 2.1.

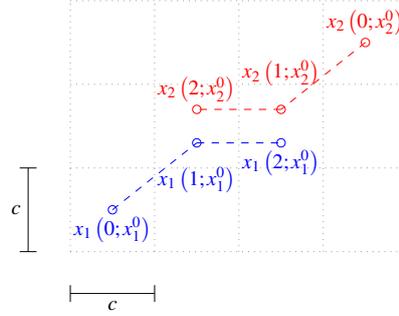


Abb. 2.1: Possible swapping of cells of two robots  $x_{\{1,2\}}$  with starting conditions  $x_{\{1,2\}}^0$  for time instants  $n = 0, 1, 2, \dots$ , which would be unnoticed if cell  $c$  size is too small.

Then, to prevent that the two robots could swap their cells in one time instant and therefore collide undetected, a minimum cell size depending on the dynamics of the robots has to be found. To obtain this, we recall the proposition from [13]:

**Proposition 2.2 (Minimum Cell Size).** Consider a time-discrete system (2.1) with state and control constraints (2.2) to be given. Then the minimum cell size to avoid that cells are skipped with  $\max_{u_p} \{ \|h(0, q(x_p)) - h(0, q(f(x_p, u_p)))\|_\infty \} \leq \underline{c}, \forall (x_p, u_p) \in \mathbb{X} \times \mathbb{U}$  and  $\underline{c} \leq d_{\min}$  where  $d_{\min} \leq 0$  describes the diameter of the robots. Then the minimum cell size satisfies  $c \geq \underline{c}$  where the lower bound is given by

$$\underline{c} := \max_{u_p \in \mathbb{U}} \left\{ c_{f_0} + \sum_{i=1}^m c_{f_i} |u_{p,i}| \right\}$$

with

$$c_{f_0} := \max_{x_p \in \mathbb{X}} \left\{ \max_{j \in \{1,2\}} |(f_0(x_p) - x_p)_j| \right\} \text{ and } c_{f_i} := \max_{x_p \in \mathbb{X}} \left\{ \max_{j \in \{1,2\}} |f_i(x_p)_j| \right\}.$$

The proof was presented in [13].

All constraints are assembled into a set of constraints according to the *occupancy grid* with

$$G(x_p^u(k; x_p^0), \mathbf{i}_{p,n,k}) = (g_{1,k}, \dots, g_{p-1,k}, g_{p+1,k}, \dots, g_{P,k}), g_{i,k} \geq 0 \forall i \in [1 : P] \setminus \{p\} \quad (2.7)$$

where  $\mathbf{i}_{p,n,k}$  defines the  $k$ -th tuples of robots  $1, \dots, p-1, p+1, \dots, P$ .

## 2.2 Problem Setting

19

```

1: Given feasible initial states and targets  $x_p^0, x_p^*$  for  $p \in [1 : P], k := 0$ 
2: for  $k = 0$  to  $N$  do
3:   Set  $\mathbf{I}_{p,n,k} := (k, q(x_p^0)) \forall p \in [1 : P]$ 
4: end for
5: Broadcast  $\mathbf{I}_{p,n}$  to all robots  $\forall p \in [1 : P]$ 
6: for  $n = 1, 2, \dots$  do
7:   for  $p = 1$  to  $P$  do
8:     Measure  $x_p^0$ 
9:     if  $\|x_p^0 - x_p^*\|_2 > \delta$  then
10:      Receive  $\mathbf{I}_{q,n}$  and construct  $G(x_p^{\mathbf{u}}(k; x_p^0), \mathbf{i}_{p,n,k})$  via (2.7) for  $k \in [1 : N]$ 
11:      Solve OCP (2.8) and apply  $u_p^*(0)$ 
12:      Broadcast  $\mathbf{I}_{p,n}$  based on (2.5)
13:    end if
14:  end for
15: end for

```

Algorithm 2.1: DMPC scheme for the overall system

These definitions allow us to discuss the idea of differential updates and prediction coherence in the following section to reduce the effort of constant communication. Each robot is equipped with a local controller to minimise a positive semi-definite objective function  $\ell_p : \mathbb{X} \times \mathbb{U} \rightarrow \mathbb{R}_{\geq 0}$  with subject to the current (measured) state  $x_p^0$  and an individual target  $x_p^*$  specified for each robot. Then, the OCP is formulated over a finite horizon  $N$  as a minimisation problem with

$$\begin{aligned}
\operatorname{argmin}_{\mathbf{u}_p} \quad & J(\mathbf{u}_p; x_p^0, x_p^*) = \operatorname{argmin}_{\mathbf{u}_p} \sum_{k=0}^{N-1} \ell_p(x_p^{\mathbf{u}}(k; x_p^0), u_p(k)) & (2.8) \\
\text{w.r.t.} \quad & x_p^{\mathbf{u}}(k+1; x_p^0) = f(x_p^{\mathbf{u}}(k; x_p^0), u_p(k)), \quad k \in [0 : N-1], \\
& u_p(k) \in \mathbb{U}, \quad k \in [0 : N-1], \\
& G(x_p^{\mathbf{u}}(k; x_p^0), \mathbf{i}_{p,n,k}) \geq 0, \quad k \in [1 : N], \\
& x_p^{\mathbf{u}}(k; x_p^0) \in \mathbb{X}, \quad k \in [1 : N], & (2.9)
\end{aligned}$$

which is executed by each robot until the target condition is matched regarding a tolerance margin  $0 < \delta \ll 1$ . The overall DMPC scheme is stated in Alg. 2.1. As this DMPC algorithm is based on the scheme of [10], no terminal costs or constraints are imposed following [23]. In the first time instant, as no robot has solved an OCP, the current feasible states are assumed to be kept over the full prediction horizon  $N$  (Alg. 2.1, line 3). Then, in a sequential order as long as the target is not reached (Alg. 2.1, line 9), each robot assembles the occupancy grid from the received tuples (Alg. 2.1, line 10), solves the OCP (Alg. 2.1, line 11) and sends the predicted trajectory in a quantised sequence of tuples to the other robots (Alg. 2.1, line 12). Note, that the quantisation does not affect the solution quality due to the execution of the optimisation in the continuous space. Hence, the additional safety margin due to the quantisation error impacts the convergence time only. To ensure that the problem is initial and recursive feasible, we recap the Theorem from [12]:

**Theorem 2.3 (Initial and Recursive Feasibility).** *Consider a set of  $P$  robots with underlying model (2.1) and given constraints (2.2) satisfying Assumptions 2.1 and the condition, that for each time instant  $n \in \mathbb{N}$ , a minimiser of (2.8) exists if the feasible set is non-empty. If Algorithm 2.1 is applied, then the problem is recursively feasible, that is for all  $n \in \mathbb{N}$  and all  $p \in [1 : P]$  there exists a solution to the OCP (2.8).*

The proof for this theorem was presented in [12]. The recursive feasibility is based on Assumption 2.1 that the algorithm does not terminate unexpectedly. If a robot is not capable to find a trajectory, which minimises the actual costs, setting  $u_p = \bar{u}_p$  will be always feasible and thus, a feasible solution for an arbitrary chosen robot exists for any time instant. Now, with this quantised information exchange we can investigate further ideas reducing the communication effort of the exchanged predicted trajectories.

### 2.3 Prediction coherence and differential updates

As the predicted trajectories are exchanged in every time instant between the robots, an investigation of these open-loop prediction promises further reduction: As the finite MPC horizon is shifted after each applied step and a warm start is used the next time instant with  $\mathbf{u}_p = (u_p(1), \dots, u_p(N-1), 0)$ , this ensures initial feasibility in every time instant incorporating the assumption, that an immediately hold of the robots is possible, see [13, Assumption 1]. But furthermore, with this warm start the trajectory of one robot has similarities in two successive time instants, which raises the question if a transmission of the full trajectory in every time instant is necessary. Let the trajectories of two successive time instants  $n = 1, n = 2$  be denoted as

$$\mathbf{x}_p^{\mathbf{u}} = (x_p^{\mathbf{u}}(k; x_p^0))_{k=1}^{N+1} \quad \text{and} \quad \bar{\mathbf{x}}_p^{\mathbf{u}} = (x_p^{\mathbf{u}}(k; x_p^0))_{k=0}^N.$$

Then, to measure the similarity of two successive trajectories, the difference between these two trajectories named as *prediction coherence*  $r_c$ : with  $r_c : \mathbb{X}^N \times \mathbb{X}^N \rightarrow \mathbb{R}_{\geq 0}$  is evaluated by

$$r_c(x_p^{\mathbf{u}}, \bar{x}_p^{\mathbf{u}}) = \sum_{k=0}^N \|x_p^{\mathbf{u}}(k+1; x_p^0) - \bar{x}_p^{\mathbf{u}}(k; x_p^0)\|_2. \quad (2.10)$$

for the trajectories in continuous space. For the prediction coherence of the quantised trajectory we define accordingly the quantised trajectories of two successive time instants with

$$\mathbf{I}_{p,n} = (n+k, q(x_p^{\mathbf{u}}(k; x_p^0)))_{k=1}^{N+1} \quad \text{and} \quad \mathbf{I}_{p,n-1} = ((n-1)+k, q(x_p^{\mathbf{u}}(k; x_p^0)))_{k=0}^N$$

for  $n > 0$ , which allows to state the prediction coherence  $r_q : \mathbb{G}^{N+1} \times \mathbb{G}^{N+1} \rightarrow \mathbb{R}_{\geq 0}$  then by

## 2.3 Prediction coherence and differential updates

21

$$r_q(\mathbf{I}_{p,n}, \mathbf{I}_{p,n-1}) = \sum_{k=0}^N \|\mathbf{I}_{p,n,k+1} - \mathbf{I}_{p,n-1,k}\|_2, \quad (2.11)$$

while the latter measures the differences of the grid indices, c.f [11].

*Beispiel 2.4 (Prediction coherence for holonomic robots).* As an example we will illustrate the problem in simulations consisting of a system of 4 mobile holonomic robots defined by the kinematic model  $\begin{pmatrix} z_p \\ y_p \end{pmatrix}^+ = \begin{pmatrix} z_p \\ y_p \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \end{pmatrix} v_p^z + \begin{pmatrix} 0 \\ 1 \end{pmatrix} v_p^y$  where  $v_p^z$  and  $v_p^y$  are defined as the control inputs and bounded by  $\sqrt{(v_p^z)^2 + (v_p^y)^2} \leq 0.5$ . The minimum cell size was obtained according to [13] with  $c = 0.5$  and the evaluation was done for cell sizes  $c = \{0.5, 1.0, 1.5, 2.0\}$  over a given continuous space  $\mathbb{X} = [-6, 6]^2$ . The stage cost function, which is used as a performance criterion is defined as

$$\ell_p(x_p, u_p) := \left\| \begin{pmatrix} (z_p - z_p^*)^2 \\ 5(y_p - y_p^*) \end{pmatrix} \right\|^2 + 0.2 \|u_p\|^2,$$

where  $x_p^* := (z_p^*, y_p^*)^\top$  defines the target position for robot  $p$  and the chosen quadratic terms for  $x$  and coefficients for  $y$  were chosen to allow for comparison of previous work [8, 11]. Individual start and target position of the robots are depicted in Table 2.1 and a plot to illustrate the comprehensive scenario is presented in Fig. 2.2. The cumulated closed-loop costs over each time instant  $n$  are defined by

Tabelle 2.1: Initial and target states of the robots

Robot ( $p$ )	Initial states ( $x_p^0$ )	Target states ( $x_p^*$ )
1	$(4.5, 4.5)^\top$	$(-4.5, -4.5)^\top$
2	$(-4.5, 4.5)^\top$	$(4.5, -4.5)^\top$
3	$(4.5, -4.5)^\top$	$(-4.5, 4.5)^\top$
4	$(-4.5, -4.5)^\top$	$(4.5, 4.5)^\top$

$$M_p(n) := \sum_{p=1}^P \ell_p(x_p^u(n; x_p^0), u_p(n)).$$

Considering the closed-loop costs, we stick to the definition stated in [8] and [13] to allow for comparison. The following Figures 2.3 and 2.4 show the prediction coherence regarding (2.10) and occupancy grid coherence regarding (2.11) for the given scenario with 4 robots for cell sizes  $c = \{0.5, 1.0, 1.5, 2.0\}$  (left figures) and for each robot in one scenario with cell size  $c = 2.0$  developing over the closed-loop time instances denoted by  $n$  (right figures). Here, the continuous prediction coherence shows that for larger cell sizes there are more similarities between two predictions as the robots need more time to cross a cell (see Fig. 2.3 left). For the middle-sized

22

2 Collision avoidance for mobile robots based on an occupancy grid

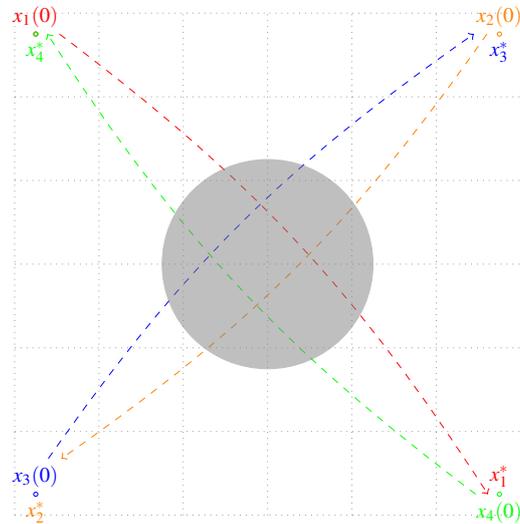


Abb. 2.2: Example for a collision avoidance conflict of four robots with given initial conditions  $x_i(0)$  and targets  $x_i^*$  for  $i \in \{1, 2, 3, 4\}$  with  $a = b = 7$  with conflicting area (gray)

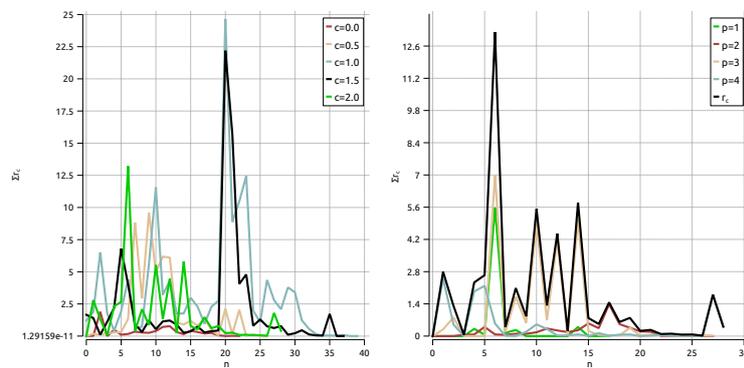


Abb. 2.3: 4 robots depicting the prediction coherence  $r_c$  for  $N = 12$  for cell sizes  $c = \{0.5, 1.0, 1.5, 2.0\}$  (left) and  $c = 2.0$  for each robot (right).

cell sizes  $c = \{1.0, 1.5\}$  it seems that the difference are larger considering the time instants around  $n = 20$ . For the occupancy grid coherence the picture is similarly to the former figures, leading to a smaller difference with larger cell sizes (see Fig. 2.4 left). Especially the situation with a chosen cell size  $c = 2.0$  shows that the fixed

2.3 Prediction coherence and differential updates

23

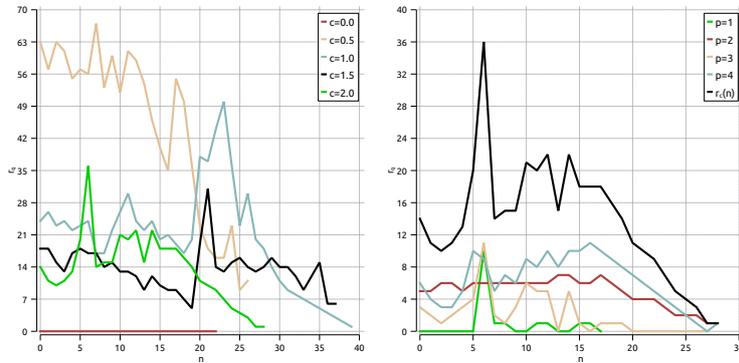


Abb. 2.4: 4 robots depicting the occupancy grid coherence  $r_q$  for  $N = 12$  for cell sizes  $c = \{0.5, 1.0, 1.5, 2.0\}$  with cumulated coherence (left) and  $c = 2.0$  for each robot individually (right).

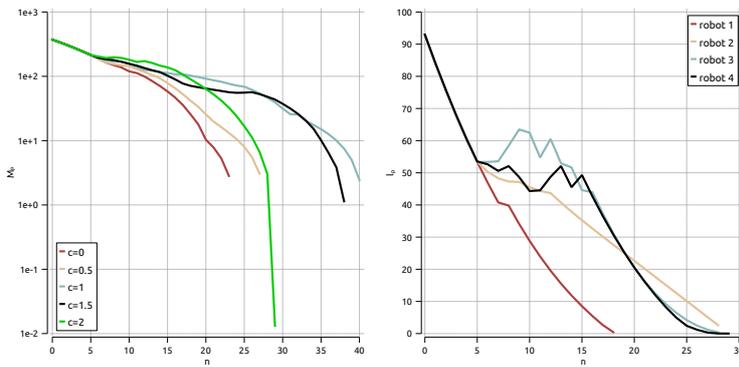


Abb. 2.5: 4 robots depicting cumulated closed-loop costs  $M_p$  for cell sizes  $c = \{0.5, 1.0, 1.5, 2.0\}$  (left) and individual costs for  $c = 2.0$  for each robot (right) over horizon length  $N = 12$ .

execution order of the robots has an impact on the coherence. This is particularly shown by the smallest difference of robot 1, which carries out its optimisation at first, as the coherence of this one is the highest (green graph). The closed-loop costs are illustrated in Fig. 2.5 showing the decrease of cumulated costs over all cell sizes (left) and for the particular scenario with cell size  $c = 2.0$  the decrease of individual costs of each robot (right). The individual decrease of costs allows the conclusion that robots 3 and 4 have to take larger detours as their optimisation order is at last in each time instant. Here, to compare the performance of the approach using grid cells against the DMPC scheme without quantisation, the continuous version of the

DMPC scheme is given with  $c = 0$ . For small and larger cell sizes the robots converges fast, while for middle-sized cell sizes the convergence times increases. In Fig. 2.5 (right) the closed-loop costs are depicted for each robot showing that the fixed sequential order of the robots gives an advantages to robot 1 as its optimisation is carried out at first, which leads to less unnecessary detours and thus, first arrival at the target.

The results from [11] and Figures 2.3 and 2.4 show a higher prediction coherence (in terms of similarity) with increasing cell sizes. Considering larger cell sizes, the prediction coherence is not proportional to the cell size. This indicates that there exists a lower bound of necessary communication, which is independent of the chosen cell size. Hence, the communication effort is bounded by two situations: On one hand, at least one message per robot has to be transmitted per time instant as the new predicted state at the end of the horizon length has to be broadcasted. On the other hand, as ensuring collision avoidance lead to detours, a certain number of cells in the prediction horizon have to be updated especially with smaller cell sizes. Therefore, depending on the dynamics of the robots, cell size and prediction horizon length, it is an open question, which cell sizes and prediction horizon length suit best for low communication load and fast convergence.

The prediction coherence for the quantised trajectory imposes the idea to broadcast differential updates instead of fully transmitted trajectories, as the grid indices of the quantised predicted trajectories are easily comparable for each successive time instant. Therefore, the quantised predicted trajectory, consisting of a sequence of tuples, is preserved for the last time instant as  $\mathbf{I}_{p,n-1}$  and allows to obtain the differential update  $\mathbf{I}_{p,n}^c$  for comparison with

$$\mathbf{I}_{p,n}^c := \begin{cases} \emptyset, & h(n+k, (a_p, b_p)) = h(n+k+1, (\bar{a}_p, \bar{b}_p)), k \in [1 : N-1] \\ (n+k, (a_p, b_p)), & h(n+k, (a_p, b_p)) \neq h(n+k+1, (\bar{a}_p, \bar{b}_p)), k \in [1 : N-1] \\ (n+k, (a_p, b_p)), & k = N \\ (n+k, (a_p, b_p)), & \text{otherwise} \end{cases} \quad (2.12)$$

$$\text{for } (\bar{a}_p, \bar{b}_p) \in \mathbf{I}_{p,n-1}, (a_p, b_p) \in \mathbf{I}_{p,n},$$

which adds here the tuples from the new trajectory  $\mathbf{I}_{p,n}$  if they are different to the previous one ( $\mathbf{I}_{p,n-1}$ ). For  $k = N$ , the last tuple is always added due to the shifted horizon. On the other hand, when the differential update  $\mathbf{I}_{q,n}^c$  is received,  $\mathbf{I}_{q,n}$  is constructed via

$$\mathbf{I}_{q,n} := \begin{cases} (n+k, (a_q, b_q)) := \mathbf{I}_{q,n,k}^c, & \exists (n+k, (a_q, b_q)) \in \mathbf{I}_{q,n}^c \\ (n+k, (\bar{a}_q, \bar{b}_q)) := \mathbf{I}_{q,n-1,k}, & \text{otherwise} \end{cases} \quad (2.13)$$

with  $q \in [1 : P] \setminus \{p\}$ . If an update exists in the received transmission from the other robot  $\mathbf{I}_{q,n}^c$ , the tuple is taken, otherwise, the tuples are taken from the preserved memory  $\mathbf{I}_{q,n-1}$ . Hence, the inclusion of the time instant is essential to enable this differential update method. Then, the communication of the DMPC algorithm is ex-

## 2.3 Prediction coherence and differential updates

25

```

1: Given feasible initial states and targets  $x_p^0, x_p^*$  for  $p \in [1 : P], k := 0$ 
2: for  $k = 0$  to  $N$  do
3:   Set  $\mathbf{I}_{p,n,k} := (k, q(x_p^0)) \forall p \in [1 : P]$ 
4: end for
5: Broadcast  $\mathbf{I}_{p,n}$  to all robots  $\forall p \in [1 : P]$ 
6: for  $n = 1, 2, \dots$  do
7:   for  $p = 1$  to  $P$  do
8:     Measure  $x_p^0$ 
9:     if  $\|x_p^0 - x_p^*\|_2 > \delta$  then
10:      Set  $\mathbf{I}_{p,n-1} := \mathbf{I}_{p,n}, \mathbf{I}_{q,n-1} := \mathbf{I}_{q,n}, q \in [1 : P] \setminus \{p\}$ 
11:      if  $n = 1$  then
12:        Receive  $\mathbf{I}_{q,n}$ 
13:      else
14:        Receive  $\mathbf{I}_{q,n}^c$  and construct  $\mathbf{I}_{q,n}$  via (2.13)
15:      end if
16:      Construct  $G(x_p^u(k; x_p^0), \mathbf{i}_{p,n,k})$  via (2.7) for  $k \in [1 : N]$ 
17:      Solve OCP (2.8) and apply  $u_p^*(0)$ 
18:      Broadcast  $\mathbf{I}_{p,n}^c$  based on (2.12)
19:    end if
20:  end for
21: end for

```

Algorithm 2.2: DMPC scheme with differential updates for the overall system

tended to incorporate the differential updates, which is presented in Alg. 2.2. Here, in the initialisation phase (Alg. 2.2, line 5) the full prediction is sent and received from the other robots (line 12). Then, the predictions, which were sent ( $\mathbf{I}_{p,n-1}$ ) and received from the other robots ( $\mathbf{I}_{q,n-1}$ ) are preserved to calculate the differential update (Alg. 2.2, line 10). While the robots have not reached their targets (Alg. 2.2, line 9), the differential updates are received and the full occupancy grid is constructed with the help of the preserved memory  $\mathbf{I}_{q,n-1}$  ((Alg. 2.2, line 14). After solving the OCP the obtained trajectory is examined, which updates have to be transmitted (Alg. 2.2, line 18). Note that the communication of differential updates is based on noiseless communication channels without delays or other disturbances. As full communication may be used to ensure robustness to use the open-loop prediction to bridge a delay in communication [47], our focus here is only on communication reduction.

*Beispiel 2.5 (Differential Updates).* We implement the setup from Example 2.4 but with differential updates in the transmitted predictions and the closed-loop costs, which are depicted in Fig. 2.6. The differential updates are presented as the average number of broadcasted tuples according to the simulation execution length  $n_\#$ , which is given by

$$K := \frac{1}{n_\#} \sum_{n=1}^{n_\#} \sum_{p=1}^P \#\mathbf{I}_{p,n}^c |n,$$

26

2 Collision avoidance for mobile robots based on an occupancy grid

where  $\#\mathbf{I}_{p,n}^c|_n$  represents the broadcasted tuples of robot  $p$  at time instant  $n$  and for the sake of completeness for the full communication (sending full predictions in each time instant) this is then

$$K = \frac{1}{n\#} \sum_{n=1}^{n\#} \sum_{p=1}^P \#\mathbf{I}_{p,n}|_n.$$

Here, the results show a significant lower number of average messages for the

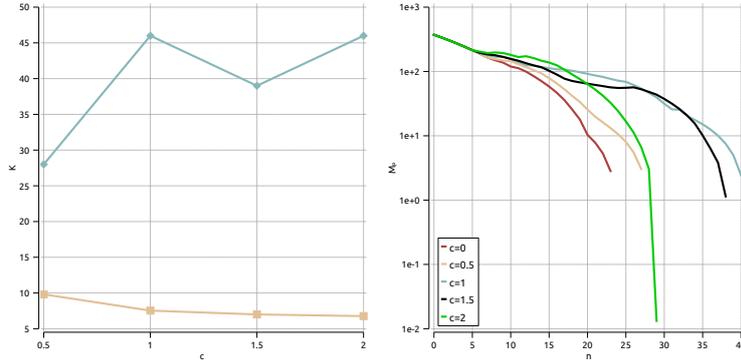


Abb. 2.6: Average number of messages in full communication manner (blue) and differential update (beige) over the cell sizes  $c = \{0.5, 1.0, 1.5, 2.0\}$  (left) and the development of the cumulated closed-loop costs  $M_p$  (right) for all cell sizes  $c$  for 4 robots over horizon length  $N = 12$ .

differential update over all cell sizes. Additionally, the average message load does not increase for the differential updates as for the full communication with larger cell sizes, also the simulation execution time varies. As the robots needs more time to cross the cells with a large cell size, the differential update shrinks to at least one cell per time instant and thus, reduces the communication load at most.

While the differential update transmission load is still influenced by the dynamics and the speed of the robots, another possibility to reduce this number of transmitted cell indices is to use the bounding box formulation, which admits a constant communication stream independent from the chosen cell size or dynamics of the underlying system.

## 2.4 Bounding box constraint formulation

In the previous two sections, we introduced the quantisation of the predicted trajectories and the calculation of differential updates. Therein, the communication load is still dependent on the dynamics and the difference of the successive open-loop trajectories. Hence, stemming from the idea of Interval Superposition Arithmetic [19], we characterise the finite quantised trajectory prediction of the robots by a bounding box. Following the quantised trajectory formulation (2.5), the bounding box is defined as a rectangle by evaluating the minimum and maximum of the predicted trajectory by robot  $p$  with

$$X_p = \left( \min_{a_p} \{C_p\}, \min_{b_p} \{C_p\}, \max_{a_p} \{C_p\}, \max_{b_p} \{C_p\} \right) \quad (2.14)$$

with  $C_p := \left\{ (a_p, b_p) \mid q(x_p^u(k; x_p^0))_{k=0}^N \right\}$ .

For an illustration, see Fig. 2.7 (left). To this end, this results in two cells to be

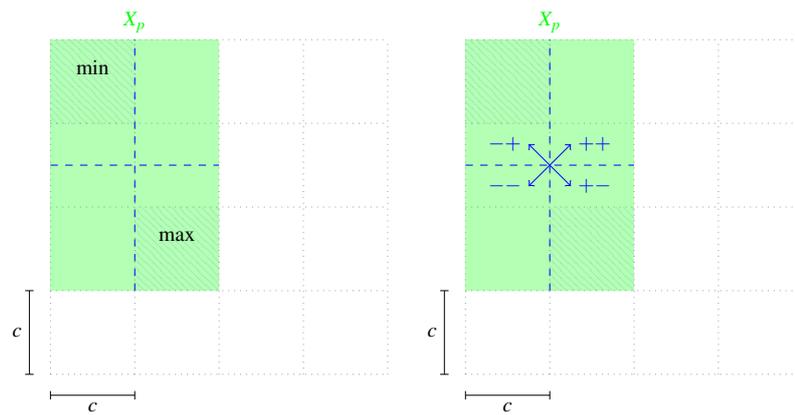


Abb. 2.7: Illustration of the bounding box declared by min/max cells (left), Illustration of usage of signs for  $(a, b)$ , depending on the direction (right).

communicated, which have to be transmitted in each time instant. Thus, the robots are able to construct the bounding box, within the other robot is moving along its trajectory. The bounding boxes of the other robots have to be avoided over the full prediction horizon as the exact trajectory is unknown, which may lead to longer detours with respect to the trajectory [12]. Now, as all robots share the same dynamics (2.1) and the same limitations considering state and control constraints (2.2), a prediction of the movement of the robots can be made if start points  $(\mathbf{I}_{q,n,k})$  and end points  $(\mathbf{I}_{q,n,k+N})$  of the quantised trajectories of all other robots  $q, q \in [1 : P] \setminus \{p\}$

are known. From the received information, which yields a bounding box, no information about the start and end points can be obtained. Therefore, as the grid indices are non-negative, that is  $(a_p, b_p) \in G$ , the encoding of the leading signs could be used to include without additional communication effort more specific information about the positions of the robots: Hence, the directions of the robots are encoded by leading signs using the combinations  $(++, --, +-, -+)$  for the first grid index  $X_q(1)$  referring to the directions "to right top", "to left bottom", "to right bottom" and "to left top" respectively, see also [22]. For an illustration, see Fig. 2.7 (right). The communicated bounding box from (2.14) is then extended to

$$\mathbf{I}_p^b := X_p = \left( \underbrace{(\pm a_1, \pm b_1)}_{\text{direction}}, (a_2, b_2) \right) \quad (2.15)$$

as a 2-tuple with  $(0, (a_1, b_1)) = \mathbf{I}_{p,1}^b$  and  $(0, (a_2, b_2)) = \mathbf{I}_{p,2}^b$ , respectively. Here, the grid indices are still interpreted as strictly positive, namely letting for example  $C_p(-2, -2) = C_p(-2, 2) = C_p(2, 2) = C_p(2, -2)$  be the same cell. To obtain the direction of a received sequence based on (2.15), the initial point  $x_q^u(n; x_q^0) := h(\mathbf{I}_{q,1}^b)$ , which represents the start position in the current sequence and the (intermediate) target point  $x_q^u(n + N; x_q^0) := h(\mathbf{I}_{q,2}^b)$ , which represents the end of the predicted trajectory, are calculated via (2.16).

Here, based on the received sequence, the grid indices, where the robots start and end, are assembled in the correct order by examining the signs via

$$\mathbf{I}_q^b := \begin{cases} (X_q), & \text{direc}(X_q) = (1, 1) \\ (X_{q,2}, -X_{q,1}), & \text{direc}(X_q) = (-1, -1) \\ ((X_{q,1,1}, X_{q,2,2}), (X_{q,2,1}, -X_{q,1,2})), & \text{direc}(X_q) = (1, -1) \\ ((X_{q,2,1}, X_{q,1,2}), (-X_{q,1,1}, X_{q,2,2})), & \text{direc}(X_q) = (-1, 1) \end{cases} \quad (2.16)$$

with  $\text{direc}(X_q) = (\text{sgn}(X_{q,1,1}), \text{sgn}(X_{q,1,2}))$

where  $X_{q,i,j}$  returns in the  $i$ -th tuple the  $j$ -th value. For an illustration of this reconstruction of the direction see Fig. 2.8. We can assume that with a large size of a bounding box the reserving robot releases the first cells quickly due to a higher speed, and therefore the cells especially in the beginning (close to  $\mathbf{I}_{q,1}^b$ ) could be used by the other robots.

Although the direction of the robot is now known, a condition is needed, which guarantee that a reserved cell is not longer in use. Otherwise, it cannot be guaranteed that a cell inside the bounding box is free at a given time  $k$ . For example, if the robot has to let pass another robot by, it is unknown to the other robots, at which speed the robot is moving inside the bounding box. Nevertheless, it is possible to calculate a lower bound, which ensures the sufficient distance between two robots  $p$  and  $q$ , assuming that robot  $q$  uses the latest possible time instant to reach the intermediate target point  $\mathbf{I}_{q,2}^b$  with maximum speed. This is formulated in the following proposi-

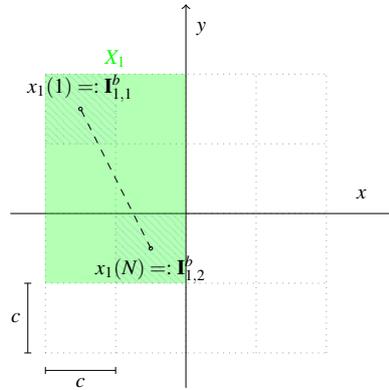


Abb. 2.8: Reconstruction of the movement of one robot (green cells) by (2.16) based on received  $X_p$  (dashed cells), for example here  $((+a_1, -b_1), (a_2, b_2))$ .

tion stating a minimum release time for each cell of the bounding box, where the robot will not use it:

**Theorem 2.6 (Minimum release time of a cell over prediction horizon length).** Consider two arbitrary robots  $p, q$  with  $x_p, x_q \in \mathbb{X}$  fulfilling (2.2) and a bounding box based on (2.15) from robot  $q$  is provided via  $\mathbf{I}_q^b := X_q$  with  $p \neq q$ . Then, for a minimum cell size  $\underline{c}$  satisfying Prop. 2.2, a finite horizon length  $N > 1, k \in [n : n + N]$  with discrete time instants  $n = 1, 2, \dots$ , the minimum release time of a cell  $\mathbf{I}_{p,n,k}$  regarding the distance to robot  $p$  to robot  $q$  is lower bounded by

$$\begin{aligned} & \max \left\{ \left\| h \left( \mathbf{I}_{q,2}^b \right) - h \left( \mathbf{I}_{q,1}^b \right) \right\|_{\infty} - (k-1)\underline{c}, \right. \\ & \quad \left. \min \left\{ \left\| h \left( \mathbf{I}_{q,2}^b \right) - h \left( \mathbf{I}_{q,1}^b \right) \right\|_{\infty}, \left\| h \left( \mathbf{I}_{q,2}^b \right) + (N-k)\underline{c} \right\|_{\infty} \right\} \right\} \\ & \leq \left\| h \left( \mathbf{I}_{p,n,k} \right) - h \left( \mathbf{I}_{q,2}^b \right) \right\|_{\infty} \end{aligned} \quad (2.17)$$

**Proof:** We distinguish here the cases  $k = 1$  and  $1 < k \leq N$ : For  $k = 1$  it is ensured by the first term of the maximum evaluation  $\left\| h \left( \mathbf{I}_{q,2}^b \right) - h \left( \mathbf{I}_{q,1}^b \right) \right\|_{\infty} - (k-1)\underline{c}$  that the bounding box of robot  $q$  is not violated by robot  $p$  with

$$\left\| h \left( \mathbf{I}_{q,2}^b \right) - h \left( \mathbf{I}_{q,1}^b \right) \right\|_{\infty} \leq \left\| h \left( \mathbf{I}_{p,n,1} \right) - h \left( \mathbf{I}_{q,2}^b \right) \right\|_{\infty}.$$

For  $1 < k \leq N$  the second term of the maximum expression ensures that as the lower cell size  $\underline{c}$  is lower bounded according to Prop. 2.2 by the maximum distance of control value  $\max_{u_p} \left\{ \left\| h \left( 0, q(x_p) \right) - h \left( 0, q(f(x_p, u_p)) \right) \right\|_{\infty} \right\} \leq \underline{c}, \forall (x_p, u_p) \in \mathbb{X} \times \mathbb{U}$  assuming that robot  $q$  is ought to achieve maximum speed. To ensure that the

30

2 Collision avoidance for mobile robots based on an occupancy grid

```

1: Input  $X_q, k$  and  $N$ 
2: Compute  $\mathbf{I}_q^b$  from (2.16)
3: Set  $i_{\min} := \mathbf{I}_{q,1,1}^b, i_{\max} := \mathbf{I}_{q,2,1}^b$ 
4: Set  $j_{\min} := \mathbf{I}_{q,1,2}^b, j_{\max} := \mathbf{I}_{q,2,2}^b$ 
5: for  $k = n$  to  $n + N$  do
6:   for  $i = i_{\min}$  to  $i_{\max}$  do
7:     for  $j = j_{\min}$  to  $j_{\max}$  do
8:       if Lemma 2.7 holds then
9:         Append  $\mathbf{I}_{q,n,k} := \mathbf{I}_{q,n,k} \cup (k, (i, j))$ 
10:        end if
11:      end for
12:    end for
13:  end for
14: return  $\mathbf{I}_{q,n}$ 

```

Algorithm 2.3: Occupancy grid construction for (robot  $p$ ) by the received sequence  $X_q$  from robots  $q \in [1 : P] \setminus \{p\}$

bounding box is not becoming larger and the predicted trajectory of robot  $p$  might become infeasible, this is avoided by taking here the minimum, that is the original bounding box stated by robot  $q$  as the estimation via  $\underline{c}$  might be larger. Then, for an increasing  $k$  the distance formulated as the radius  $(N - k)\underline{c}$  decreases and hence, the bounding box decreases. As the trajectory of robot  $q$  is assumed to be feasible, the upper possible speed limited by  $\underline{c}$  ensures that the position  $x_q^u(k; x_q^0)$  of robot  $q$  is inside  $\left\| h(\mathbf{I}_{q,2}^b) + (N - k)\underline{c} \right\|_{\infty}$  for any time instant  $1 \leq k \leq N$ .  $\square$

We can conclude that for a cell, which distance is less than the lower bound stated in the former proposition, still has to be considered for the bounding box.

**Lemma 2.7 (Examination of cells for the bounding box).** *If for a given cell  $(i, j)$  with  $(i, j) \in \mathbb{G}$  Prop. 2.6 does not hold, that is the lower bound does not hold with*

$$\begin{aligned}
& \max \left\{ \left\| h(\mathbf{I}_{q,2}^b) - h(\mathbf{I}_{q,1}^b) \right\|_{\infty} - (k - 1)\underline{c}, \right. \\
& \quad \left. \min \left\{ \left\| h(\mathbf{I}_{q,2}^b) - h(\mathbf{I}_{q,1}^b) \right\|_{\infty}, \left\| h(\mathbf{I}_{q,2}^b) + (N - k)\underline{c} \right\|_{\infty} \right\} \right\} \\
& \geq \left\| h(k, (i, j)) - h(\mathbf{I}_{q,2}^b) \right\|_{\infty}
\end{aligned}$$

then  $(k, (i, j))$  has to be added to the occupancy grid  $\mathbf{I}_{q,n}$ .

The concluded Lemma 2.7 allows the examination of all the cells included in the bounding box for all robots  $q \in [1 : P] \setminus \{p\}$  with  $\mathbf{I}_{q,1}^b \leq (\cdot, (i, j)) \leq \mathbf{I}_{q,2}^b$ , the coupling constraints can be constructed according to (2.6), which is carried out in Alg. 2.3. This algorithm ensures that all cells, which are capable of being used by the other robot  $q$  over the prediction horizon length  $N$ , are included in the occupancy grid. With an increasing  $k$ , the bounding box becomes smaller, imposing the assumption that the trajectories of each robot are not intertwined and they are moving straightly to their intermediate target  $\mathbf{I}_{q,2}^b$ , which is discussed in the next subsection.

## 2.5 DMPC with decreasing bounding box constraints

31

```

1: Given feasible initial states and targets  $x_p^0, x_p^*$  for  $p \in [1 : P], k := 0$ 
2: for  $k = 0$  to  $N$  do
3:   Set  $\mathbf{I}_{p,n,k} := (k, q(x_p^0)) \forall p \in [1 : P]$ 
4: end for
5: for  $n = 1, 2, \dots$  do
6:   for  $p = 1$  to  $P$  do
7:     Measure  $x_p^0$ 
8:     if  $\|x_p^0 - x_p^*\|_2 > \delta$  then
9:       if  $n = 1$  then
10:        Broadcast  $\mathbf{I}_p^b$  to all robots according to (2.15)
11:       end if
12:       Receive  $X_q$  for  $q \in [1 : P] \setminus \{p\}$  and construct  $\mathbf{I}_{q,n}$  via Alg. 2.3
13:       Construct  $G(x_p^u(k; x_p^0), \mathbf{I}_{p,n,k})$  via (2.7) for  $k \in [1 : N]$ 
14:       Solve OCP (2.8) and apply  $u_p^*(0)$ 
15:       Broadcast  $\mathbf{I}_p^b$  to all robots according to (2.15)
16:     end if
17:   end for
18: end for

```

Algorithm 2.4: DMPC scheme with differential updates for the overall system

## 2.5 DMPC with decreasing bounding box constraints

Now, with the reconstruction of the occupancy grid from the bounding box and usage of the direction from incorporated signs to attenuate the used number of cells, the DMPC algorithm using the decreasing bounding box method is defined in Alg. 2.4: Here, in difference to the previous communication methods, even in the initialisation phase only two cells are transmitted (see Alg. 2.4, line 10). Then, the bounding box is received from the other robots  $q$  with  $q \in [1 : P] \setminus \{p\}$  and the occupancy grid is constructed (Alg. 2.4, line 12) to be able to create the coupling constraints for the OCP. After that, the bounding box of the predicted trajectory is calculated with leading signs included and broadcasted. The overall procedure continues until the target condition is matched (Alg. 2.4, line 8).

Note that the trajectory of each robot is assumed to be inside the bounding box and the movement is assumed as an linearisation. Therefore, the underlying dynamics  $(k; x_p^0)$  of the robots has to fulfill

$$\left\| h\left(\mathbf{I}_{q,2}^b\right) - (N-k)c \right\|_{\infty} \geq \left\| h\left(\mathbf{I}_{q,2}^b\right) - (N-k)x_p^u(k; x_p^0) \right\|_{\infty}$$

for each  $k \in [k : k + N]$ . In other words, if the kinematic model is non-holonomic, the linearisation is only able to guarantee collision avoidance for those types of non-holonomic models, which dynamics are capable to keep the trajectory inside the decreasing bounding box over the prediction horizon length.

*Beispiel 2.8 (DMPC with decreasing bounding boxes).* For the decreasing bounding box approach, we refer to the parameters of Example 2.4, which are also utilised here. The closed-loop costs are depicted in Fig. 2.9. As the robots are utilising the

32

## 2 Collision avoidance for mobile robots based on an occupancy grid

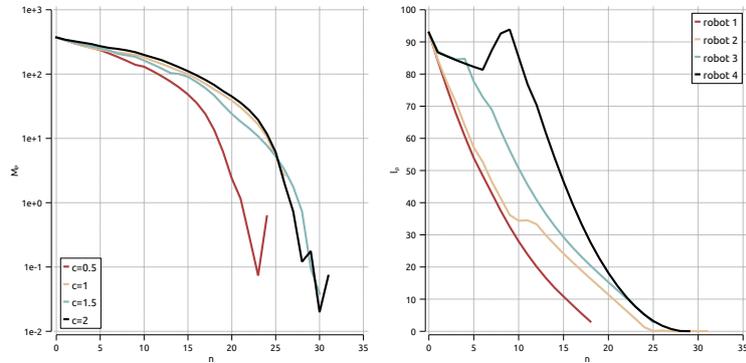


Abb. 2.9: Development of closed-loop costs  $M_p$  over time  $n$  (left) and of individual closed-loop costs for  $c = 2.0$  (right) with horizon length  $N = 12$ .

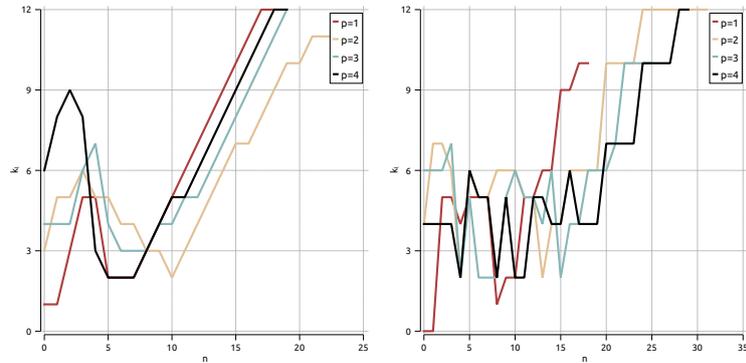


Abb. 2.10: Development of  $k_l$  (2.18) with  $c = 0.5$  (left)  $c = 2.0$  (right) for horizon length  $N = 12$ .

bounding box and restrict the robots to move on vertical or horizontal lines, the closed-loop costs are quite close among all cell sizes. To illustrate the proportion of the bounding box and the maximum possible speed of the robots, this is evaluated via

$$k_l := \left\| h \left( \mathbf{I}_{q,2}^b \right) - N_{\underline{c}} \right\|_2 - \left\| h \left( \mathbf{I}_{q,2}^b \right) - h \left( \mathbf{I}_{q,1}^b \right) \right\|_2 \quad (2.18)$$

to indicate, if the robots might leave their initial cells early or they have to wait either to let other robots passing or reaching their end point.

Here, on the left of Fig. 2.10, for small cell sizes (with  $c = 0.5$ ) the conflicts are minor for robot 1 starting at first, as its optimisation is carried out at first and therefo-

re is able to reserve most of the available space. Therefore, the latest start time is low as this robot is moving fast in the beginning. The other robots have to incorporate the imposed constraints by robot 1, therefore they cannot impose maximum speed, which leads to a higher latest start time (for example robot 4 as last robot to be in order). With ongoing simulations, the robots have to slow down, especially robot 4, which leads to a slower speed and thus, to a larger  $k_l$ . As this robot is the last one to optimise, it has to include all imposed trajectories from the other robots, which reduces the number of available cells. At the end of the simulations all robots are slowing down revealing a large latest start time and therefore a much smaller rectangle, which is then claimed by the robots. Note that by using a holonomic model, this ensures the lowest bound for the latest start time. As other kinematic models (for example four-wheels, tricycles) are constrained in their possible directions, all models have at least same or lower start times, as turn arounds or necessary curves to reach a certain position might require more steps to follow a predicted trajectory.

## 2.6 Conclusion

In this article, we have discussed and reviewed a quantisation technique based on state quantisation to minimise the communication via differential updates. Furthermore, utilising state bounding boxes leads to larger required safety margins, which can be attenuated via the estimation of the movement of the other robots. Using leading signs gives more insight information for the robots about the expected movement of the other ones, which leads to shorter convergence times. Numerical simulations were presented as examples for the differential update method and the bounding box method in a setting with holonomic robots to allow for comparison to former approaches and to show the improvements considering less communication loads and reduction of the convergence times.

Further considerations should cover theoretical aspects about the optimal choice of the cell size in the context of shorter convergence times and low communication rates. Other practical issues are the introduction of priority rules, which may reveal an optimal order of the robots regarding the current state of the whole system.

## Acknowledgements

I want to seize the opportunity to thank both reviewers for their patience and detailed comments, which helped me a lot to improve this manuscript.

**Literaturverzeichnis**

1. Panagiotis D. Christofides, Riccardo Scattolini, David Muñoz de la Peña, and Jinfeng Liu. Distributed model predictive control: A tutorial review and future research directions. *Computers & Chemical Engineering*, 51:21–41, 2013.
2. Daniel J Fagnant and Kara Kockelman. Preparing a nation for autonomous vehicles: opportunities , barriers and policy recommendations. *Transportation Research Part A*, 77:167–181, 2015.
3. Lars Grüne. NMPC without terminal constraints. *IFAC Proceedings Volumes (IFAC-PapersOnline)*, 4(PART 1):1–13, 2012.
4. Rolf Isermann. *Digital Control Systems*. Springer Berlin Heidelberg, Heidelberg, 1991.
5. Shuai Liu, Lihua Xie, and Daniel E Quevedo. Event-triggered Quantized Communication Based Distributed Convex Optimization. *IEEE Transactions on Control of Network Systems*, 5870(99):1–11, 2016.
6. Francisco J. Martinez, Juan C. Cano, Carlos T. Calafate, and Pietro Manzoni. A performance evaluation of warning message dissemination in 802.11p based VANETs. *Proceedings - Conference on Local Computer Networks, LCN*, (October):221–224, 2009.
7. Mohamed W Mehrez, George K I Mann, and Raymond G Gosine. An Optimization Based Approach for Relative Localization and Relative Tracking Control in Multi-Robot Systems. *Journal of Intelligent & Robotic Systems*,85(2):385–408, 2017.
8. Mohamed W. Mehrez, Tobias Sprodowski, Karl Worthmann, George K.I. Mann, Raymond G. Gosine, Juliana K. Sagawa, and Jürgen Pannek. Occupancy grid based distributed MPC for mobile robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 4842–4847, Vancouver, Canada, 2017, IEEE.
9. Ye Pu, Melanie N. Zeilinger, and Colin N. Jones. Quantization design for distributed optimization with time-varying parameters. In *54th IEEE Conference on Decision and Control (CDC)*, 2037–2042, Osaka, Japan, 2015, IEEE.
10. Arthur Richards and Jonathan How. Decentralized model predictive control of cooperating UAVs. In *43th IEEE Conference on Decision and Control*, (4):4286–4291, Nassau, Bahamas, 2004. IEEE.
11. Tobias Sprodowski, Juliana K. Sagawa, and Jürgen Pannek. Connection between Quantisation and Bandwidth Requirements of Distributed Model Predictive Control. *IFAC-PapersOnLine*, 50(1):10329-10334, Toulouse, France, 2017.
12. Tobias Sprodowski, Yanlin Zha, and Jürgen Pannek. Interval Superposition Arithmetic Inspired Communication for Distributed Model Predictive Control. In Michael Freitag, Herbert Kotzab, and Jürgen Pannek, editors, *Proceedings of the 6th International Conference on Dynamics in Logistics (LDIC 2018)*, 327–334, Bremen, Germany, 2018. Springer International Publishing.
13. Tobias Sprodowski, Mohamed W Mehrez, Karl Worthmann, George K I Mann, Raymond G Gosine, Juliana K Sagawa, and Jürgen Pannek. Differential Communication with Distributed Model Predictive Control of Mobile Robots based on an Occupancy Grid. *Information Sciences*, 453:426–441, 2018.
14. Paul Varutti, Timm Faulwasser, Benjamin Kern, Markus Kögel, and Rolf Findeisen. Event-based reduced-attention predictive control for nonlinear uncertain systems. *IEEE International Symposium on Computer-Aided Control System Design (CACSD), part of Multi-Conference on Systems and Control (MSC)*, 1085–1090, Yokohama, Japan, 2010, IEEE.
15. Aswin N. Venkat, James B. Rawlings, and Stephen J. Wright. Stability and optimality of distributed model predictive control. In *44th IEEE Conference on Decision and Control (CDC)*, 6680–6685, Seville, Spain, 2005. IEEE.
16. Zia Wadud, Don Mackenzie, and Paul Leiby. Help or hindrance? The travel , energy and carbon impacts of highly automated vehicles. *Transportation Research Part A*, 86:1–18, 2016.

17. Lihui Wang, Martin Törngren, and Mauro Onori. Current status and advancement of cyber-physical systems in manufacturing. *Journal of Manufacturing Systems*, 37:517–527, 2015.
18. Peng Yi and Yiguang Hong. Quantized Sub-gradient Algorithm and Data Rate Analysis for Distributed Optimization. *IEEE Transactions on Control of Network Systems*, 1(4):380–392, 2014.
19. Yanlin Zha and Boris Houska. Interval Superposition Arithmetic, <http://arxiv.org/abs/1610.05862>, 2016
20. Lu Weining, Zhang Tao, Yang Jun, and Wang Xueqian. A Formation Control Approach with Autonomous Navigation of Multi-Robot System in Unknown Environment In *Proceedings of the 34th Chinese Control Conference*:5230–5234, Hangzhou, China, 2015, IEEE.
21. Stefan Brending, Michael Lawo, Jürgen Pannek, Tobias Sprodowski, Patrick Zeising and Daniela Zimmermann. Certifiable Software Architecture for Human Robot Collaboration in Industrial Production Environments In *IFAC-PapersOnLine*, 50(1):1983–1990, Toulouse, France, 2017.
22. Tobias Sprodowski and Jürgen Pannek. Relaxed collision constraints based on interval superposition principle in a DMPC scheme. In *24th International Conference on Parallel and Distributed Systems*:831-838, Singapore, Singapore, 2018, IEEE.
23. Lars Grüne and Karl Worthmann. A distributed NMPC scheme without stabilizing terminal constraints In *Distributed Decision Making and Control*:261-287, Springer, London, 2013
24. Eduardo Camponogara, Dong Jia, Bruce H. Krogh, and Sarosh Talukdar Distributed model predictive control In *IEEE Control Systems Magazine*, 22(1):44-52, IEEE, 2002
25. William B. Dunbar Distributed receding horizon control of coupled nonlinear oscillators: Theory and application In *Proceedings of the IEEE Conference on Decision and Control (CDC)*:4854-4860, San Diego, CA, USA, 2006, IEEE
26. William B. Dunbar Distributed receding horizon control of cost coupled systems In *46th IEEE Conference on Decision and Control*:2510-2515, New Orleans, LA, USA, 2007, IEEE
27. Walter F. Tichy. Design, Implementation, and Evaluation of a Revision Control System In *6th International Conference on Software Engineering*:58-67, Los Alamitos, CA, USA, 1982, IEEE
28. Pontus Giselsson and Anders Rantzer. Distributed Model Predictive Control with Suboptimality and Stability Guarantees In *49th IEEE Conference on Decision and Control*:7272-7277, Atlanta, Georgia, USA, 2010, IEEE
29. Riccardo Scattolini. Architectures for distributed and hierarchical Model Predictive Control – A review In *Journal of Process Control*, 5:723-731, Elsevier Ltd., 2009
30. Marcello Farina, Andrea Perizzato and Riccardo Scattolini. Application of distributed predictive control to motion and coordination problems for unicycle autonomous robots In *Robotics and Autonomous Systems*, 72:248-260, 2015, Elsevier Ltd.
31. Kishwer Abdul Khaliq, Amir Qayyum and Jürgen Pannek Synergies of Advanced Technologies and Role of VANET in Logistics and Transportation In *International Journal of Advanced Computer Science and Applications*, 7(11):359-369, 2016, SAI
32. Jürgen Pannek Parallelizing a state exchange strategy for noncooperative distributed NMPC In *Systems and Control Letters*, 62(1):29-36, 2013, Elsevier Ltd.
33. Kishwer Abdul Khaliq, Amir Qayyum and Jürgen Pannek Performance Analysis of Proposed Congestion Avoiding Protocol for IEEE 802.11s In *International Journal of Advanced Computer Science and Applications*, 2:356-369, 2017, SAI
34. Xiaoduo Li, Xiwang Dong, Qingdong Li and Zhang Ren. Decentralized Event-triggered Formation of Linear Multi-agent System In *13th IEEE International Conference on Control & Automation (ICCA)*:988-993, Ohrid, Macedonia, 2017, IEEE
35. Alina Eqtami, Dimos V. Dimarogonas and Kostas J. Kyriakopoulos Novel event-triggered strategies for Model Predictive Controllers In *Proceedings of the IEEE Conference on Decision and Control (CDC)*:3392-3397, Orlando, FL, USA, 2011, IEEE

36. Akshay Kashyap, Tamer Başsar and R. Srikant Quantized consensus In *Automatica*, 43:1192-1203, 2007, Elsevier Ltd.
37. Girish N. Nair, Minyi Huang and Robin J. Evans Optimal infinite horizon control under a low data rate In *14th IFAC Symposium on System Identification*, Newcastle, Australia, 2006, IFAC
38. Teddy M. Cheng, Veerachai Malyavej and Andrey V. Savkin Set-valued state estimation for uncertain continuous-time systems via limited capacity communication channels In *IFAC Proceedings Volumes*, 41(2):3755-3760, 2008, IFAC
39. Meng Guoa and Dimos V. Dimarogonasa Consensus with quantized relative state measurement In *Automatica*, 49(8):2531-2537, 2013, Elsevier
40. Shuanghe Yu, Yilin Wang, Lina Jin and Kai Zheng Asymptotic Average Consensus of Continuous-time Multi-agent Systems with Dynamically Quantized Communication In *IFAC Proceedings Volumes*, 47(3):1819-1824, 2014, IFAC
41. Peng Yi and Yiguang Hong Quantized Sub-gradient Algorithm and Data Rate Analysis for Distributed Optimization In *IEEE Transactions on Control of Network Systems*, 1(4):380-392, 2014, IEEE
42. Shuai Liu, Lihua Xie and Daniel E. Quevedo Event-Triggered Quantized Communication-Based Distributed Convex Optimization In *IEEE Transactions on Control of Network Systems*, 5(1):167-178, 2018, IEEE
43. Huaqing Li, Shuai Liu, Yeng Chai Soh and Lihua Xie Event-Triggered Communication and Data Rate Constraint for Distributed Optimization of Multiagent Systems In *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 48(11):1908-1919, 2018, IEEE
44. Tobias Sprodownski, Juliana K. Sagawa and Jürgen Pannek Connection between Quantisation and Bandwidth Requirements of Distributed Model Predictive Control In *IFAC-PapersOnLine*, 50(1):10329-10334, 2017, IFAC
45. Huiping Li, Weisheng Yan and Yang Shi Adaptive Self-Triggered Model Predictive Control of Discrete-Time Linear Systems In *IEEE 56th Annual Conference on Decision and Control*:6165-6170, Melbourne, Australia, 2017, IEEE
46. Lars Grüne and Florian Müller An algorithm for event-based optimal feedback control In *48th IEEE Conference on Decision and Control and 28th Chinese Control Conference*, 5311-5316, Shanghai, China, 2009, IEEE
47. Lars Grüne, Jürgen Pannek and Karl Worthmann A prediction based control scheme for networked systems with delays and packet dropouts In *48h IEEE Conference on Decision and Control (CDC)*:537-542, Shanghai, China, 2009, IEEE
48. Dong Jia, and Bruce Krogh Min-max feedback model predictive control for distributed control with communication In *Proceedings of the American Control Conference*:4507-4512, Anchorage, AK, USA, 2002, IEEE
49. Brett T. Stewart, Aswin N. Venkat, James B. Rawlings, Stephen J. Wright and Gabriele Pannocchia Cooperative distributed model predictive control In *Systems & Control Letters*, 59(8):460-469, 2010, Elsevier Ltd.
50. Brett T. Stewart, Stephen J. Wright and James B. Rawlings Cooperative distributed model predictive control for nonlinear systems In *Journal of Process Control*, 21(5):698-704, 2011, Elsevier Ltd.
51. Brett T. Stewart and James B. Rawlings Coordinating multiple optimization-based controllers: New opportunities and challenges In *Journal of Process Control*, 18(9):839-845, 2008, Elsevier Ltd.
52. Muhammet A. Karabulut, A F. M. Shahen Shah and Haci Ilhan The Performance of the IEEE 802.11 DCF for Different Contention Window in VANETs In 41st International Conference on Telecommunications and Signal Processing (TSP):1-4, Athens, Greece, IEEE

This is a preprint of the following chapter: T. Sprodowski, Collision avoidance for mobile robots based on an occupancy grid, published in *Recent Advances in Model Predictive Control*, edited by T. Faulwasser, M. A. Müller, K. Worthmann, 2021, reproduced with permission of Springer Nature Switzerland AG. The final authenticated version is available online at: <http://dx.doi.org/10.1007/978-3-030-63281-6>.

## ARTICLE TEMPLATE

**Dynamic-Priority-based DMPC with an Occupancy Grid for Mobile Systems**Tobias Sprodowski<sup>a</sup> and Mohamed W. Mehrez<sup>b</sup> and Jürgen Pannek<sup>a,c</sup>

<sup>a</sup>University of Bremen, Faculty of Production Engineering, Hochschulring 20, 28359 Bremen, Germany, ORCID: 0000-0002-6792-5126; <sup>b</sup>Mechanical and Mechatronics Engineering, University of Waterloo, Waterloo, Ontario, N2L 3G1, Canada; <sup>c</sup>BIBA - Bremer Institut für Produktion und Logistik GmbH, Hochschulring 20, 28359 Bremen, Germany, ORCID: 0000-0001-5109-9627

**ARTICLE HISTORY**

Compiled March 30, 2020

**ABSTRACT**

Distributed Model Predictive Control (DMPC) is a wide-spread control method for systems such as mobile robots or vehicles, which operate in a shared space. Each system computes in an iterative way its control actions by locally solving an optimal control problem (OCP) with regards to an objective function and subject to constraints arising from the system itself and from shared and possibly quantised information. Within state-of-the-art methods, local problems are solved sequentially and applied in a fixed order, which may not be optimal considering the overall system performance. We introduce dynamic priority rules, which are evaluated locally to establish a dynamic sequence of systems in each time instant. To avoid deadlocks possibly arising from periodic reordering, the idea is complemented by a memory rule. The impact of the proposed methods with and without quantisation is analysed via numerical simulations, revealing shorter convergence times in comparison to the state-of-the art approach.

**KEYWORDS**

Distributed Model Predictive Control (DMPC); Non-cooperative control; Robot planning; Dynamic priority rules; Holonomic robots; Multi-agent system;

**1. Introduction**

Today, distributed systems are widely used in the fields of autonomous mobile robots, aircrafts or vehicles, which are naturally distributed. Since mobile robots or vehicles are increasingly equipped with computation and communication capabilities, even complex tasks may be solved locally. As a central instance may be unable to calculate a solution due to scalability and complexity issues as well as real-time requirements, a distributed approach presents a reasonable alternative to address computational issues and reduce complexities. The Distributed Model Predictive Control (DMPC) scheme provides a flexible method to track either a cooperative goal, e.g. platooning (Dunbar & Caveney, 2012), formation control (Saffarian & Fahimi, 2009) or leader-follower patterns (Farina, Perizzato, & Scattolini, 2015). Simultaneously, constraints could be ensured either on input (e.g. speed limit), state (e.g. collision avoidance constraints) or

tracking non-cooperative goals as with autonomous vehicles or robots following their individual targets (Farina & Scattolini, 2011). A survey about practical applications utilising MPC is provided by Badgwell, Qin, and Angeles (2013). The two main challenges brought by DMPC are the additional communication effort and the order of execution of the system, additionally allowing for parallel or serial execution. The first issue is heavily influenced by the choice of coordination: Iterative schemes, which allow an infinite exchange of information among the systems between two time instants (Camponogara, Jia, Krogh, & Talukdar, 2002), may achieve a consensus in a cooperative way with higher communication load, while non-iterative scheme, allowing one information exchange per time instant may only reveal a Nash equilibrium (Venkat, Rawlings, & Wright, 2005). The latter issue may be addressed by different roles of the mobile systems (leader-follower), which implicitly set the order of execution, or the execution can be carried out in an a-priori order as proposed in the DMPC scheme of Richards and How (2004a), or in parallel if a concurrent execution is allowed considering system restrictions depending on the coupling among the systems (Camponogara et al., 2002).

Distributed systems, which are not coupled or intertwined in their states, e.g. regarding collision avoidance between mobile robots, may execute their optimisation and control in parallel or deduce a dependency tree, either in a static coupling of agents or dynamic determination of dependencies, see Negenborn, De Schutter, and Hellendoorn (2007) and Pannek (2013), respectively. The question remains, how these priority rules are calculated to obtain an order in a decentralised way and which criteria are appropriate to establish a dependency between homogeneous systems?

In large-scale systems as in traffic scenarios where couplings are short lived a randomised rule is advantageous due to calculation time and complexity (Asadi & Richards, 2015). Traffic scenarios with longer coupling times among the systems may use as a criterion pre-defined static established (legislative) laws (Bali & Richards, 2018) or infrastructure constraints (e.g. lanes of the street, Wu (2001)). Other may use properties, which emerges naturally from the system, e.g. the arrival time (Evers & Proost, 2015), where each arriving vehicle is able to decide based on the decision of the earlier arrived ones (Stackelberg Game). A comprehensive survey about cooperative coordination is given in Chen and Englund (2016).

Considering mobile robotic scenarios without infrastructure restrictions, e.g. streets or lanes, at first priority criteria were firstly obtained centrally a-priori based regarding the importance of tasks (Erdmann & Lozano-Pérez, 1987) or using heuristic approaches to reduce the complexity of path panning (Yongjie, 2009). Therefore, the order of mobile robots was preserved. Dynamic approaches, which evaluate an order in every time instant, depend either on state properties or time properties. To mention some examples for state property criteria, start and goal conflicts with subject to speed or motion pattern of the robot and positions were considered in Buckley (1989); Marigo, Piccoli, and Vergni (2004), or using the inertia of moving robots to preserve energy in Fankhauser, Makareem, and Gillet (2011), or utilise the minimum path length of each to establish an order Bennewitz and Burgard (2000). For the latter, e.g. the lowest visiting time a vehicle is inside a crossing section is used (Altché & De La Fortelle, 2016) or the cumulated arrival time, utilising a centralised Pareto-optimal solution minimised among all robots (Zhao & Zhu, 2018). Combinations of time and space named as *dynamic priority assignments* were used in Azarm and Schmidt (1997). As all these criteria use a determinable system property (energy, path length, task importance), a more abstract approach may use the length of the dependency tree (Luo, Chakraborty, & Sycara, 2016) or establish a framework using dependencies in

connection with abstract criteria to achieve a maximum level of parallelism (Pannek, 2013).

Here, a distributed group of mobile systems is considered, where each is equipped with a local controller to apply a three-step DMPC scheme as proposed in Richards and How (2004b) without terminal costs or constraints based on Grüne and Worthmann (2012) aiming for individual target, i.e. the overall system is non-cooperative: First, each system receives information on planned trajectories or states of other systems in the shared space. Second, for each system an optimal control problem (OCP) is solved locally over a finite prediction horizon with regards to the (coupling) constraints emanating from the planned trajectories of the other systems. Third, each system broadcasts its (open-loop) trajectory while the first element of the solution (open-loop control) is applied. The scheme repeats in a sequential order until the target conditions are met. In our previous work, we examined the distributed control in scenarios of holonomic and non-holonomic robots, quantising the space into equidistant cells yielding a grid. Instead of transmitting the exact position, the robots mapped their predicted trajectories onto cells, which were then broadcasted as *occupancy tuples* (Sprodowski et al., 2018).

In this article, we break up the sequential and fixed order of the DMPC scheme applied in Sprodowski et al. (2018) and introduce a dynamic priority rule, which is evaluated in a distributed manner at each time instant to establish and update a dynamic sequence of systems. This approach deviates from previous proposed DMPC schemes by Richards and How (2004b); Worthmann (2012) using a fixed order, where the sequence of systems is fixed for all time instants and not necessarily adapting to the state of the overall system in terms of position, cost or other criteria. The aim of this paper is to close the gap between an ideally but mostly non-computable optimal central solution and the classical suboptimal DMPC approach with a fixed and sequential order and demonstrate the improved convergence time in numerical results. To this end, we propose to use the objective function as an updating criterion using both open- and closed-loop costs combined with different priority schemes: We first elaborate this by using a simple priority rule, which is calculated independently for every time instant and then equipped with a simple memory rule to retain the dependencies between the robot. Then, by evaluating dependencies via active constraints (Nocedal & Wright, 2000, Chapter 12), which can easily be detected via the same assignment of cell indices in the same time instant, hierarchies are used to derive an execution order and possible parallel execution of the mobile systems. Therefore, the contribution regarding the priority rules affects two properties of the DMPC scheme: At first, the order of the system is evaluated in every iteration and second, a maximum parallelism of execution is established via priority hierarchies. The computation of the priority order is carried out in a decentralised fashion as the necessary criterion values are additionally exchanged between the systems.

To assess the efficiency of our approach, we compare our results with and without quantisation of communication to the fixed order scheme from Richards and How (2004b); Worthmann (2012) without terminal costs or constraints and point out respective improvements.

The paper is organized as follows: First, the problem setting is formalised in the first part of Section 2, where the grid generation (quantisation) and coupling constraints ensuring collision avoidance are presented. Then, after stating the optimal control problem, the DMPC scheme is adapted to fit dynamic priority rules. To this end, we investigate the proposed method by means of numerical simulations for holonomic systems in Section 3 by depicting the comprehensive results, whereas the detailed

results for all applied priority rules are shown in the appendix. Last, we draw conclusions and give an outlook on future topics in Section 4.

**Notation:**  $\mathbb{R}$  and  $\mathbb{N}$  denote the real and natural numbers, respectively.  $\mathbb{N}_0 := \mathbb{N} \cup \{0\}$  represents the non-negative integers and  $\mathbb{R}_{\geq 0}$  represents the non-negative real numbers. For integers  $a, b \in \mathbb{N}_0$  with  $a \leq b$ , the expression  $[a : b]$  denotes the set  $\{a, a + 1, \dots, b\}$ . Moreover, for a vector  $x \in \mathbb{R}^n$ ,  $n \in \mathbb{N}$ , we define the infinity norm  $\|x\|_\infty := \max_{i \in [1:n]} |x_i|$ . Moreover, we define a function  $s : \mathbb{N}^P \times \mathbb{R}^P \rightarrow \mathbb{N}^P$ , which sorts elements with respect to their criterion values and returns a sequence of ascending sorted indices.

## 2. Problem setting

We first recap the discrete time model for each system and introduce the quantisation approach, which is used to derive the collision avoidance constraints. Then, the dynamic priority rules with and without memory rules are presented and the DMPC scheme is adapted.

### 2.1. Model and trajectory quantisation

Within this work, we consider a set of  $P \in \mathbb{N}_0$  systems, each driven by the discrete time model

$$\begin{aligned} x_p^\dagger &= f(x_p(n), u_p(n)) \\ &= f_0(x_p(n)) + \sum_{i=1}^m f_i(x_p(n)) u_{p,i}(n) \end{aligned} \quad (1)$$

where for all  $p \in \{1, \dots, P\}$  the vector fields  $f_i : \mathbb{R}^d \rightarrow \mathbb{R}^d$  are analytic for all  $i \in \{1, \dots, m\}$  with  $d \geq 2$  and  $d \in \mathbb{N}_0$ . The states and controls of each system are denoted by  $x_p \in \mathbb{X} \subset \mathbb{R}^d$  and  $u_p \in \mathbb{U} \subset \mathbb{R}^m$  with  $m \in \mathbb{N}_0$ , where  $\mathbb{X}$  and  $\mathbb{U}$  represent state and control constraints. We distinguish between the shared and separate spaces  $\tilde{\mathbb{X}} \subset \mathbb{R}^{\tilde{d}}$  and  $\hat{\mathbb{X}} \subset \mathbb{R}^{d-\tilde{d}}$  satisfying  $\mathbb{X} = \tilde{\mathbb{X}} \times \hat{\mathbb{X}}$ , where  $\tilde{\mathbb{X}}$  is assumed to be compact. To illustrate this in an example, the shared space could be the 2D-plane, shared by wheeled robots and has to regard collision avoidance, while the separate space could be assigned for the orientation, which does not apply for collision avoidance. The control  $u_p := (u_{p,1}, \dots, u_{p,m})$  acts component-wise on the system. To simplify notation regarding trajectories, we abbreviate a control sequence on a time interval  $\{0, 1, \dots, N\}$  via

$$\mathbf{u}_p = (u_p(0), \dots, u_p(N-1))$$

and a state trajectory corresponding to a given initial value  $x_p^0 \in \mathbb{X}$  and an applied control sequence  $\mathbf{u}_p$  via

$$\mathbf{x}_p^{\mathbf{u}}(\cdot; x_p^0) := (x_p^u(0; x_p^0), \dots, x_p^u(N; x_p^0)).$$

For this set of systems, we assume the following to hold:

**Assumption 1.** For each system  $p \in \{1, \dots, P\}$  satisfying (1), we have

$$\forall x_p \in \mathbb{X} \quad \exists \bar{u}_p \in \mathbb{U} : \quad x_p = f(x_p, \bar{u}_p). \quad (2)$$

This assumption states that each system can come to an immediate hold by imposing a control  $\bar{u}_p$ , which in control-affine systems is defined as  $\bar{u}_{p,i} = 0$  for each  $i \in \{1, \dots, m\}$ . To coordinate actions on the shared space using local controllers and reduce the required information exchange, we introduce a quantization to project respective components of states onto a grid, see Sprodowski et al. (2018, Fig. 7). To this end, we overbound  $\tilde{\mathbb{X}}$  by a set  $\bigotimes_{i=1}^{\tilde{d}} [a_i, b_i]$  with component-wise infima and suprema  $a_i := \inf\{x_i \in \mathbb{R} \mid x = (x_1, \dots, x_{\tilde{d}}) \in \tilde{\mathbb{X}}\}$ ,  $b_i := \sup\{x_i \in \mathbb{R} \mid x = (x_1, \dots, x_{\tilde{d}}) \in \tilde{\mathbb{X}}\}$ . Considering a fixed cell width  $c$ ,  $c \in \mathbb{R}_{>0}$ , we obtain a grid  $\mathcal{G} := \bigotimes_{i=1}^{\tilde{d}} \{g_1 - 1, \dots, g_i - 1\}$ , where  $g_i := \lceil (b_i - a_i)/c \rceil$ . Note that there exists a positive lower bound on  $c$ , which links the grid to the dynamic and is sufficient to avoid skipping of cells between time instances (intersampling behaviour) and to cover for safety margins, cf. (Sprodowski et al., 2018).

Similar to Sprodowski et al. (2018), we define the quantization map

$$q(x) = \begin{pmatrix} \min\{g_1 - 1, \lfloor \frac{x_1 - a_1}{c} \rfloor\} \\ \vdots \\ \min\{g_{\tilde{d}} - 1, \lfloor \frac{x_{\tilde{d}} - a_{\tilde{d}}}{c} \rfloor\} \end{pmatrix}, \quad (3)$$

which allows us to project the respective components of the state along a trajectory  $\mathbf{x}_p^u(\cdot; x_p^0)$  onto the grid  $\mathcal{G}$ . To contemplate each cell with a time index, i.e. which cell is occupied by system  $p$  at a certain time index, we refer to

$$\mathcal{I}_p(n) := (n + k; q(x_p^u(k; x_p^0)))_{k=0}^N$$

as occupancy tuples and denote the  $k$ -th tuple in this sequence by  $\mathcal{I}_p(n)(k)$ . Combining the occupancy tuples of a set of systems  $\mathcal{P}$  to be considered by system  $p$ , we obtain the occupancy grid

$$\mathbf{i}_p(n) := \{\mathcal{I}_j(n) \mid j \in \mathcal{P} \subset \{1, \dots, P\}\}. \quad (4)$$

To utilize this condensed location information within the constraints arising from using a shared space, the quantization is contemplated by the backwards map from a grid cell to its centre

$$f_c(n, q) = \begin{pmatrix} (q_1 + 0.5)c - a_1 \\ \vdots \\ (q_{\tilde{d}} + 0.5)c - a_{\tilde{d}} \end{pmatrix}.$$

Based on these maps, we can then formulate the constraints for any point along a trajectory  $\mathbf{x}_p^u(\cdot; x_p^0)$  via

$$G(x_p^u(k; x_p^0), \mathbf{i}_p(n)(k), \mathbf{i}_p(n)(k-1)) \geq 0 \quad (5)$$

using the ideas of quantized and intermediate safety margins, cf. (Sprodownski et al., 2018, Proposition 2 and 3).

## 2.2. Optimal Control Problem

In this section, based on the previous defined constraints and communication, the optimal control problem (OCP) is formulated with a stop assumption, which is later used to show feasibility of the overall DMPC scheme. Based on the discrete model (1) of each system and the shared constraints (5), we now define a local control law utilising a model predictive control scheme. We denote the target state of a system by  $x_p^*$  and state the following assumption for the performance index:

**Assumption 2.** For a given target state  $x_p^*$ , and a control  $u_p = 0$  the performance index  $\ell : \mathbb{X} \times \mathbb{U} \rightarrow \mathbb{R}_{\geq 0}$  satisfies

$$\ell_p(x_p^*, 0) = 0 \quad \text{and} \quad \inf_{u_p \in \mathbb{U}} \ell_p(x_p, u_p) > 0 \quad \forall x_p \in \mathbb{X} \setminus \{x_p^*\}.$$

Given the current state of a system  $x_p^0 = x_p(n)$  and the received information  $\mathbf{i}_p^0 = \mathbf{i}_p(n)$ , we can formulate the optimal control problem based on Assumption 2 via

$$\min_{\mathbf{u}_p} J_p^N(\mathbf{u}_p; x_p^0, \mathbf{i}_p^0) := \sum_{k=0}^{N-1} \ell_p(x_p^u(k; x_p^0), u_p(k)) \quad (6)$$

$$\text{subject to} \quad (7)$$

$$x_p^u(k+1; x_p^0) = f(x_p^u(k; x_p^0), u_p(k)), \quad (8)$$

$$u_p(k) \in \mathbb{U}, \quad (9)$$

$$x_p^u(k; x_p^0) \in \mathbb{X}, \quad \text{with } k \in [0 : N-1] \quad (10)$$

$$G(x_p^u(k; x_p^0), \mathcal{I}_q^0(k), \mathcal{I}_q^0(k-1)) \geq 0, \quad \text{with } k \in [1 : N] \quad (11)$$

Here, a cost function (6) is minimised regarding the given system dynamics (8), input constraints (9), state constraints (10), and the coupling constraints from the other systems (11), obtained by their occupancy tuples. We denote a minimizer of the above optimal control problem by  $\mathbf{u}_p^*$ , which allows us to define the corresponding value function  $V_p^N : \mathbb{X} \times (\mathbb{N}_0 \times \mathcal{G})^{(P-1)(N)} \rightarrow \mathbb{R}_{\geq 0}$  via

$$V_p^N(x_p^0, \mathbf{i}_p^0) = J_p^N(\mathbf{u}_p^*; x_p^0, \mathbf{i}_p^0). \quad (12)$$

From the formulation of the latter problem, we see that the optimal control problems are linked via the occupancy grids  $\mathbf{i}_p^0$  in the constraints  $G$  only (11). Such constraints in the shared space are necessary if multiple systems plan to occupy cells at identical time instants, yet it is not clear which system should be preferred. To solve this problem, Richards and How (2007) proposed a decomposition of the subproblems, ordered in a fixed time-invariant order and imposed terminal set constraints to ensure stability. Grüne (2012) used this scheme based on an information exchange between the subsystems and showed stability without imposing terminal constraints. This approach was generalized in Pannek (2013) to allow for parallel computation of systems via hierarchy levels, which depend on active constraints and on a fixed priority list of systems.

### 2.3. DMPC scheme with priority rules

The basis of our work is given by the DMPC scheme from Richards and How (2004a, 2007), which we modify to allow for a dynamic sequencing of systems in solving their respected OCP's (dynamic priority rule). This modification is done in several steps: in the first step, the dynamic priority rule is incorporated without any memory, i.e. sorting for each time instant the order of execution of the systems; this is achieved based on the provided criteria in the DMPC scheme in Algorithm 2 on page 8. Second, in order to avoid deadlocks or cycles in the order of execution, a memory function is introduced, which retains the order of two systems while they are coupled by active constraints. This is illustrated in the memory function rule in Algorithm 3 on page 9, which extends the DMPC scheme presented in Algorithm 2. Last, the calculation of priority rules based on hierarchies using memory and dependency evaluation to allow parallel execution is defined in Algorithm 4 on page 11 with the accompanied DMPC scheme presented in Algorithm 6 on page 12.

The priority criterion is based on the open-loop and closed-loop costs of each system. For simplicity we assume, that the calculation of the priority order rule is handled by a central instance. Beyond that, a distributed calculation by the systems is possible with necessary exchange of the criterion values, which is explicitly defined in the following algorithms. The approach is robust in the meaning that always an order could be obtained, even if participants provide false cost values. The order might be not optimal if any cost values are false but the system is still able to obtain an order as long as cost values are provided between the subsystems. Note that this covers not robustness yet by incorporating communication delays caused, e.g. by packet loss but the case that systems try to achieve a higher ranking and therefore provide smaller or zero costs instead of their correct values. To extend the priority rule by a memory function to avoid deadlocks, two different approaches are introduced: one is based on a memory function using a blocking technique, and the second based on priority hierarchies. The motivation to use a memory rule stems from the possibility of occurring periodic deadlocks: if two systems on both sides of a bottleneck are trying to traverse it; naturally one system has to step aside. If a priority rule, based on maximum stage costs, evaluates the execution order in each time instant, the system stepping aside would increase its costs. Then, the priority rule would evaluate the system with the higher costs to execute in the following time instant at first, which forces the other system in the bottleneck to step backwards. Then, the system stepping backwards will achieve higher costs and the behaviour is repeated periodically vice versa, c.f. Grüne and Pannek (2017, Chapter 9, Example 9.35). To avoid this, the priority is equipped with a memory to prevent reordering if two system depend on each other until the dependency is solved.

To allow a sorting of the optimisation order of the systems, a criterion  $\Gamma$  is chosen. An example for such a criterion are the open-loop costs  $\Gamma_p := \{V_p^N\}$  of system  $p$ . Here, to sort the systems in an ascending manner, a sort function  $s : \mathbb{N}^P \times \mathbb{R}^P \rightarrow \mathbb{N}^P$  is applied with criterion values and indexes as input with

$$m_p := s(1, \dots, P, \Gamma_1, \dots, \Gamma_P) \quad (13)$$

The initialisation and communication procedure (Algorithm 1) is executed for all variants of the DMPC algorithms proposed in this study.

The DMPC scheme using priority rules without memory is defined in Algorithm 2. Here — extending the scheme proposed in Sprodowski et al. (2018)—the systems

---

**Algorithm 1** Initialisation phase for the DMPC algorithms

---

```

1: Given admissible states  $x_p$  for initial states  $x_p^0$  for all  $p \in [1 : P]$  and time instant
    $n$ 
2: for all  $p = 1, \dots, P$  do
3:   if  $n = 0$  then
4:     for  $k = 1$  to  $N$  do
5:       Set  $\mathcal{I}_p(0)(k) := (k, q(x_p^0))$ 
6:       Set  $\Gamma_p := J_p^N(x_p^0, \bar{u}_p(x_p^0))$ 
7:     end for
8:   else
9:     Measure  $x_p(n)$ 
10:  end if
11:  Broadcast  $\mathcal{I}_p(n)$  and criterion value  $\Gamma_p := J_p^N(x_p, u_p(x_p))$ 
12:  Receive  $\mathbf{i}_p(n)$  and  $\Gamma_q$  with  $q \in [1 : P] \setminus p$ 
13: end for

```

---



---

**Algorithm 2** DMPC-scheme considering dynamic priority without memory

---

```

1: for  $n = 0, 1, \dots$  do
2:   Call Algorithm 1
3:   for  $p = 1$  to  $p = P$  do
4:      $m_p := s(1, \dots, P, \Gamma_1, \dots, \Gamma_P)$ 
5:   end for
6:   for  $p = m_p(1)$  to  $m_p(P)$  do
7:     Solve OCP (6)
8:     Apply  $\mathbf{u}_p^*(0)$ 
9:   end for
10: end for

```

---

are initialised to hold in the first time instant their current positions over the entire prediction horizon, see Algorithm 1, line 5, which represents an initial feasible solution. The resulting trajectory is quantised and broadcasted as an *occupancy grid* to the other systems, in addition the criterion value is also communicated. Then in Algorithm 2, after each system received the occupancy grid and the criterion value, a priority order is calculated based on the defined criterion values  $\Gamma_1, \dots, \Gamma_P$  yielding an ascending order, see Algorithm 2, line 4; this allows for a decentralised calculation performed by each system locally. Then, according to the obtained order, the occupancy grids are included as coupling constraints (5) in the local optimal control problem (10). After solving the latter and applying the first control value, each systems broadcasts the occupancy tuple to the others and the procedure repeats now with the actual measurement of the current state until the target condition is matched. Note, that in this algorithm an order of the systems may not be retained over two consecutive time instants, which may result in periodic behaviour or deadlocks.

In the following Algorithm 3, a memory function is introduced for conserving the order between two systems until the dependency is vanished, i.e. the coupling constraint becomes inactive. At first, the independent systems, which are not sharing active con-

---

**Algorithm 3** Memory function for priority sorting

---

**Require:** Given priority order map  $m_p$ , time instant  $n$ ,  $\mathbf{i}_p$  and  $\Gamma$

```

1: Set  $\bar{m}_p := \emptyset$ 
2: for  $i = m_p(1), \dots, m_p(P)$  do
3:   Set  $w := 0$ 
4:   for  $j = m_p(1), \dots, m_p(P)$  do
5:     if  $\mathcal{I}_i(n)(k) = \mathcal{I}_j(n)(k), k \in [1 : N]$  then
6:       Set  $w := 1$ 
7:     end if
8:   end for
9:   if  $w = 0$  then
10:    Append  $\bar{m}_p \cup i$ 
11:   end if
12: end for
13:  $\bar{m}_p := s(\bar{m}_p(1), \dots, \bar{m}_p(\#\bar{m}_p), \Gamma_{\bar{m}_p(1)}, \dots, \Gamma_{\bar{m}_p(\#\bar{m}_p)})$ 
14: Set  $b := 1$ 
15: for  $j = \bar{m}_p(1), \dots, \bar{m}_p(\#\bar{m}_p)$  do
16:   while  $m_p(b) \notin \bar{m}_p$  do
17:     Set  $b := b + 1$ 
18:   end while
19:   Set  $m_p(b) := j$ 
20:   Set  $b := b + 1$ 
21: end for
22: return  $m_p$ 

```

---

straints, are evaluated and added to an independent map  $\bar{m}_p$ , see line 10. After sorting the independent systems, the positions of these are replaced in the original map  $m_p$  with the sorted result by (13), i.e. interchanging the positions of the independent systems, see loop from line 15 ff. in Algorithm 3. Therefore, an order among dependent systems is preserved, while the exchange of independent systems is still possible. This

algorithm replaces line 4 in Algorithm 2 by

**Set**  $m_p$  by calling Algorithm 3

to include the memory functionality in the DMPC scheme. Note that this algorithm is on the one hand simple to handle the memory rule in a binary state (either one system is independent or not). On the other hand the drawback comes in the blockage for all members of the group of dependent systems; even if their dependency is transitive (system  $i$  blocks system  $j$ , system  $j$  blocks system  $k$ ), they are not allowed to change the order at all. Hence, the positions for systems, which are dependent due to active constraints, are blocked until the constraints become inactive and the blockage is revoked. Moreover, this algorithm still has difficulties with cycles as one system may enter and leave the dependency to another system periodically as mentioned before and illustrated in Grüne and Pannek (2017)[Chapter 9.4, Example 9.35]. As the priority order is carried out based on the conflicting cells obtained by the prediction of the robots, the horizon length has to be chosen long enough such that a recurrent dependency is included and possible cycles are incorporated. As long horizons may impose a high computational load, alternatives could be a memory for each subsystem, which keeps a once established order for the run of the simulation, storing all occurring dependencies and reuse this order if a dependency occurs again independently from the new cost values. Nevertheless the drawback of the latter is a less flexible order, which might be suboptimal for the later time instants as more dependencies are occurring.

Considering the next step, in order to allow concurrent executions of systems by using a dependency rule, we introduce priority hierarchies in Algorithm 4 to distinguish between the dependencies between systems. Each system is equipped with a priority variable  $\Pi_p \subset \mathbb{N}, p \in [1 : P]$ , which determines the execution order, i.e. a higher hierarchy level is executed before the following lower hierarchy level and so on. In the initialisation phase, all systems priorities are set to  $\Pi_p := 1, p \in [1 : P]$ , as they can start optimising independently. If one system observes a conflict with another system in the same hierarchy level, i.e., they are supposed to use the same cell at the same time instant (line 3 in Algorithm 4) the sorting algorithm (line 4 in Algorithm 4) is executed for both and the systems with lower priority is set to the next lower hierarchy level (condition from line 5 ff.in Algorithm 4). If a system enters a lower or higher level, it has to immediately check for possible conflicts between other systems in the same hierarchy level (line 10 in Algorithm 4).

The adopted DMPC scheme considering priority hierarchies is presented in Algorithm 6: First, in the initialisation phase for  $n = 0$  each system is initialised by calling Algorithm 1 in Algorithm 6, line 5. The priority values for all systems are initialised with  $\Pi_p = 1$ , which means that all systems with  $\Pi_p = 1$  can be executed concurrently. Systems with  $\Pi_p = 2$  are executed after all systems with  $\Pi_p = 1$  finished their execution. Systems, which get a lower hierarchy level (priority number increases), have to evaluate if a conflict exists and to shift to the next execution level and therefore withdraw the obtained control sequence (line 12 in Algorithm 6), as they have to reevaluate one on the next priority level again. At the end of the algorithm, each system examines, if they can climb up in the hierarchy again (line 20). If at least one constraint is active, the examination terminates and the current hierarchy level is kept (Algorithm 5, line 6). If no conflict exists, i.e. no active constraints from the higher hierarchy level is imposed on the current system  $p$ , the system is shifted to the higher hierarchy level and repeats this procedure until a conflict is found or the highest hierarchy level is reached ( $\Pi_p = 1$ ), see line 3. Algorithm 4 ensures that the value of priority variables

---

**Algorithm 4** Dependency evaluation among systems via hierarchies
 

---

**Require:** Given system  $p$  with  $\Pi_p$

```

1: for all  $q \in [1 : M]$  with  $\Pi_p = \Pi_q$  do
2:   for  $k = 0$  to  $N$  do
3:     if  $\mathcal{I}_p(n)(k) = \mathcal{I}_q(n)(k)$  then
4:        $m := s(p, q, \Gamma_p, \Gamma_q)$ 
5:       if  $m(1) = p$  then
6:         Set  $\Pi_q := \Pi_q + 1$ 
7:       else
8:         Set  $\Pi_p := \Pi_p + 1$ 
9:       end if
10:      Call recursively Algorithm 4 for  $i := m(2)$  with  $\Gamma_{m(2)}$ 
11:    end if
12:  end for
13: end for
14: return  $\Pi_p$ 

```

---



---

**Algorithm 5** Check existing dependencies among systems via hierarchies
 

---

```

1: if  $\Pi_p > 1$  then
2:   Set  $w := 0$ 
3:   while  $w = 0$  and  $\Pi_p > 1$  do
4:     for all  $q \in [1 : M]$  with  $\Pi_q = \Pi_p - 1$  do
5:       if  $\mathcal{I}_p(n)(k) = \mathcal{I}_q(n)(k), k \in [1 : N]$  then
6:         Set  $w := 1$ 
7:       else
8:         Set  $\Pi_p = \Pi_p - 1$ 
9:       end if
10:    end for
11:  end while
12: end if

```

---

defines the degree of parallelism. If all priority variables are  $\Pi_p = 1$  for all  $p \in P$ , then all systems  $p \in P$  can be executed in parallel. We remark that both Algorithms 2 and 6 are different from Algorithm 1 in Sprodowski et al. (2018) in the order of solving OCP (6). The latter one implements two versions of solving the OCP (6) without and with a memory rule by using Alg. 3. In essence, in Algorithm 6, a dynamic order of the optimization, which is represented by lines (6-8), 12, and (19-21), is used in contrast to the fixed order optimization used in Algorithm 1 in Sprodowski et al. (2018).

---

**Algorithm 6** DMPC-scheme for the overall system considering priority sets

---

```

1: for  $p = 1$  to  $P$  do
2:   Set  $\Pi_p = 1$ 
3: end for
4: for  $n = 0, 1, \dots$  do
5:   Call Algorithm 1
6:   for  $p = 1$  to  $P$  do
7:     Calculate priority variable  $\Pi_p$  by Algorithm 4
8:   end for
9:   for  $i := 1$  to  $\max_i \{\Pi_i\}$  do
10:    for all  $p$ , with  $\Pi_p = i$  in parallel do
11:      Solve OCP (6)
12:      Calculate  $\Pi_p$  for system  $p$  by Algorithm 4
13:      if  $\Pi_p = i$  then
14:        Apply  $\mathbf{u}_p^*(0)$ 
15:      end if
16:    end for
17:  end for
18: end for
19: for  $p = 1$  to  $P$  do
20:   Call Algorithm 5
21: end for

```

---

To show initial and recursive feasibility, Sprodowski et al. (2018, Theorem 1) has to be extended for dynamic order execution, which is stated in the following theorem:

**Theorem 1** (Initial and Recursive Feasibility). *Consider a set of  $P$  systems with dynamics (1) and constraints  $\mathbb{X}, \mathbb{U}$ . Furthermore, the set of initialization trajectories*

$$\{x_p^u(k; x_p^0) \in \mathbb{X}^N \mid x_p^u(k; x_p^0) \equiv x_p^0 \forall k \in [0 : N] \text{ and } \forall p \in P\}$$

*is feasible, i.e.*

$$G(x_p^u(k; x_p^0), \mathbf{i}_p(n)(k), \mathbf{i}_p(n)(k-1)) \geq 0 \\ \forall p \in [1 : P] \text{ and } \forall k = [0 : N]$$

*holds. If Algorithms 3, or 6 with a dynamic priority are applied, then the problem is recursively feasible, i.e. for all  $n \in \mathbb{N}_0$  and all  $p \in [1 : P]$  there exists a solution to optimal control problem (6).*

**Proof:** For  $n = 0$ , no specific order of the systems is defined. Hence, the proof of Sprodowski et al. (2018, Theorem 1) applies for the initialization showing existence of a

minimizer of (6) for all  $p \in [1 : P]$ . For  $n > 0$ , the extension to the aforementioned proof is the dynamic order. For the step from  $n-1$  to  $n$ , in all variants of the DMPC scheme (Algorithm 2 and 5)  $\mathbf{i}_p(n)$  is obtained for the current calculated order for an arbitrary system  $p$  with an assigned priority. Based on  $\mathcal{I}_q(n)$  the choice  $u_p(k) := u_p(k+1)$  is feasible and hence, there exists a feasible solution for system  $p$ . As  $p$  was chosen arbitrary, this holds for each system  $p \in [1 : P]$ .

### 3. Numerical Simulations

In this section, the numerical simulations for the DMPC scheme are explored using the introduced dynamic priority rules. To give the reader a better insight into the performance criteria of the proposed approaches, in the following Section 3.1 we present the detailed results for the simulations exemplary for priority rules with hierarchies and shift the detailed results of the remaining approaches to Appendix A and B. The comprehensive results for the remaining priority rules are discussed and concluded in Section 3.2. To allow the comparison with the classical DMPC approach, i.e. DMPC without quantisation and with a fixed optimisation order, the convergence times of those simulations from Sprodowski et al. (2018) are included in the later presented Table 2 with  $c = 0.0$  for the setting without quantisation mentioned as state-of-the-art and with  $c > 0.0$  for the fixed-order approach using quantisation. Furthermore, to eliminate the quantisation effect and, thus, illustrate the impact of the priority rules only, in all following simulations the applied priority rule without quantisation is presented additionally, indicated by  $c = 0.0$ .

The kinematic model of the holonomic robots is given by

$$\begin{pmatrix} x_p \\ y_p \end{pmatrix}^+ = \begin{pmatrix} x_p \\ y_p \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \end{pmatrix} v_p^x + \begin{pmatrix} 0 \\ 1 \end{pmatrix} v_p^y \quad (14)$$

with  $d = 2$  and  $\hat{\mathbb{X}} = \emptyset$  and states constrained by  $\mathbb{X}$  according to the control-affine structure (1). The vector fields  $f_1 = (1 \ 0)^\top$  and  $f_2 = (0 \ 1)^\top$  describe the transition of the holonomic system driven by the control variables. Here,  $v_p^x$  and  $v_p^y$  define the linear speeds in  $x$  and  $y$  directions. In addition, the set  $\mathbb{U} \subset \mathbb{R}^2$  is defined as

$$\mathbb{U} := \left\{ u_p = \begin{pmatrix} v_p^x \\ v_p^y \end{pmatrix}, \|u_p\| \leq \bar{v} \right\}, \quad \bar{v} > 0, \quad (15)$$

that is the maximum distance travelled is uniformly bounded in terms of the Euclidean distance. This induces a *coupling* of the inputs  $v_p^x$  and  $v_p^y$ .

We consider a group of  $P = 4$  mobile robots and set the state constraints  $\mathbb{X} := [-6(\text{m}), 6(\text{m})]^2 \times \hat{\mathbb{X}}$ . The control bounds are set as  $\sqrt{(\bar{v}^x)^2 + (\bar{v}^y)^2} \leq 0.5$  (m/s). Moreover, the minimum distance between the robots  $d_{\min} = 0.5 + \varepsilon$  (m) is used where  $0 < \varepsilon \ll 1$  denotes a numerical safety margin. Based on (15) in Sprodowski et al. (2018) we obtain the minimum cell size  $\underline{c} = 0.5$  (m) with  $\underline{c} = \max\{\bar{v}^x, \bar{v}^y\} + \varepsilon = 0.5(m) + \varepsilon$ .

All simulations are carried out for both priority criteria considering the minimum open-loop costs  $\Gamma_p^V := \{V_p^N\}$  based on (12) and the minimum closed-loop costs as  $\Gamma_p^\ell := \ell_p(x_p^u(n; x_p^0), u_p(n))$ . The performance criteria are stated as follows:

- the number of closed-loop iterations  $n_\#$  until all robots reached their target;

**Table 1.** Initial conditions and targets of the robots

System ( $i$ )	Initial Conditions ( $x_i^0$ )	targets ( $x_i^*$ )
1	$(4.5, 4.5, -3\pi/4)^\top$	$(-4.5, -4.5, \pi)^\top$
2	$(-4.5, 4.5, -\pi/4)^\top$	$(4.5, -4.5, 0)^\top$
3	$(4.5, -4.5, 3\pi/4)^\top$	$(-4.5, 4.5, \pi)^\top$
4	$(-4.5, -4.5, \pi/4)^\top$	$(4.5, 4.5, 0)^\top$

- the open-loop costs for each robot for a given time instant  $n$  with

$$\ell_p(x_p^u(n; x_p^0), u_p(n)); \quad (16)$$

- the cumulated closed-loop stage costs for all robots for one time instant  $n$ , which are defined as

$$M_P(n) := \sum_{p=1}^P \ell_p(x_p^u(n; x_p^0), u_p(n)); \quad (17)$$

- the maximum priority queue length of the dependency queue  $M(n)$  for a given time instant  $n$ , here  $\#M(n)$ , i.e. the dependencies between the robots and therefore, how many have to be executed in serial order, e.g. ( $\#M(n) = 4$  with 4 robots: all robots are dependent based on our setting);
- and the average number of broadcasted tuples, which is given by

$$K := \frac{1}{n\#} \sum_{n=0}^{n\#} \sum_{p=1}^P \#\mathcal{I}_p^c|n, \quad (18)$$

where  $\#\mathcal{I}_p^c|n$  represents the updated difference of new obtained tuples, which have to be broadcasted at time instant  $n$ .

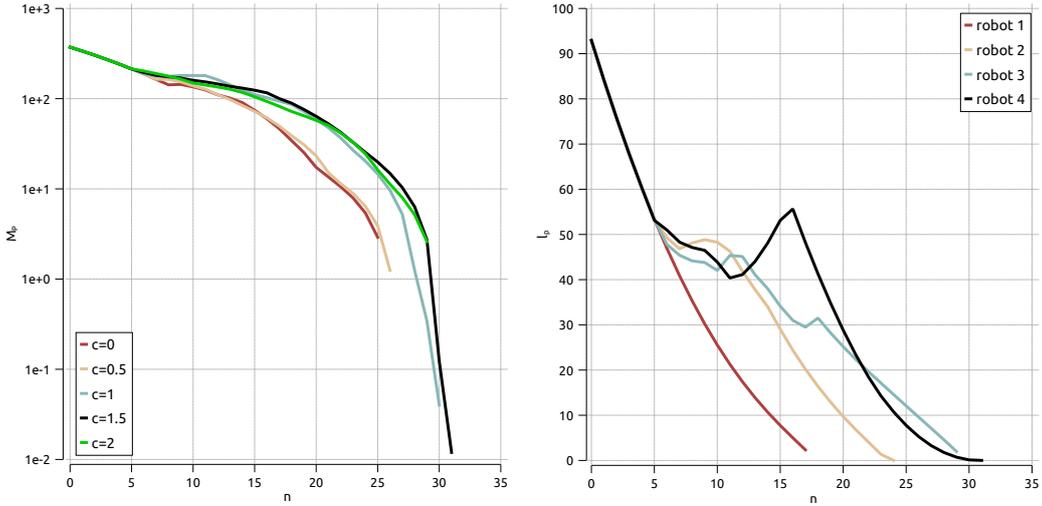
Initial and target positions are individually chosen for the robots and illustrated in Table 1. The cell size is chosen as  $c \in \{0.5, 1, 1.5, 2\}$ . As termination condition for the simulation, we utilise

$$\|x_p(n) - x_p^*\| \leq 0.01.$$

To keep the comparability with the previous publications concerning a fixed execution order regarding Sprodowski et al. (2018), the running costs  $\ell_p$  are chosen as

$$\ell_p(x_p, u_p) := \left\| \left( \begin{array}{c} (x_p - x_p^*)^2 \\ 5(y_p - y_p^*) \end{array} \right) \right\|^2 + 0.2 \|u_p\|^2.$$

The simulations utilise the infinity norm in constructing the coupling constraints as

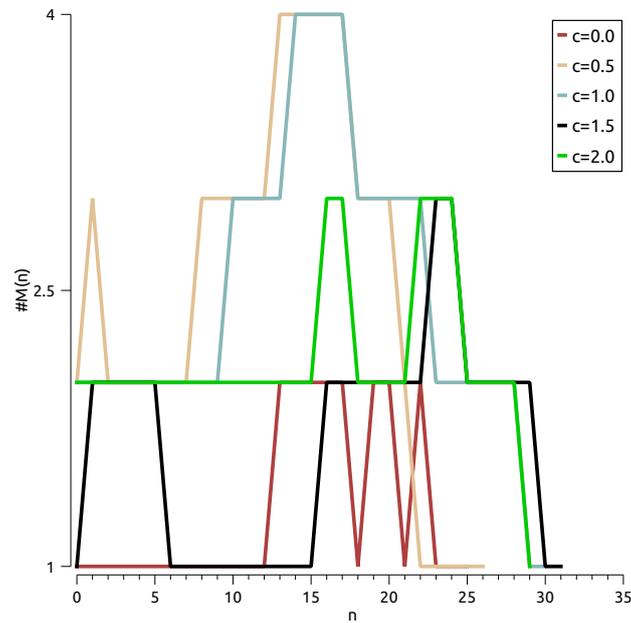


**Figure 1.** Four holonomic systems applying the priority rule based on the open-loop cost criterion  $\Gamma_V$  using a priority hierarchy: development of accumulated closed-loop costs  $M_P$  with horizon length  $N = 9$  (left) and of closed-loop costs  $\ell_p$  for each robot with cell size  $c = 1.5$  and  $N = 9$  (right).

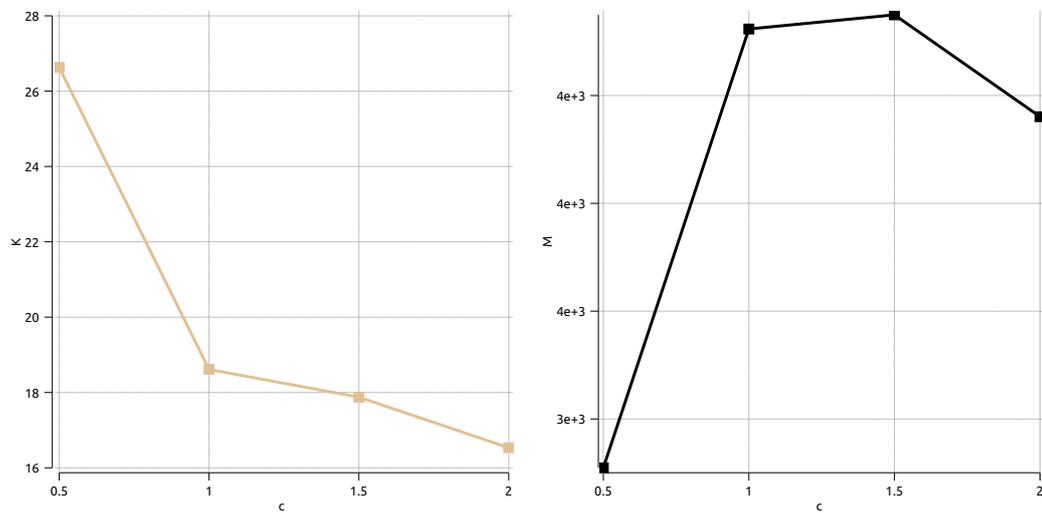
shown in Sprodowski et al. (2018, Sec. 2.2). As a derivative-free optimisation method is required to handle the infinity norm, COBYLA (Constrained Optimisation by Linear Approximations, Powell (1998)) is employed from the NLOpt optimisation package (Johnson, 2004). Overall, the simulations are implemented in C++.

### 3.1. Simulations with priority hierarchies

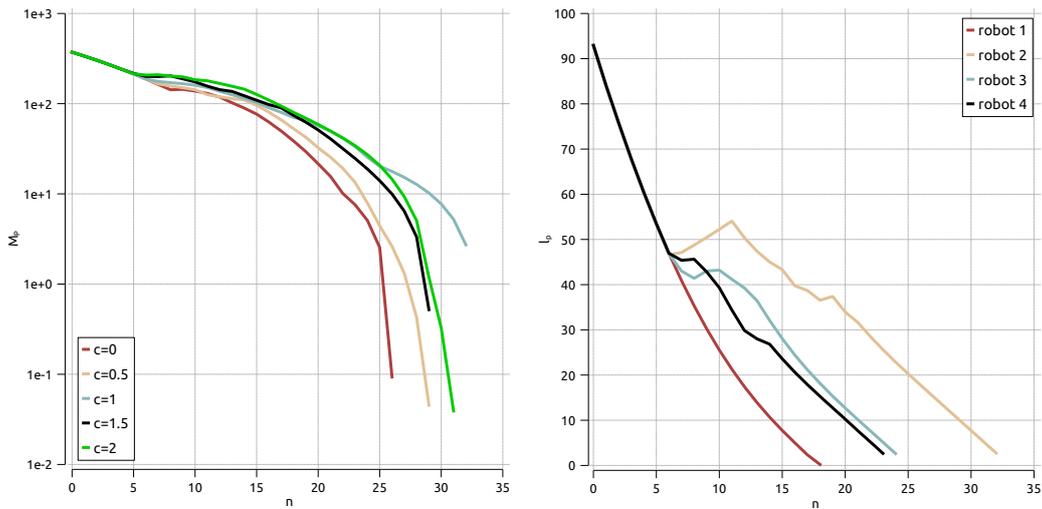
In this section, the simulations are carried out for priority hierarchies using priority variables formulated in Algorithm 6, allowing additionally a parallel execution on each hierarchy level. Fig. 1 depicts the closed-loop costs for the priority hierarchies considering the open-loop criterion. Comparing the open-loop-costs in Fig. 1 to the state-of-the-art in Table 2 (first row), the convergence time shows significantly shorter execution times for all cell sizes. Furthermore, the impact of larger cell sizes has not the same increasing effect on the convergence times as for the fixed order approach, which demonstrates numerically the efficiency of the priority rules. Moreover, as the robots with lower priority have to obey the constraints imposed by the robots with higher priority, the advantages reveal a lower complexity and shorter computation times for the robots optimising and executing first. As the movement of the robots is very similar in all different cell sizes leading to sidesteps for the lower prioritised robots and letting the higher prioritised robots pass by, the simulation times are quite close. Considering the dependencies among the robot, i.e. how many priority levels (hierarchies) occur over the simulation, see Fig. 2: Here, interestingly for the smaller cell sizes  $c = \{0.5, 1.0\}$  a full dependency among all robot is necessary (4 robots = 4 priority levels). For larger cell sizes, as the detour to obey collision avoidance is taken in an earlier time instant, the robots avoid the centre and are therefore mostly pairwise dependent. The communication effort and cumulated closed-loop costs over the cell sizes are depicted in Fig. 3. As for smaller cell sizes the robots meet in the centre, the effort for replanning is higher and leads to many update messages, while with larger cell sizes and less dependencies (fewer priority hierarchy levels) the number of



**Figure 2.** Four holonomic systems applying a priority rule based on the open-loop cost criterion  $I_V$  using a priority hierarchy: maximum priority queue length  $\#M(n)$  with horizon length  $N = 9$  for each time instant  $n$  for all cell sizes  $c$ .



**Figure 3.** Four holonomic robots (horizon length  $N = 9$ ) applying the priority rule based on the open-loop costs criterion  $I_V$  using a priority hierarchy: cell size  $c$  vs. communication load  $K$  (left) and accumulated closed-loop costs  $M_P^{n\#}$  vs. cell size  $c$  (right).



**Figure 4.** Four holonomic systems applying a priority rule based on the closed-loop costs criterion  $\Gamma_\ell$  using a priority hierarchy: development of accumulated closed-loop costs  $M_P$  (left) and closed-loop costs  $l_p$  for each robot with cell size  $c = 1.0$  and horizon length  $N = 9$  (right).

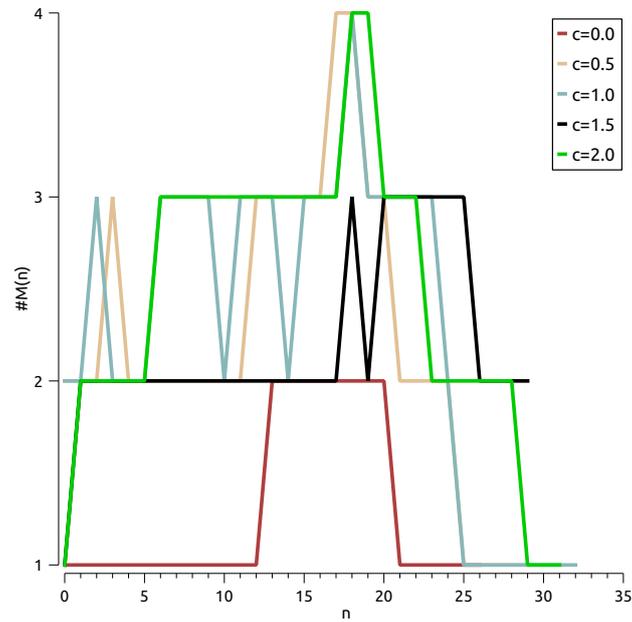
messages decreases. Nevertheless, with taking detours, the closed-loop costs cumulated over the cell sizes increases due to longer detours caused by larger cell sizes, although by applying the priority rule for large cell sizes the closed-loop costs decrease marginally.

In Fig. 4 the hierarchy scenario is depicted applying the closed-loop criterion  $\Gamma_\ell$ , illustrating the closed-loop costs over the evolution of time:

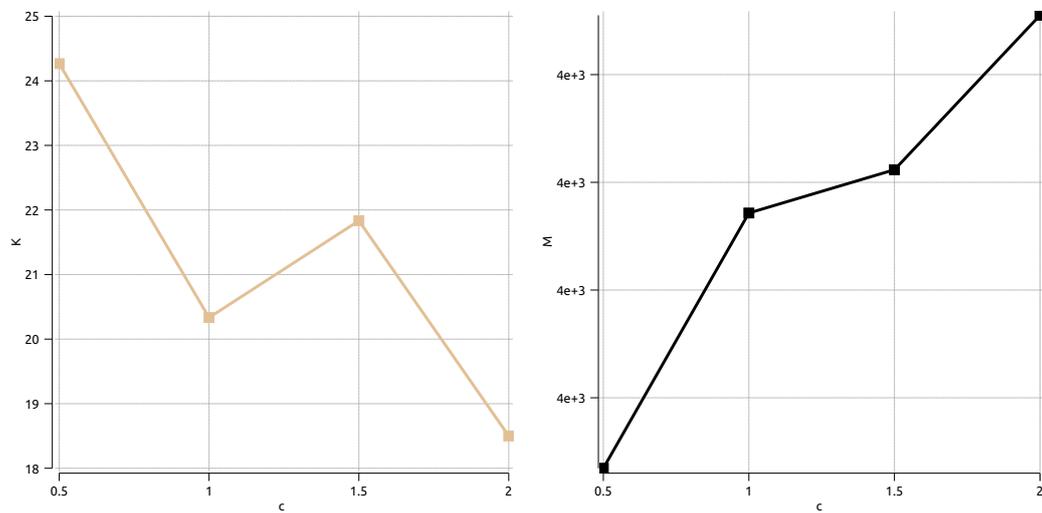
The results are slightly better for  $c = 1.5$  in comparison to the open-loop criterion, while for the other cell sizes the open-loop criterion leads to better results. A reason for that could be found by looking at the number of priority hierarchy levels assigned by the closed-loop criteria, which are depicted in Fig. 5. Here, the number of hierarchy levels oscillates more at higher frequency for smaller cell sizes. The last robot switches the dependencies more often, which leads to the lowest priority level, and therefore to the largest detour (see Fig. 4 right), hence, a much later arrival time in comparison to the other robots emerges. Considering the communication effort and culminated closed-loop costs illustrated in Fig. 6, the number of exchanged messages here is lower than for the open-loop criterion. The higher dependencies of middle-sized cell sizes  $c = 1.5$  lead to more replanning for the lower prioritised robots and therefore to an increased number of exchanged messages.

### 3.2. Conclusion of the simulations

To provide a comprehensive insight and overview over all applied approaches (fixed, dynamic priority with and without memory, and priority hierarchies) in combination with the two priority criteria (open- and closed-loop), the results are consolidated in Table 2. Detailed results are shown for the other approaches in the appendix. Note that the communication effort for the continuous case the communication effort is not presented as without quantisation the full prediction has to be broadcasted, which leads to a number of  $n_{\#}NP$  communicated tuples for one simulation. Discussing the advantages of priority rules, the overall conclusion reveals improvements in shorter convergence times by applying priority rules. An exception is observed for the priority



**Figure 5.** Four holonomic systems applying a priority rule based on the closed-loop costs criterion  $\Gamma_\ell$  using a priority hierarchy: maximum priority queue length  $\#M(n)$  with horizon length  $N = 9$  for each time instant  $n$ .



**Figure 6.** Four holonomic robots ( $N = 9$ ) applying the priority rule based on the closed-loop costs criterion  $\Gamma_\ell$  using a priority hierarchy: cell size  $c$  vs. communication load  $K$  (left) and accumulated closed-loop costs  $M_{D^\#}^{n\#}$  vs. cell size  $c$  (right).

**Table 2.** Convergence times of both priority criteria  $T_V$  (open-loop costs) and  $T_\ell$  (closed-loop costs) for all approaches and cell sizes in comparison to fixed order (state of the art)

Method	Cell Sizes					Cell Sizes				Cell Sizes				
	0.0	0.5	1.0	1.5	2.0	0.5	1.0	1.5	2.0	0.0	0.5	1.0	1.5	2.0
state of the art	24	33	34	41	46	18	12	9	7	3123	3800	4400	6100	6300
min $\ell_p$	20	27	32	32	37	9	6	8	7	3222	3700	4100	4150	4550
min $\ell_p$ (mem.)	20	22	23	32	35	49	21	20	17	3127	3420	2740	3760	4790
min $\ell_p$ (hier.)	26	28	32	28	31	24	20	22	19	2766	3550	4350	4420	4720
min $V^N$	25	48	55	30	27	7	6	8	8	3187	4800	5700	3980	3600
min $V^N$ (mem.)	20	25	28	32	32	30	24	20	19	3035	3100	3050	3800	4200
min $V^N$ (hier.)	25	26	30	31	28	27	19	18	17	2766	3550	4850	4900	4700

rule without memory using an open-loop criterion. Here, for smaller cell sizes, the deadlock problem occurs, see beginning of Sec. 2.3. Despite this, especially for the simulations using larger cell sizes the convergence time is nearly cut in half. Considering a memory function to keep the order for existing conflicts among robots even leads to a shorter convergence time for almost all cell sizes. Considering the different architectures of either keeping the order by blocking the reordering or on the other hand, introducing a priority hierarchy, the hierarchical approach leads to good and deadlock-free results for all cell sizes. Regarding the criteria, the closed-loop criterion  $T_\ell$  leads (for small cell sizes) to shorter convergence times as the evaluation of the current position inside the operational set seems to be more suitable for smaller cell sizes. For larger cell sizes, with less occurring updates of cells and thus with a higher consistency in the ongoing predictions, the open-loop criterion  $T_V$  leads to a better performance as the replanning possibilities are less due to fewer reserveable cells.

To examine the evolution of the cumulated costs, these are mostly correlated with the convergence times, as all approaches stay below the state-of-the-art approach. Higher cost values with shorter convergence times leads to the conclusion that longer detours have to be taken. This can be seen especially for the hierarchical priority rule utilising the open-loop criterion with cell size  $c = 2.0$ : Although the convergence times are shorter especially comparing these against the priority rule with/without memory considering the closed-loop criterion, the cumulated costs are mostly on the same level or even higher for both approaches. Here, as the priority hierarchy rule keeps order of the robots, which apply their control first on a straight path, the detour is longer for the low prioritised robots.

Considering the communication load, the number of exchanged messages increases according to necessary additional calculations to obtain the priority hierarchy. Nevertheless, as this is not correlated with the convergence time, it has to be weighted, if shorter convergence time predominates the additional exchange of messages. Looking for an appropriate cell size in combination with the communication load, all approaches follow the same trend as for the fixed size approach (state-of-the-art), i.e.

the communication load is reduced with increasing cell sizes. To link our results to the state-of-the-art considering both areas, i.e. traffic scenarios and distributed mobile robots, it may lead to a different selection of appropriate approaches: Looking on the one hand at the applicability in traffic scenarios with a high density, which may occur in urban areas, reducing the communication load may be a more important issue due to limits of wireless capacities. On the other hand, with a small and restricted set of mobile robots, faster convergence times might be more important as the number of participants is low. To obtain a good trade-off between both goals, the priority rule without memory based on the closed-loop criterion may achieve both: moderate communication load and short convergence times.

The approaches, which use a dynamic priority rule, show that all of them improve the convergence times in comparison to the state-of-the-art approach considering a fixed order. Beyond that, increasing the cell size leads to less impact on the convergence time for the dynamic priority rules than for the fixed order approach. Aiming for shortest convergence times independently from the cell size, the priority rule using hierarchies based on the open-loop criterion reveals the shortest convergence times. Targeting for a lower communication load and moderate convergence times, the simple priority rule without memory based on the closed-loop criterion may represent an appropriate compromise.

#### 4. Conclusion

In this paper, we utilised a distributed MPC scheme based on communication of occupancy tuples for mobile robots. Here, dynamic priority rules based on two different criteria were applied to evaluate the optimisation order of robots in each time instant. These rules were examined in robotic simulations considering different approaches: First, a simple rule without considering dependencies was applied. Then, a memory function keeping the dependencies was linked with the dynamic priority rule, which was extended to a hierarchical dependency rule. The memory function helps to avoid deadlocks while the dependency rule reorders the robots in a hierarchical structure to evaluate, possibly in parallel, their control actions. While even simple applying priority rules lead to better performances regarding convergence time in comparison to a fixed order, incorporating a memory function shows even more improved performance. By extending the memory function with hierarchies instead of using a binary blockage only has advantages if large cell sizes are imposed. Regarding dynamic priority rules in general, we can also conclude that increasing the cell size is not proportional to the convergence time when the robots reach their targets.

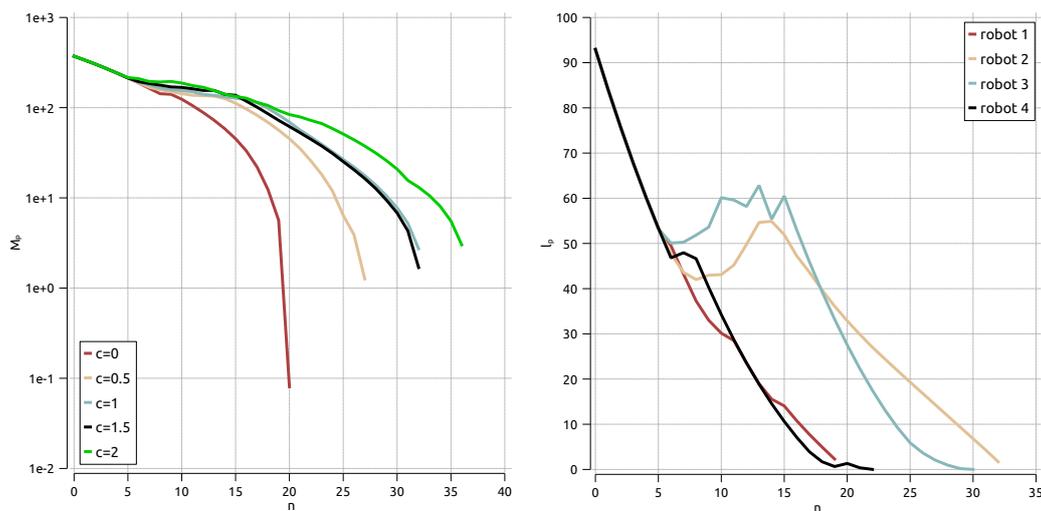
Future considerations are the dynamic variation of the occupancy grid cell size. It could be suitable to decrease the cell size when many robots are close to each other to avoid waste of space. Furthermore, as with greater cell sizes the robots need more time for crossing the cells, communicating time intervals of occupied cells instead of updating every time instant would reduce the communication effort more. Therefore, we can combine this with an event-triggered approach. Other aspects would cover the analytical examination of the properties considering stability of the occupancy grid in combination with dynamic priority rules.

## References

- Altché, F., & De La Fortelle, A. (2016). Analysis of optimal solutions to robot coordination problems to improve autonomous intersection management policies. In *Ieee intelligent vehicles symposium* (pp. 86–91). Gothenburg, Sweden: IEEE. Retrieved from <https://ieeexplore.ieee.org/document/7535369>
- Asadi, F., & Richards, A. (2015). Self-Organized Model Predictive Control for Air Traffic Management. In *Proceedings of the 5th international conference on application and theory of automation in command and control systems (ataccs '15)* (pp. 151–159). Toulouse, France: Association for Computing Machinery (ACM). Retrieved from <https://doi.org/10.1145/2899361.2899377>
- Azarm, K., & Schmidt, G. (1997). Conflict-free motion of multiple mobile robots based on decentralized motion planning and negotiation. In *Proceedings of the ieee international conference on robotics and automation* (pp. 3526–3533). Albuquerque, New Mexico, USA: IEEE. Retrieved from <https://ieeexplore.ieee.org/document/606881>
- Badgwell, T. A., Qin, S. J., & Angeles, L. (2013). Model-Predictive Control in Practice. *Encyclopedia of Systems and Control*, 500, 6–11.
- Bali, C., & Richards, A. (2018). Merging Vehicles at Junctions using Mixed-Integer Model Predictive Control. In *2018 european control conference (ecc)* (pp. 1740–1745). Limassol, Cyprus: European Control Association (EUCA). Retrieved from <https://doi.org/10.23919/ECC.2018.8550577>
- Bennewitz, M., & Burgard, W. (2000). An Experimental Comparison of Path Planning Techniques for Teams of Mobile Robots. In *Autonome mobile systeme* (pp. 175–182). Karlsruhe, Germany: Springer Berlin Heidelberg. Retrieved from [https://doi.org/10.1007/978-3-642-59576-9\\_{ }21](https://doi.org/10.1007/978-3-642-59576-9_{ }21)
- Buckley, S. J. (1989). Fast motion planning for multiple moving robots. In *International conference on robotics and automation* (pp. 322–326). Scottsdale, AZ, USA: IEEE. Retrieved from <https://doi.org/10.1109/ROBOT.1989.100008>
- Camponogara, E., Jia, D., Krogh, B., & Talukdar, S. (2002). Distributed model predictive control. *IEEE Control Systems Magazine*, 22(1), 44–52. Retrieved from <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=980246>
- Chen, L., & Englund, C. (2016). Cooperative Intersection Management: A Survey. *IEEE Transactions on Intelligent Transportation Systems*, 17(2), 570–586.
- Dunbar, W. B., & Caveney, D. S. (2012). Distributed receding horizon control of vehicle platoons: Stability and string stability. *IEEE Transactions on Automatic Control*, 57(3), 620–633. Retrieved from <http://dx.doi.org/10.1109/TAC.2011.2159651>
- Erdmann, M., & Lozano-Pérez, T. (1987). On multiple moving objects. *Algorithmica*, 2(1-4), 477–521.
- Evers, R., & Proost, S. (2015). Optimizing intersections. *Transportation Research Part B*, 71, 100–119. Retrieved from <http://dx.doi.org/10.1016/j.trb.2014.10.006>
- Fankhauser, B., Makarewicz, L., & Gillet, D. (2011). Collision-free intersection crossing of mobile robots using decentralized navigation functions on predefined paths. In *Ieee 5th international conference on cybernetics and intelligent systems (cis 2011)* (pp. 392–397). Qingdao, China: IEEE. Retrieved from <https://doi.org/10.1109/ICCIS.2011.6070361>
- Farina, M., Perizzato, A., & Scattolini, R. (2015). Application of distributed predictive control to motion and coordination problems for unicycle autonomous robots. *Robotics and Autonomous Systems*, 72, 248–260. Retrieved from <http://linkinghub.elsevier.com/retrieve/pii/S0921889015001359>
- Farina, M., & Scattolini, R. (2011). An output feedback distributed predictive control algorithm. In *Ieee conference on decision and control and european control conference* (pp. 8139–8144). Orlando, FL, USA: IEEE. Retrieved from <https://ieeexplore.ieee.org/document/6160366/>
- Grüne, L. (2012). NMPC without terminal constraints. *IFAC Proceedings Volumes (IFAC-PapersOnline)*, 45(17), 1–13. Retrieved from <https://doi.org/10.3182/20120823-5-NL>

- 3013.00030
- Grüne, L., & Pannek, J. (2017). *Nonlinear Model Predictive Control: Theory and Algorithms*. London: Springer. Retrieved from <http://link.springer.com/book/10.1007/978-3-319-46024-6>
- Grüne, L., & Worthmann, K. (2012). A distributed NMPC scheme without stabilizing terminal constraints. In *Distributed decision making and control* (pp. 261–287). London: Springer. Retrieved from [http://dx.doi.org/10.1007/978-1-4471-2265-4\\_12](http://dx.doi.org/10.1007/978-1-4471-2265-4_12)
- Johnson, S. G. (2004). *The NLOpt nonlinear-optimization package*. <http://ab-initio.mit.edu/nlopt> (accessed 2019-04-02). Retrieved from <http://ab-initio.mit.edu/nlopt>
- Luo, W., Chakraborty, N., & Sycara, K. (2016). Distributed dynamic priority assignment and motion planning for multiple mobile robots with kinodynamic constraints. In *Proceedings of the american control conference* (pp. 148–154). Boston, MA, USA: IEEE. Retrieved from <https://doi.org/10.1109/ACC.2016.7524907>
- Marigo, A., Piccoli, B., & Vergni, D. (2004). Cooperative controls for car-like robot coordination. In *Proceedings of the 42nd IEEE conference on decision and control* (pp. 3287–3292). Maui, Hawaii, USA: IEEE. Retrieved from <https://ieeexplore.ieee.org/document/1271650/>
- Negenborn, R. R., De Schutter, B., & Hellendoorn, J. (2007). Efficient implementation of serial multi-agent model predictive control by parallelization. In *IEEE international conference on networking, sensing and control* (Vol. 19, pp. 175–180). London, UK: IEEE. Retrieved from <https://ieeexplore.ieee.org/document/4238985/>
- Nocedal, J., & Wright, S. (2000). *Numerical Optimization*. Springer New York. Retrieved from <https://books.google.de/books?id=epc5fX01qRIC>
- Pannek, J. (2013). Parallelizing a state exchange strategy for noncooperative distributed NMPC. *Systems and Control Letters*, 62(1), 29–36. Retrieved from <http://dx.doi.org/10.1016/j.sysconle.2012.10.015>
- Powell, M. J. D. (1998). Direct search algorithms for optimization calculations. *Acta Numerica*, 7, 287–336.
- Richards, A., & How, J. (2004a). A decentralized algorithm for robust constrained model predictive control. In *American control conference (acc)* (pp. 4261–4266). Boston, MA, USA: IEEE. Retrieved from <https://ieeexplore.ieee.org/document/1383977/>
- Richards, A., & How, J. (2004b). Decentralized model predictive control of cooperating UAVs. In *43th IEEE conference on decision and control* (Vol. 4, pp. 4286–4291). Nassau, Bahamas: IEEE. Retrieved from <https://doi.org/10.1109/CDC.2004.1429425>
- Richards, A., & How, J. P. (2007). Robust distributed model predictive control. *International Journal of Control*, 80(9), 1517–1531. Retrieved from <http://dx.doi.org/10.1080/00207170701491070>
- Saffarian, M., & Fahimi, F. (2009). Non-Iterative nonlinear model predictive approach applied to the control of helicopters group formation. *Robotics and Autonomous Systems*, 57(6-7), 749–757. Retrieved from <https://doi.org/10.1016/j.robot.2008.10.021>
- Sprodownski, T., Mehrez, M. W., Worthmann, K., Mann, G. K., Gosine, R. G., Sagawa, J. K., & Pannek, J. (2018). Differential communication with distributed MPC based on occupancy grid. *Information Sciences*, 453, 426–441. Retrieved from <https://doi.org/10.1016/j.ins.2018.04.034>
- Venkat, A., Rawlings, J., & Wright, S. (2005). Stability and optimality of distributed model predictive control. In *44th IEEE conference on decision and control* (pp. 6680–6685). Seville, Spain: IEEE. Retrieved from <http://dx.doi.org/10.1109/CDC.2005.1583235>
- Worthmann, K. (2012). Estimates of the prediction horizon length in MPC: A numerical case study. *IFAC Proceedings Volumes (IFAC-PapersOnline)*, 45(17), 232–237. Retrieved from <http://dx.doi.org/10.3182/20120823-5-NL-3013.00037>
- Wu, N. (2001). A universal procedure for capacity determination at unsignalized (priority-controlled) intersections. *Transportation Research Part B*, 35, 593–623. Retrieved from [https://doi.org/10.1016/S0191-2615\(00\)00012-6](https://doi.org/10.1016/S0191-2615(00)00012-6)

- Yongjie, Y. (2009). Collision Avoidance Planning in Multi-robot based on Improved Artificial Potential Field and Rules. In *Ieee international conference on robotics and biomimetics* (pp. 1026–1031). Bangkok, Thailand: IEEE. Retrieved from <https://doi.org/10.1109/ROBIO.2009.4913141>
- Zhao, G., & Zhu, M. (2018). Pareto optimal multi-robot motion planning. In *Proceedings of the american control conference* (pp. 4020–4025). Milwaukee, WI, USA: IEEE. Retrieved from <https://ieeexplore.ieee.org/document/8431249>



**Figure A1.** Four holonomic systems applying the priority rule based on  $\Gamma_\ell$ : evolution of  $M_P$  and  $N = 9$  (left). Evolution of  $l_p$  for each robot with  $c = 1.5$  and  $N = 9$  (right).

### Appendix A. Priority rule simulations without memory

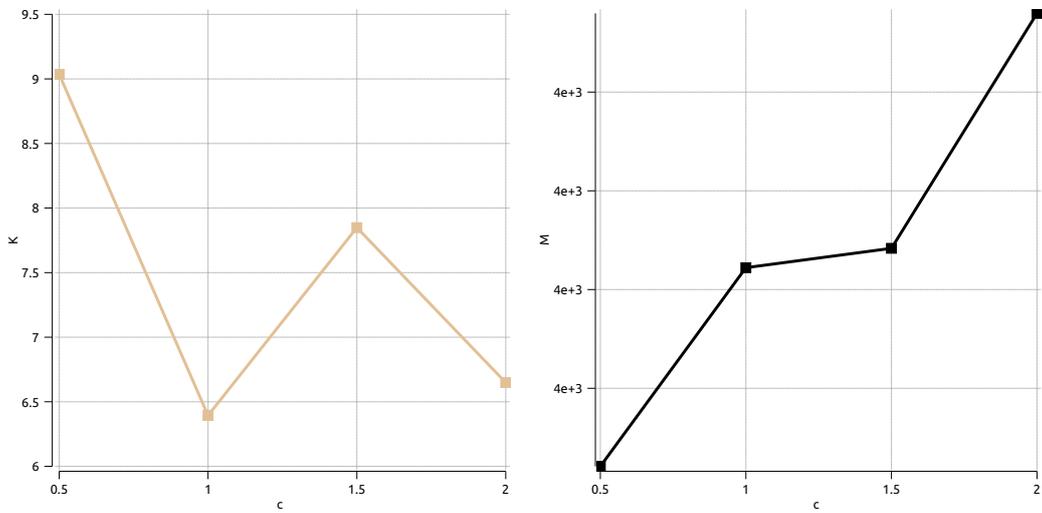
In this section, we examine the priority rules without memory according to Algorithm 2: in each time instant, the priority is evaluated for all robots conducting a sequential execution, where the order of optimising robots is calculated with subject to the chosen criterion values. The closed-loop costs applying the criterion  $\Gamma_\ell$  are illustrated in Fig. A1.

The average communication load over convergence time  $K$  is depicted in Fig. A2.

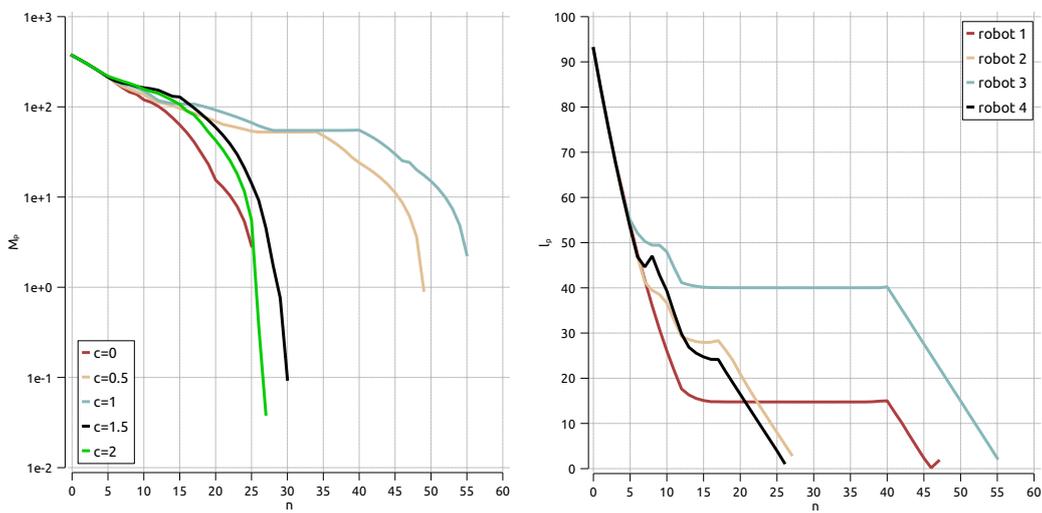
Here, the minimum closed-loop criterion  $\Gamma_\ell$  shows that, in comparison to the fixed priority setting depicted in Table 2 (state-of-the-art column), the convergence times for all cell sizes decrease. The simulation time for middle-sized cell sizes increases due to the elitism of the scheme, that is with a later position in the order, the robot has to obey all constraints from the previous robots and keeps the higher costs, which can be seen for robot 3 in Fig. A1 (right). The communication effort is reduced in comparison to the fixed order setting due to shorter convergence times, see Fig. A2, while with larger cell sizes the number of updated cells also shrinks as the robots need longer times to cross one cell. The cumulated closed-loop costs over the cell sizes in Fig. A2 (right) increases due to longer detours of the robots.

The results for the open-loop minimum costs priority criteria  $\Gamma_V$  are depicted in Fig. A3 for the closed-loop costs.

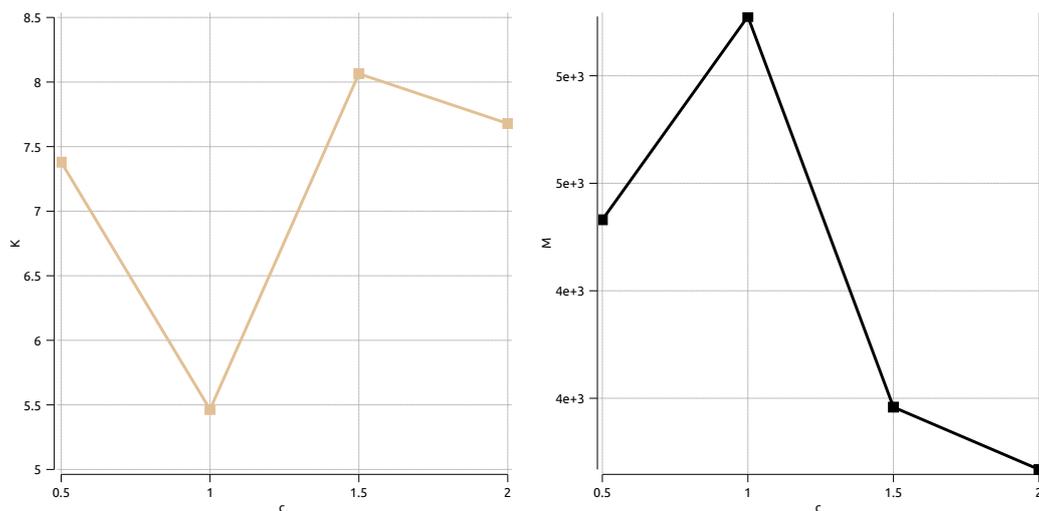
The open-loop criterion  $\Gamma_V$  shows a mixed performance. While for large cell sizes  $c = \{1.5, 2.0\}$  the robots tend to block each other, the horizon has to be extended to  $N = 13$  and despite this, for medium cell sizes the simulation times are even worse. Due to the incapability of the algorithm to avoid deadlocks, robot 1 and 3 are blocking each other as the order is exchanged in every time instant, which shows the need to implement a memory rule to keep a fixed order. The communication effort is depicted in Fig. A4. Here, the simulation time are also higher for middle-sized cell sizes. The average effort over simulation time only increases slightly due to the differential update method. Considering the blockage, only one cell with the new time instant has to be transmitted at least by each robot. Considering the longer simulation times



**Figure A2.** Four holonomic robots ( $N = 9$ ) applying the priority rule based on  $\Gamma_\ell$ : cell size  $c$  vs. communication load  $K$  (left) and closed-loop costs  $M_P^{n\#}$  vs. cell size  $c$  (right).



**Figure A3.** Four holonomic systems applying the priority rule based on  $\Gamma_V$ : evolution of  $M_P$  and  $N = 13$  (left). Evolution of  $l_p$  for each robot with  $c = 1.0$  and  $N = 13$  (right).



**Figure A4.** Four holonomic robots ( $N = 13$ ) applying the priority rule based on  $\Gamma_V$ : cell size  $c$  vs. communication load  $K$  (left) and closed-loop costs  $M_p^{n\#}$  vs. cell size  $c$  (right).

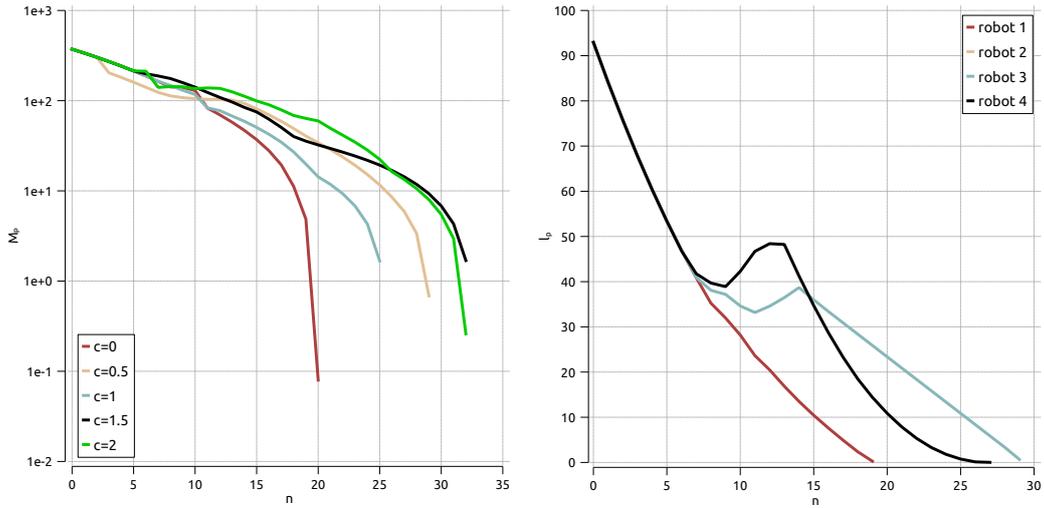
for smaller cell sizes, this increases the cumulated closed-loop costs (Fig. A4 right) for  $c = \{0.5, 1.0\}$ , while due to shorter simulation times for the larger cell sizes, the cumulated costs are smaller although the taken detours of the robots are longer.

## Appendix B. Minimum-costs priority rule simulations with memory

In this section we apply the priority rules combined with a memory to avoid deadlocks or cycles, which may occur as stated in (Pannek, 2013, Example 2). In the first subsection, the algorithm described in Algorithm 3 is complemented by Algorithm 2: If a robot has any active constraints (dependencies) imposed by another robot, the order is kept (dependency blockage). In the second approach, the priorities are stored in hierarchy levels by using priority variables as formulated in Algorithm 6, which allows co-existence of robots on any priority level and therefore parallel execution.

### B.1. Simulations with dependency blockage

The memory function ensures that all robots are kept in the same priority order. If they face a conflict, i.e. share the same cell for the same time instant, a priority order is established among the concerned robots and the calculated order is kept until the constraints are no longer active, i.e. the dependencies no longer exist. In each time instant, each robots tests, if still active constraints exist for any other robot. Are there new conflicts for robots in other blocked groups, all groups are concatenated and sorted after the chosen criterion. If no active constraints for one robot exist, it is unblocked. First, the results for the open-loop criterion  $\Gamma_V$  are presented in Fig. B1, depicting the time evolution of the closed-loop costs. Considering the used memory, which keeps the priority order while the constraints are active, the scenario tends to a stronger elitism for larger cell sizes, as here robot 3 and 4 keep the lower priority until the constraints are no longer active. Comparing this with the priority rule application without memory, even though a stronger elitism occurs, the overall simulation times decreases for all cell

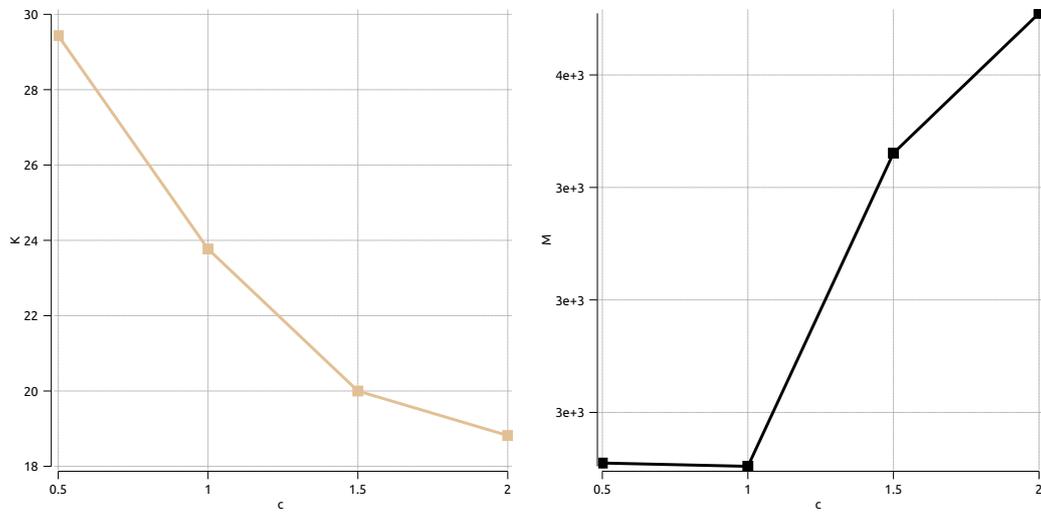


**Figure B1.** Four holonomic systems applying the priority rule based on  $\Gamma_V$  with memory: development of  $M_P$  with  $N = 9$  (left) and of  $l_p$  for each robot with  $c = 0.5$  and  $N = 9$  (right).

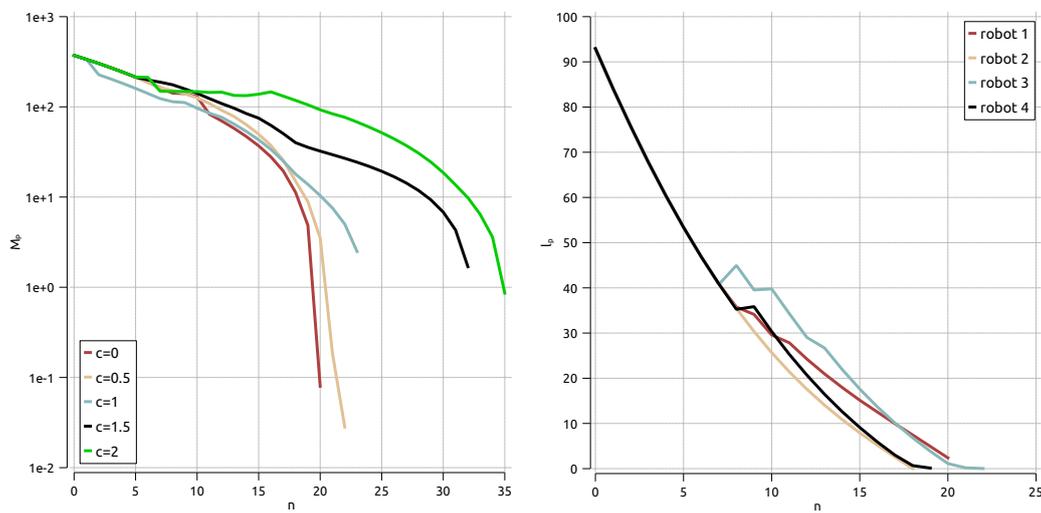
sizes. From the depicted closed-loop costs we can conclude that the incorporation of the memory helps to decrease the convergence times, especially for the middle-sized cell sizes, while for larger cell sizes the convergence times are comparable to the closed-loop costs without memory. The communication effort and the cumulated closed-loop costs are depicted in Fig. B2. The overall communication effort decreases with increasing cell sizes. Again, with less cells, the number of updated cells is reduced. On the other hand, by taking the necessary detours the accumulated closed-loop costs increases with larger cell sizes Fig. B2 (right). In contrast to the previously used priority rule without memory, the messages to exchange the information of the criterion values are incorporated to be counted, which leads to a higher average of communicated tuples, c.f. Fig. A4 (left).

The results for the closed-loop priority criterion  $\Gamma_\ell$  applying the memory rule are presented in Fig. B3 (closed-loop costs). Comparing the results with the priority rule based on the open-loop criterion  $\Gamma_V$ , the simulation times for all cell sizes are slightly lower. A hint is the attenuated effect of elitism, which is presented in the setting for  $c = 2.0$  (compare closed-loop costs in Fig. B1 (right) to B3 (right), extending the simulation times for the applied open-loop criteria  $\Gamma_V$  as in that setting robot 3 reaches the given target later than the other robots.

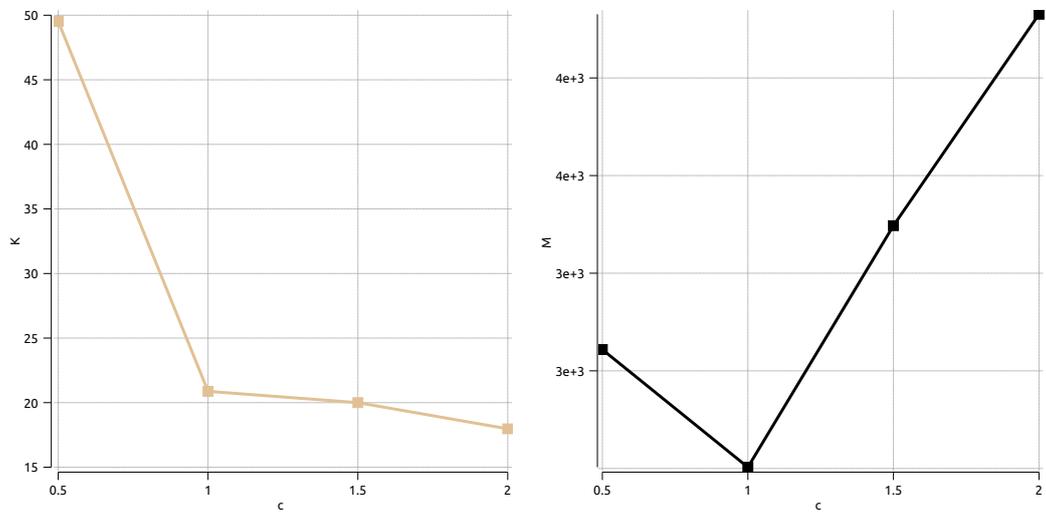
In detail, the simulation time for smaller cell sizes decreases in comparison to the open-loop criteria, which is shown for  $c = \{0.5, 1.0\}$ , compare Fig. B3 to B1 (closed-loop costs). The reason is a higher priority switching that is only the current state (closed-loop criterion  $\Gamma_\ell$ ) is incorporated into the sorting process and not the decreasing cost function over the full horizon length, which induces a higher replanning rate of the robots. With larger cell sizes the open-loop criteria  $\Gamma_V$  keeps its advantage to keep one robot with a higher priority due to its lowest open-loop costs over the full horizon length. The communication effort depicted in Fig. B4 is higher for smaller cell sizes due to more replanning changes resulting from the switches in the priority order, which leads to a higher number of updated cells. For larger cell sizes, this effect is negligible due to higher times for the robots to cross the cells. Hence, the number of communicated cells decreases.



**Figure B2.** Four holonomic robots ( $N = 9$ ) applying the priority rule based on  $\Gamma_V$  with memory: cell size  $c$  vs. communication load  $K$  (left) and closed-loop costs  $M_P^{n\#}$  vs. cell size  $c$  (right).



**Figure B3.** Four holonomic robots with priority rule based on  $\Gamma_\ell$  with memory: development of  $M_P$  with  $N = 9$  (left) and of  $l_p$  for each robot with  $c = 0.5$  and  $N = 9$  (right).



**Figure B4.** Four holonomic robots ( $N = 9$ ) applying the priority rule based on  $\Gamma_\ell$  with memory: cell size  $c$  vs. communication load  $K$  (left) and closed-loop costs  $M_P^{n\#}$  vs. cell size  $c$  (right).

This article is reprinted via the license agreement for doctoral thesis from Taylor & Francis.

# Impact of Quantisation on Consistency of DMPC in Street Traffic with Dynamic Priority Rules

Tobias Sprodowski<sup>1,\*</sup> and Jürgen Pannek<sup>1,2</sup>

<sup>1</sup> University of Bremen, Department of Production Engineering, Hochschulring 20, 28359 Bremen, Germany

<sup>2</sup> BIBA - Bremer Institut für Produktion und Logistik GmbH, Hochschulring 20, 28359 Bremen, Germany

Distributed systems typically require communication to steer the overall system to a reference or target state and keep it there. We consider an intersection scenario as a spacial set with vehicles as multi-agents with individual initial and target conditions. Utilising a Distributed Model Predictive Control scheme (DMPC), in every time step each vehicle solves a finite horizon optimal control problem. The resulting optimal state trajectory is projected onto a quantisation of the spatial set and the quantised trajectories are broadcasted to the other vehicles to ensure collision avoidance. The quantisation is mainly motivated to reduce the necessary communication effort among the agents. Here, we introduce prediction coherence as difference of two predictions in two successive time steps. We numerically explore the idea to utilise prediction coherence to derive a lower bound for the communication requirements among the vehicles.

Copyright line will be provided by the publisher

## 1 Introduction

Today, distributed control is a widely common method to reduce complexity by splitting a central problem into subproblems. For distributed systems subject to couplings in constraints or dynamics, e.g. formation control of robots [1], traffic control [2], the drawback of this approach is that communication among subsystems is required to steer the system to a target state [3]. Here, one distinguishes between cooperative or non-cooperative control. In the first approach, a common overall objective to steer the system to a target state is given. The latter approach, stemming from game theory [4], relies on individual goals for each subsystem without considering a common global objective.

In our case, an intersection scenario with non-cooperative vehicles is considered, where each vehicle exhibits individual initial and target positions. We apply a Distributed Model Predictive Control (DMPC) scheme [5] to obtain a feedback control for each vehicle in a fixed order, which shall steer each subsystem to its target. Similar to approaches in the literature [6], each vehicle is subject to coupling constraints ensuring collision avoidance, which involves the received information from other vehicles. Here, we suppose that within each optimal control problem the state is an element of a continuous set while the communication data is quantised to reduce the communication burden, cf. [7] for details: The predicted state trajectories are projected onto a grid with equidistant cells and then broadcasted to the other vehicles. Based on a traffic scenario [8], we aim to utilise the property that two successive MPC predictions are typically close to one another. We refer to this similarity as prediction coherence to conjecture that this property can be used to derive a lower bound for the necessary communication bandwidth among the vehicles. Here, we numerically explore this idea in a fixed and dynamic order setting with varying the cell size.

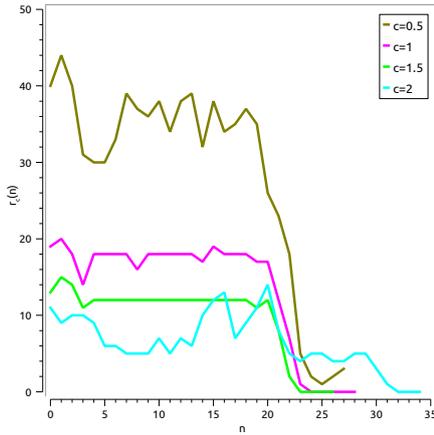
## 2 Problem Setting

We consider a group of  $P \in \mathbb{N}$  vehicles, where each vehicle  $p \in P$  is modelled as a time discrete control system  $z_p^+ = f(z_p, u_p)$  where  $z_p \in Z \subset \mathbb{R}^2$  describes the current state in planar coordinates  $(x_p, y_p)^\top$  of vehicle  $p$  and  $u_p \in U \subset \mathbb{R}^2$  the control of vehicle  $p$ . State and control constraints are bounded by  $Z := [-\bar{x}, -\bar{y}] \times [\bar{x}, \bar{y}]$  and  $U := [-\bar{v}^x, \bar{v}^x] \times [-\bar{v}^y, \bar{v}^y]$ . The spatial set is quantised into a grid  $\mathcal{G} \subset \mathbb{N}^2$  of squared cells with  $2\bar{x} = a_{\max}c$  and  $2\bar{y} = b_{\max}c$  and  $\mathcal{G} := [0, \dots, a_{\max}] \times [0, \dots, b_{\max}]$ . A control sequence of a vehicle over a prediction horizon  $N$  is defined as  $u_p := (u_p(0), \dots, u_p(N-1))$ . The state trajectory of a vehicle based on an initial condition  $z_p^0$  is defined as  $z_p^u(\cdot; z_p^0) := (z_p^u(0; z_p^0), \dots, z_p^u(N; z_p^0))$ . The states are quantised with  $q(z_p) = (a_p, b_p) := \left( \left\lfloor \frac{x_p + \bar{x}}{c} \right\rfloor, \left\lfloor \frac{y_p + \bar{y}}{c} \right\rfloor \right)$  to obtain the cell indexes  $(a_p, b_p)$ . We abbreviate the predictions for comparison in two successive time instants of one vehicle by  $z_p^u(\cdot) := z_p^u(\cdot; z_p(n))$  and  $\tilde{z}_p^u(\cdot) := z_p^u(\cdot + 1; z_p(n-1))$ . The *prediction coherence*  $r_p$  is then defined as a measure of two state trajectories at successive time instants of a vehicle  $p$  via

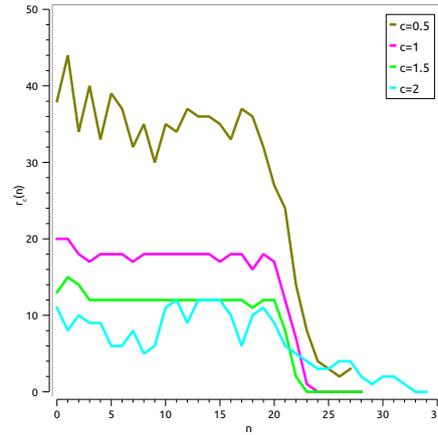
$$r_p(q(z_p^u(\cdot)), q(\tilde{z}_p^u(\cdot))) := \sum_{k=0}^{N-1} \|q(z_p^u(k)) - q(\tilde{z}_p^u(k))\|_2, \quad r_c(n) := \sum_{p=1}^P r_p(q(z_p^u(\cdot)), q(\tilde{z}_p^u(\cdot))) \quad (1)$$

where  $r_c$  denotes the cumulated prediction coherence over all vehicles for a time instant  $n$ . The difference can be interpreted as the required communication among the vehicles as this induces the changes on the coupling constraints among them.

\* Corresponding author: e-mail spr@biba.uni-bremen.de, phone +49 421 218 50097



**Fig. 1:** 8 holonomic systems with fixed priority ( $N = 4$ ): time evolution of  $r_c(n)$



**Fig. 2:** 8 holonomic systems with dynamic priority ( $N = 4$ ): time evolution of  $r_c(n)$

We apply the DMPC algorithm from [5] sequentially for each vehicle, i.e., in each step a vehicle solves an optimal control problem based on the initial and target positions as well as on the received information from other vehicles. Then, only the first value of the calculated optimal control sequence  $u_p(0)$  is applied and the horizon is shifted forward which renders the procedure to be iteratively applicable. In contrast to the fixed priority rule utilized in [5], we also adapt the priority dynamically in each time instant  $n$  based on the minimum closed-loop costs  $\ell_p : Z \times U \rightarrow \mathbb{R}_{\geq 0}$  with  $\min(\ell_1(z_1(n), u_1(n)), \dots, \ell_P(z_P(n), u_P(n)))$ . With subject to ascending costs, we define the sort function  $s : \mathbb{R}^P \times \mathbb{N} \rightarrow \mathbb{R}^P \times \mathbb{N}$  with  $s((1, \ell_1(z_1(n), u_1(n))), \dots, (P, \ell_P(z_P(n), u_P(n)))) = (g_1, \ell_1(\cdot, \cdot), \dots, (g_P, \ell_P(\cdot, \cdot)))$  where  $g$  denotes the position in the optimisation sequence.

### 3 Results

To explore the prediction coherence (1) which illustrates the minimum communication bandwidth for a chosen cell size, we compare the fixed and dynamic priority rules regarding minimum costs in Figures 1 and 2 by illustrating the time evolution of  $r_c(n)$ . The maximum value of each curve represents the minimum communication requirements among the vehicles in relation to the chosen cell size  $c$ . Here, we observe that for an increasing cell size  $c$  of the grid, the decrease in bandwidth requirements is slowing down drastically, which indicates that a lower bound may be derived from this measure. As with larger cell sizes the collision avoidance constraints become more conservative, longer detours extend the simulation time. Applying the dynamic priority rule reduces the communication requirements for large cell sizes ( $c = 2.0$ ) due to the fact, that the system tends to elitism and for vehicles with minimum costs the prediction differs as they face a high probability to retain their first positions in the optimisation order. In opposite, the subsequent vehicles have to face more necessary adaptations in their predictions.

### References

- [1] M. W. Mehrez, G. K. I. Mann, and R. G. Gosine, Formation stabilization of nonholonomic robots using nonlinear model predictive control, in: IEEE 27th Canadian Conference on Electrical and Computer Engineering (CCECE), (IEEE, Toronto, Ontario, Canada, 2014), pp. 1–6.
- [2] E. Camponogara and M. L. de Lima, IEEE Trans. Autom. Control **57**(3), 804–809 (2012).
- [3] A. Venkat, J. Rawlings, and S. Wright, Stability and optimality of distributed model predictive control, in: Proceedings of the 44th IEEE Conference on Decision and Control, (IEEE, Seville, Spain, 2005), pp. 6680–6685.
- [4] H. Brock, A generalization and partial reconceptualization of the theory of non-cooperative games, in: Proceedings of the Decision and Control Conference including the 12th Symposium on Adaptive Processes, (IEEE, San Diego, CA, USA, 1973), pp. 286–293.
- [5] A. Richards and J. How, A decentralized algorithm for robust constrained model predictive control, in: Proceedings of the American Control Conference (ACC), (IEEE, Boston, MA, USA, 2004), pp. 4261–4266.
- [6] L. Makarem and D. Gillet, Model predictive coordination of autonomous vehicles crossing intersections, in: IEEE Conference on Intelligent Transportation Systems, Proceedings, (IEEE, The Hague, Netherlands, 2013), pp. 1799–04.
- [7] M. W. Mehrez, T. Sprodowski, K. Worthmann, G. K. I. Mann, R. G. Gosine, J. K. Sagawa, and J. Pannek, Occupancy Grid based Distributed Model Predictive Control of Mobile Robots, in: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) - accepted, (IEEE, Vancouver, Canada, 2017).
- [8] T. Sprodowski, J. K. Sagawa, and J. Pannek, Impact of quantization on consistency of distributed model predictive control in street traffic, in: 20th IFAC World Congress (accepted), (IFAC-PapersOnLine, Toulouse, France, 2017).

This article is reprinted via the license agreement for doctoral thesis via Wiley-VCH Verlag GmbH & Co. KGaA, Weinheim, ©2017



Article

# Analytical Aspects of Distributed MPC Based on an Occupancy Grid for Mobile Robots

Tobias Sprodowski <sup>1,\*</sup>  and Jürgen Pannek <sup>1,2</sup> 

<sup>1</sup> Faculty of Production Engineering, Hochschulring 20, University of Bremen, 28359 Bremen, Germany

<sup>2</sup> BIBA Bremer Institut für Produktion und Logistik GmbH, Hochschulring 20, 28359 Bremen, Germany; pan@biba.uni-bremen.de

\* Correspondence: spr@biba.uni-bremen.de; Tel.: +49-421-218-50097

Received: 7 January 2020; Accepted: 30 January 2020 ; Published: 4 February 2020



**Abstract:** In this paper, we evaluate theoretical aspects of a distributed system of noncooperative robots controlled by a distributed model predictive control scheme, which operates in a shared space. Here, for collision avoidance, the future predicted state trajectories are projected on a grid and exchanged via discrete cell indexes to reduce the communication burden. The predicted trajectories are obtained locally by each robot and carried out in the continuous space. Therefore, the quantisation does not impose the quality of the solution. We derive sufficient conditions to show convergence and practical stability for the distributed control system by using an idea of a temporary roundabout derived from crossing patterns of street traffic rules, which is established in a fixed and flexible circle size. Furthermore, a condition for the sufficient prediction horizon length to recognise necessary detours is presented, which is adapted for the occupancy grid. The theoretical results match with the trajectory patterns from former numerical simulations, showing that this pattern is naturally chosen as an overall solution.

**Keywords:** distributed model predictive control; quantisation; stability; prediction horizon length; mobile robots

## 1. Introduction

Today, control problems with high dynamics have been of high interest in production and logistic processes or in traffic scenarios. In manufacturing processes, the technical advances have increased the utilisation of robots, especially in assembly lines in the last decades [1]. In logistic processes, autonomous, connected pallet carriers for material have increasingly been used to supply production lines with material. In common traffic scenarios, autonomous connected cars (or mobile robots) have aggregated different research fields, e.g., sensor fusion, modelling, and handling communication. One central issue is to be able to calculate a feasible solution which ensures safety in terms of collision avoidances. Examples for robotic systems can be found in formation control of robots [2], exploration of unknown environment to achieve an optimal covering with limited number of robots [3], or real-time control of distributed helicopters [4]. All these distributed systems are linked by common properties and arising problems: complexity and availability of information. Concerning complexity, these system have to be handled in a distributed manner. One approach is to use multi-agent systems to decrease the complexity and to enable the distributed system to calculate a solution sufficiently fast. The available computational capabilities on distributed systems like robots or cars ease this approach.

Additionally, the distributed system is relying on the requirement of an information exchange: In contrast to a central system, where the information about the full system state is globally available, here, each agent has to rely on local information (decentralised system) or some communication methods have to be implemented such that the agents are able to gather information from neighbours

or aggregated information about the full system state. If stability is taken into account, such information exchange is essential to achieve and guarantee that stability could be kept; see References [5–7].

The scenario here describes a group of robots in a shared operation space (spacial set), which shall be steered to individual targets while ensuring collision avoidance. This is implemented in a noncooperative control scheme, where each robot optimises a local optimal control problem (OCP) to minimise a cost function regarding the current distance to an individual target. Therefore, additional communication among the robots is necessary to inform the other robots by the calculated prediction to let them regard the collision avoidance via coupling constraints in the OCP. On the other hand, cooperative control schemes are targetting a common goal, which could be imposed for example via a common objective; see Reference [8] for flight formation of robots. The optimisation is executed in the spacial set, while for the communication among the robots, a quantisation of the continuous space to a grid with equidistant cells is established. The predicted trajectories are quantised obtaining grid indices and used to communicate them between the robots. This approach, introduced in References [9,10], allows to keep the problem suitable for optimisation algorithms in a continuous setting while taking advantage in the communication reduction via discretisation. The optimisation and therefore the communication is carried out in a sequential manner according to Reference [11]; the future predicted state trajectories of each robot (here, discrete occupancy tuples) are communicated iteratively for each time instant. While in the former work [9,10] conditions considering initial and recursive feasibility were examined, the focus here is set to derive conditions for convergence and practical stability of the overall system.

We focus on theoretical properties of this approach to ensure two aspects: deadlock-free execution and convergence. Considering convergence properties in Distributed Model Predictive Control (DMPC), many advances were made in the last years, c.f. Reference [12]. For cooperative schemes, in Reference [13], the authors introduced DMPC with bounded disturbances where the bound of the disturbance is time varying. For both decentralised and distributed versions, feasibility was shown. Several cooperative control schemes were utilised to show these properties: the authors of Reference [5] used terminal costs to speed up the convergence rate while assuming bounded interactions among the subsystems. In Reference [14], a cooperative control was applied where the subsystems optimise sequentially with predetermined terminal regions and cost functions to achieve feasibility and convergence. The authors of Reference [15] added a global stability constraint to show convergence, which is locally impaired by each subsystem. In Reference [16], the authors address linear networked systems in cooperative control regarding the consensus via static couplings, which are imposed on the input of each subsystem from the output of other systems. For gradient-based DMPC schemes based on dual decomposition, an upper bound on the system dynamics is used to improve the convergence rate [17]. The results were extended in Reference [18] to general DMPC problems, and convergence is shown via a stopping criterion which computes lower and upper bounds when a sufficient control is found to keep the system in the desired state. Considering noncooperative schemes, a global objective was completely decomposed in Reference [19] and applied on network traffic control. Moreover, the control values of all affected neighbours were considered in a proposed communication protocol to obtain a fixed decision order for execution. In Reference [20], an iterative exchange of information in each time instant was not needed, as reference trajectories are provided for each subsystem while convergence is ensured via terminal constraints. The authors of Reference [21] utilised the scheme of Reference [11] to show feasibility and convergence without using terminal costs or constraints. Therefore, asymptotic controllability of the distributed system was assumed to derive stability properties. Instead of using a fixed optimisation order, in Reference [22], a dynamic deordering rule was presented to iteratively change the order dynamically on an abstract criterion. Convergence was shown based on a relaxed Lyapunov condition. A leader–follower system with flexible leader was considered in Reference [23] with an arbitrary chosen convex hull. The sequential order of the execution of optimisation is fixed, while small perturbations were incorporated. Regarding slow dynamics, in Reference [24] a quantisation scheme was applied for the communication among subsystems which

allow slow variation of the quantisation interval. Here, the quantisation parameters are exchanged among the subsystems to establish terminal regions which ensured that, from a warm start initial condition, the equilibrium was reached within finite time.

Other approaches to stabilise nonlinear systems used Lyapunov-based controllers to keep the system stable with tight state and input constraints, while relaxations on the states were possible via a switched control [25]. On the distributed level, in Reference [26], the Lyapunov controllers were executed in either a sequential or iterative manner while stability was ensured by a cost decrease constraint. In Reference [27], the authors relaxed the Lyapunov scheme via a time delay to allow for a temporary increase of costs. The authors in Reference [28] applied a Lyapunov controller with a global bounded state feedback controller for a tracking problem of a non-holonomic robot.

In these articles, the convergence or stability aspects were shown by utilising either a Lyapunov function or a controller, which guarantees the decrease of costs in any time instant, or of terminal constraints or costs, which then forces the predicted trajectory to achieve an end point or region, or to steer that by a terminal cost penalty or a connective constraint, ensuring that, e.g., a minimum distance or explicit bounds on the optimal value function were used to calculate the suboptimality degree of a finite horizon controller. As the considered scenario here is noncooperative and as a Lyapunov function for one robot may be infeasible if one robot has to increase its costs by taking a detour and if terminal regions or costs will restrict the freedom of feasible trajectories, we use a weaker assumption, which is similar to the movement pattern from former numerical results based on the pattern of a roundabout; see Figures 4 and 7 in Reference [10].

We focus on a noncooperative setting as each robot exhibits an individual target without imposing terminal costs or constraints following Reference [21]. In this article, we derive sufficient conditions to avoid blockings or deadlocks with the assumption that the robots optimise in a fixed order in the spacial set with quantised communication. We avoid imposing Lyapunov conditions for the cost function, which would either impose additional constraints for the OCP or terminal costs. Hence, we implement a switched control, which is inspired from street traffic rules for a roundabout structure. If the robots are below a certain distance to each other, a circle with the mean-based middle point based on the end points of the predicted trajectories of the affected systems is created. Each participating system calculates individual intersection points with this circle and follows the circular curve via a circular control law until the original control law could be reinstated again. Moreover, as a second aspect, we derive a sufficient condition for the prediction horizon length based on a given cost function to examine the sufficient length, which allows the robot to recognise the decrease of costs to take the detour instead of holding their positions probably infinitely long.

The paper is structured as follows: In the subsequent section, the problem is stated with the model of the robots including state and control constraints. We recap the construction of the occupancy grid shortly and give an outline on the construction of the constraints including a safety margin regarding the quantised communication. Then, these parts are assembled to define an OCP in Section 3, which is incorporated in the DMPC scheme on each robot. The theoretical contributions concerning a sufficient prediction horizon length and the convergence to achieve practical stability are presented in Section 4. For the convergence part, we consider two possibilities: First a fixed circle size is considered to possibly include all robots on the circle. Second, we extend this approach to a flexible circle, which considers the maximum size without interfering with targets of other robots. To complete the article, a short conclusion and outlook on future extensions is given.

**Notation:**  $\mathbb{R}$  and  $\mathbb{N}$  denote the real and natural numbers, respectively.  $\mathbb{N}_0 := \mathbb{N} \cup \{0\}$  represents the nonnegative integers, and  $\mathbb{R}_{\geq 0}$  represents the nonnegative real numbers. For integers  $a, b \in \mathbb{N}_0$  with  $a \leq b$ , the expression  $[a : b]$  denotes the sequence  $\{a, a + 1, \dots, b\}$ . Moreover, for a vector  $x \in \mathbb{R}^n$ ,  $n \in \mathbb{N}$ , we define the infinity norm  $\|x\|_\infty := \max_{i \in [1:n]} |x_i|$ .

## 2. Problem

Based on Reference [10], we introduce the problem and necessary definitions for the robots, occupancy grid, and constraint construction shortly: Here, we utilise a distributed system of  $P \in \mathbb{N}$  mobile robots. The robots follow a time discrete model stated as a control-affine system:

$$z_p^+ = f(z_p, u_p) := f_0(z_p) + \sum_{i=1}^m f_i(z_p) u_{p,i}, \quad p \in [1 : P] \quad (1)$$

with smooth vector fields  $f_0$  and  $f_i: \mathbb{R}^d \rightarrow \mathbb{R}^d$ ,  $d \geq 2$ ,  $i \in \{1, \dots, m\}$ ,  $d, m \in \mathbb{N}$  where, for each vector field  $f_i$ ,  $i \in \{1, \dots, m\}$ , one control value  $u_{p,i}$  is imposed. A successive state of a robot is denoted by  $z_p^+$  and determined by  $f_i$  based on its current state  $z_p$  and imposed control  $u_p$ . The state is represented here without loss of generality by the planar coordinates  $(x_p, y_p)^\top = z_p \in \mathbb{R}^d$ . Constraints for state and control are given by

$$z_p \in Z := [-\bar{x}, \bar{x}] \times [-\bar{y}, \bar{y}] \times \tilde{Z} \quad \text{with } \tilde{Z} \subseteq \mathbb{R}^{d-2} \text{ and } \bar{x}, \bar{y} > 0, \quad (2)$$

$$u_p \in U \subset \mathbb{R}^m, \quad (3)$$

respectively, with  $U$  containing the origin.

Assuming that our systems in Equation (1) are driftless ( $f_0 \equiv 0$ ), describing the kinematic model of a mobile robot, we assume that robots may immediately stop by the following assumption:

**Assumption 1** (Immediate Hold). *Each robot can come to an immediate hold:*

$$\forall z_p \in Z \quad \exists \bar{u}_p \in U \quad : \quad z_p = f(z_p, \bar{u}_p). \quad (4)$$

Note that delays of actuator dynamics, which might invalidate this assumption are not taken into account. For a finite horizon  $N$  and individual initial positions  $z_p(0)$ , for each robot  $p$ , a finite control sequence is defined by

$$\mathbf{u}_p = (u_p(0), u_p(1), \dots, u_p(N-1)),$$

which is utilised to formulate the predicted state trajectory of robot  $p$  as

$$(z_p^u(0; z_p^0), z_p^u(1; z_p^0), \dots, z_p^u(N; z_p^0)).$$

### 2.1. Occupancy Grid and Quantised Communication

Based on a spacial set given by  $[-\bar{x}, \bar{x}] \times [-\bar{y}, \bar{y}]$ , we construct the occupancy grid by partitioning the spacial set into equidistant cells:  $\mathcal{G} := [0 : a_{\max} - 1] \times [0 : b_{\max} - 1] \subset \mathbb{N}_0^2$  with cell width (= height)  $c$ . The partitioning is defined by  $\bar{x}$  and  $\bar{y}$ , both multipliers of  $c$  with

$$a_{\max} = \frac{2\bar{x}}{c} \quad \text{and} \quad b_{\max} = \frac{2\bar{y}}{c}$$

where  $\bar{x}$  and  $\bar{y}$  are both multiples of  $c$ . Furthermore, we assign an index  $(a, b) \in \mathcal{G}$  to each cell; see Figure 1 for details.

For the quantisation of communication among the robots, we define  $q: Z \rightarrow \mathcal{G}$ :

$$q(z_p) = (a_p, b_p) := \left( \left\lfloor \frac{x_p + \bar{x}}{c} \right\rfloor, \left\lfloor \frac{y_p + \bar{y}}{c} \right\rfloor \right), \quad (5)$$

to map a state  $z_p$  onto the grid  $\mathcal{G}$ . Now, this enables robot  $p$  to broadcast the occupied cells instead of a continuous predicted state trajectory as a sequence of *occupancy tuples*  $\mathcal{I}_p(n) \in (\mathbb{N}_0 \times \mathcal{G})^{N+1}$  at time  $n$  with

$$\mathcal{I}_p(n) := \left( q \left( z_p^u \left( n+k; z_p^0 \right) \right) \right)_{k=0}^N.$$

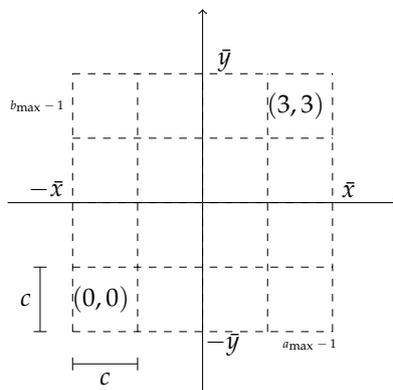


Figure 1. Example of a grid  $\mathcal{G}$  with  $a_{\max} = b_{\max} = 4$  from Reference [9].

All received occupancy grid tuples  $\mathcal{I}_p(n)$  from the other robots  $q$  with  $q \in [1 : P] \setminus \{p\}$  are assembled to

$$i_p(n) := (\mathcal{I}_1(n), \dots, \mathcal{I}_{p-1}(n), \mathcal{I}_{p+1}(n), \dots, \mathcal{I}_P(n)) \tag{6}$$

and denoted as the *occupancy grid*. To convert a quantised state back to obtain the spacial coordinates, we require the backward mapping  $h: \mathcal{G} \rightarrow \mathbb{R}^2$  by

$$h(a, b) = ((a + 0.5)c - \bar{x}, (b + 0.5)c - \bar{y})^\top \tag{7}$$

which maps a cell index  $(a, b) \in \mathcal{G}$  to the spatial coordinates of the cell centre ignoring in both cases the time index  $n$ . This is then used in the following subsection to construct the coupling constraints, which are later used in the OCP of each robot.

### 2.2. Construction of the Constraints

The mobile robots share a collective operation space defined by  $Z$ . As their trajectories may cross each other, we have to ensure collision avoidance regarding two aspects: The first aspect is a minimum cell size of the occupancy grid to prevent robots from skipping cells or swapping cells with another robot. As the dynamics of the robots is discretised with a fixed sampling step, an overlapping of two trajectories may occur within two time instants, leading to an unnoticed crossing of robots. Hence, a minimum cell size to prevent this is imperative. The second aspect stems from a necessary physical distance among the robots and the adherence of the constraints in the discrete time instants and in between two successive time instants.

**Cell size:** In Reference [10], we proposed a minimum cell size  $\underline{c}$  based on the maximum control value which can be imposed on the robot. This minimum cell size has to cover all possible moves of the robot in  $Z$ ; therefore, it is lower bounded for each vector field  $f_0$  and  $f_i$  with  $i \in \{1, \dots, m\}$

$$\underline{c} := \max_{u_p \in \mathcal{U}} \left\{ c_{f_0} + \sum_{i=1}^m c_{f_i} |u_{p,i}| \right\} \tag{8}$$

where each  $c_i$  with  $i \in \{0, \dots, m\}$  covers the dimensions with

$$c_{f_0} := \max_{z_p \in Z} \left\{ \max_{j \in \{1,2\}} |(f_0(z_p) - z_p)_j| \right\} \text{ and } c_{f_i} := \max_{z_p \in Z} \left\{ \max_{j \in \{1,2\}} |f_{i,j}(z_p)| \right\}.$$

Here, obeying the kinematic model of the robot, this is sufficient to avoid skipping cells.

**Safety margin in discrete time instants:** Considering the safety margin, we have to ensure that each robot keeps a physical minimum distance to other robots, which is given by  $d_{min}$ . As the minimum cell size can be smaller or larger than the necessary physical distance among the robots, we define the safety margin as follows:

**Definition 1** (Safety Margin). *The safety margin among any two robots  $p, q \in P, p \neq q$  is given by the necessary physical minimum distance  $d_{min}$  and the given minimum cell size  $\underline{c}$  such that*

$$\left\| \begin{pmatrix} x_p \\ y_p \end{pmatrix} - \begin{pmatrix} x_q \\ y_q \end{pmatrix} \right\|_{\infty} \geq \max\{d_{min}, \underline{c}\} \quad (9)$$

holds. Here,  $(x_i, y_i)^T = z_i, i \in \{p, q\}$  denote the planar positions of the robots.

Furthermore, we have to ensure that this safety margin holds over all predicted states  $z_p^u(k; z_p^0)$  with  $k \in [1 : N]$  of one robot  $p$  to any other robot  $q$ . As the information of the states of robot  $q$  is given via the quantised occupancy tuples  $\mathcal{I}_q(n)(k) := q(z_q^u(n+k; z_q^0))$  for  $k \in [1 : N]$ ; for a given cell size  $c \geq \underline{c}$ , a robot may be in the centre of the communicated cell or on the boundary. Therefore, we obtain the maximal deviation of the quantised state to the continuous state of the robot by

$$\left\| \begin{pmatrix} x_q \\ y_q \end{pmatrix} - h(q(z_q)) \right\|_{\infty} \leq \frac{c}{2}.$$

This leads to the definition of the safety margin  $\Psi$  regarding the quantised error:

**Proposition 1** (Quantised Prediction Safety Margin). *For a given cell size  $c \geq \underline{c}$ , minimum distance among robots  $d_{min}$ , and discrete time instants  $k$  with  $k \in [1 : N]$ , the safety margin between two robots  $p, q \in P, p \neq q$  based on the received quantised state  $\mathcal{I}_q(n)(k)$  from robot  $q$  is*

$$\left\| \begin{pmatrix} x_p(k) \\ y_p(k) \end{pmatrix} - h(\mathcal{I}_q(n)(k)) \right\|_{\infty} \geq \max\{d_{min}, \underline{c}\} + \frac{c}{2} =: \Psi. \quad (10)$$

If this holds, then Definition 1 is satisfied with

$$\begin{pmatrix} x_p \\ y_p \end{pmatrix} = \begin{pmatrix} x_p(k) \\ y_p(k) \end{pmatrix}, \quad \forall k \in [1 : N]. \quad (11)$$

The details of the proof utilising the triangle inequality are given in Reference [10].

**Proposition 2** (Intermediate Safety Margin). *Suppose that Equation (9) holds for two robots  $p, q \in P, p \neq q$  for  $c \geq \underline{c}$ . Then, the minimum safety margin to prevent intermediate constraint violation in two successive time instants  $k-1, k \in [1 : N]$  is*

$$\Psi_q(\mathcal{I}_q(n)(k), \mathcal{I}_q(n)(k-1)) = \begin{cases} \overline{\Psi} := \Psi + \underline{c}, & \text{if } \mathcal{I}_q(n)(k) \neq \mathcal{I}_q(n)(k-1) \\ \underline{\Psi} := \Psi + \frac{c}{2} \cos\left(\frac{\pi}{4}\right), & \text{else.} \end{cases} \quad (12)$$

Again, the details of the proof are given in Reference [10].

**Derivation of the Constraints:** Now, we utilise the received occupancy tuples  $\mathcal{I}_q(n), q \in [1 : P] \setminus \{p\}$  of the other robots to construct the coupling constraints including the safety margin  $\Psi_q$ . Here, the infinity norm is used as this approximate the geometry of the squared cells.

Based on the received information, we map the state back to the continuous spacial set via  $h(\mathcal{I}_q(n)(k))$ . Given the safety margin  $\Psi_q(\mathcal{I}_q(n)(k), \mathcal{I}_q(n)(k-1))$ , the coupling constraint for robot  $p$  regarding robot  $q$  is defined as

$$g_{q,k}^p := g(z_p^u(k; z_p^0), \mathcal{I}_q(n)(k), \mathcal{I}_q(n)(k-1)) \tag{13}$$

$$= \left\| \begin{pmatrix} x_p(k) \\ y_p(k) \end{pmatrix} - h(\mathcal{I}_q(n)(k)) \right\|_\infty - \Psi_q(\mathcal{I}_q(n)(k), \mathcal{I}_q(n)(k-1)) \geq 0 \tag{14}$$

for all  $k \in [1 : N]$ . Note that, although the infinity-norm describes the property of the occupancy grid very well, it is not differentiable. Hence, a gradient-free optimisation method should be used, which is in general more time consuming than gradient-based methods; see, e.g., Reference [29].

For all robots  $q$  with  $q \neq p$ , the received information is condensed in  $i_p(n)$ . Based on the coupling constraints obeyed by robot  $p$ , the combined constraints are given by

$$G(z_p^u(k; z_p^0), i_p(n)(k), i_p(n)(k-1)) = (g_{1,k}^p \dots g_{p-1,k}^p g_{p+1,k}^p \dots g_{P,k}^p) \tag{15}$$

for  $g_{i,k}^p \geq 0$  with  $i \in (1, \dots, p-1, p+1, \dots, P)$  and for  $k \in [1 : N]$ . If these constraints are satisfied, the quantised prediction safety margin based on the information  $i_p(n)$  and in turn Definition 1 is ensured.

### 3. Distributed Model Predictive Control

To complete the requirements to state an OCP, we introduce a strict convex stage cost function  $\ell_p: Z \times U \rightarrow \mathbb{R}_{\geq 0}$ , which is a positive definite with respect to an individual target  $z_p^* \in Z$  for each robot. In this distributed setting, each robot is steered by a controller which solves an OCP over a finite horizon length  $N$  based on the measured state. Then, after solving its OCP, the first control value of the obtained control sequence is applied, and the procedure is repeated until a termination condition is met, i.e., the target  $z_p^*$  is reached. To this end, the overall OCP is stated as

$$\min_{\mathbf{u}_p} J_p^N(\mathbf{u}_p; z_p^0, i_p(n)) := \sum_{k=0}^{N-1} \ell_p(z_p^u(k; z_p^0), u_p(k)) \tag{16}$$

subject to

$$\begin{aligned} z_p^u(k+1; z_p^0) &= f(z_p^u(k; z_p^0), u_p(k)), & k \in [0 : N-1], \\ u_p(k) &\in U, & k \in [0 : N-1], \\ G(z_p^u(k; z_p^0), i_p(n)(k), i_p(n)(k-1)) &\geq 0, & k \in [1 : N], \\ z_p^u(k; z_p^0) &\in Z, & k \in [1 : N], \\ z_p^* &\in Z, \\ z_p(0) &\in Z \end{aligned} \tag{17}$$

regarding the given constraints concerning the kinematic model, control, state constraints, and coupling constraints induced by the information of the other robots and by individual initial conditions and targets. By solving the OCP, an optimal control sequence

$$\mathbf{u}_p^* = \left( u_p^*(0), \dots, u_p^*(N-1) \right), \quad (18)$$

is obtained, where uniqueness is ensured by the strict convexity of the cost function  $\ell_p$ . We introduce the corresponding value function as  $V_p^N : Z \times (\mathbb{N}_0 \times \mathcal{G})^{(P-1)(N-1)} \rightarrow \mathbb{R}_{\geq 0}^+$  via

$$V_p^N \left( z_p^0, i_p(n) \right) = J_p^N \left( \mathbf{u}_p^*; z_p^0, i_p(n) \right). \quad (19)$$

The DMPC scheme, which is executed by each robot based on the scheme by References [30,31], is presented in algorithmic form and is divided in an initialisation phase (Algorithm 1) and an execution phase (Algorithm 2).

---

**Algorithm 1** DMPC initialisation for the overall system
 

---

```

1: Given admissible states  $z_p$  for initial states  $z_p^0$  for all  $p \in [1 : P]$ 
2: for  $p = 1$  to  $P$  do
3:   for  $k = 1$  to  $N$  do
4:     Set  $\mathcal{I}_p(0)(k) := (k, q(z_p^0))$ 
5:   end for
6:   Broadcast  $\mathcal{I}_p(0)$ 
7:   Set  $w_p = false$ 
8: end for
9: Set  $\mathbf{Q} = \emptyset$ 

```

---

In Algorithm 1, every robot keeps its initial state in the first time instant and broadcasts this before the OCP is solved.  $\mathbf{Q}$  and  $w_p$  are needed later for the derivation of conditions for convergence and practical stability.

---

**Algorithm 2** DMPC execution for the overall system
 

---

```

1: Call Algorithm 1
2: for  $n = 0, 1, \dots$  do
3:   for  $p = 1$  to  $P$  do
4:     if  $n > 0$  then
5:       Measure  $z_p(n)$ 
6:       Receive  $\mathcal{I}_q(n)$  for  $q \in [1 : P] \setminus \{p\}$ 
7:       Assemble  $i_p(n)$  using Algorithm 3
8:     else
9:       Receive  $\mathcal{I}_q(n)$  for  $q \in [1 : P] \setminus \{p\}$ 
10:      Set  $i_p(n)$  according to Equation (6)
11:     end if
12:     Solve OCP in Equation (16) and Apply  $u_p^*(0)$ 
13:     Broadcast  $\mathcal{I}_p(n)$ 
14:   end for
15: end for

```

---

As stated in Reference [10], the asymmetry of information has to be handled: Robot  $p$  has information  $\mathcal{I}_q(n)$  of the robots  $q, q \in [1 : p-1]$ , while for the successive robots  $q, q \in [p+1 : P]$ , the information of the last time instant  $\mathcal{I}_q(n-1)$  is accessible. The latter issue is solved in Algorithm 3. Here, in line 4, the information of the last time instant is shifted to the next time instant to construct the coupling constraints  $g_{q,1}^p, \dots, g_{q,N-1}^p$ . The last prediction step is duplicated by using Assumption 1 to obtain  $g_{q,N}^p$  in line 6, which is used to show the recursive feasibility of the system. Furthermore, the uniqueness of the solution of the OCP is ensured by selecting a strictly convex stage cost function  $\ell_p$ .

---

**Algorithm 3** Resolving asymmetry of communication data for robot  $p$

---

```

1: Given robot  $p$  and communication data  $\mathcal{I}_q(n)$  for  $q \in [1 : p - 1]$  and  $\mathcal{I}_q(n - 1)$  for  $q \in [p + 1 : P]$ 
2: for  $q = p + 1$  to  $P$  do
3:   for  $k = 1$  to  $N - 1$  do
4:     Set  $\mathcal{I}_q(n)(k) := \mathcal{I}_q(n - 1)(k + 1)$ 
5:   end for
6:   Set  $\mathcal{I}_q(n)(N) := \mathcal{I}_q(n - 1)(N)$ 
7: end for
8: Set  $i_p(n)$  according to Equation (6)

```

---

Each robot executes Algorithm 2 to perform the DMPC scheme with given initialisation conditions and targets until the target condition is matched.

#### 4. Prediction Horizon Length and Convergence

In this section, we derive sufficient conditions on the prediction horizon length to guarantee a decrease of the costs function and practical stability of the overall system. As the property of initial and recursive feasibility is needed to ensure that the algorithm does not terminate unexpectedly and a feasible solution is found for any time instant  $n \geq 0$ , we recap shortly the assumptions and theorems which were already presented in Reference [10]:

**Assumption 2** (Feasible initial conditions). *Given a set of robots  $[1 : P]$ , we have that  $z_p^0 \in Z$  and for all  $p, q \in [1 : P]$  with  $p \neq q$  condition  $g_{q,0}^p(z_p^0, q(z_q^0)) \geq 0$  holds.*

Assumption 2 states that the scenario is well defined regarding the state and coupling constraints. Based on this assumption, we can conclude the following:

**Lemma 1** (Initial feasibility). *Consider a set of robots  $[1 : P]$  with the model in Equation (1) and the constraints in Equations (2) and (3) satisfying Assumption 1. If Assumption 2 holds, then there exists a solution to the OCP in Equation (16).*

Concluding that there exists a solution to the initial optimal control problem (initial feasibility), recursive feasibility ensures that there exists a solution for any following time instant  $n > 0$ :

**Theorem 1** (Recursive feasibility). *Suppose a set of robots  $[1 : P]$  with the underlying model in Equation (1) and the constraints in Equations (2) and (3) satisfying Assumptions 1 and 2. If Algorithm 2 is applied, then the problem is recursively feasible, i.e., for all  $n \in \mathbb{N}_0$  and all  $p \in [1 : P]$ , there exists a solution to OCP in Equation (16).*

The detailed proofs are presented in Reference [10].

##### 4.1. Prediction horizon length with an occupancy grid

In Model Predictive Control (MPC), an appropriate prediction horizon length is crucial for convergence. Here, as the constraints utilising the maximum norm and the intermediate safety margin have to be obeyed (Proposition 2), the indifferentiability of the occupied cells has to be taken into account, which requires a suitable horizon length to enable the optimiser to recognise a decrease of costs by taking a detour. The following assumption is stated to ensure disjunct, feasible initial conditions, and targets of the robots:

**Assumption 3** (Disjunct initial conditions/targets). *Consider a set of robots  $[1 : P]$  with the underlying model in Equation (1), the state in Equation (2), and the control constraints in Equation (3). For each robot  $p \in [1 : P]$ , let  $z_p^0, z_p^* \in Z$ , and for all  $p, q \in [1 : P]$  with  $p \neq q$  and  $\hat{z}_j \in \{z_j^0, z_j^*\}$  with  $j \in \{p, q\}$ , we have*

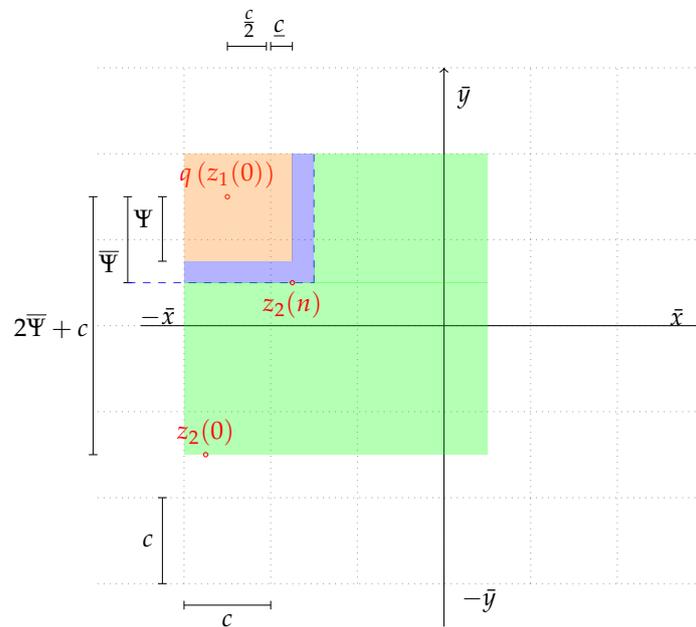
$$g_{q,0}^p(\hat{z}_p, q(\hat{z}_q)) \geq 2\bar{\Psi} + c, \quad (20)$$

and for  $\hat{z}_p = (x_p, y_p)^\top$

$$-\bar{x} + 2\bar{\Psi} + c \leq x_p \leq \bar{x} - (2\bar{\Psi} + c), \quad -\bar{y} + 2\bar{\Psi} + c \leq y_p \leq \bar{y} - (2\bar{\Psi} + c) \quad (21)$$

shall hold.

This assumption ensures the existence of a connected feasible path between all initialization points and to all targets, i.e., between any two of these points, there is one grid cell available for a third robot such that all constraints are satisfied. Additionally, to allow that other robots could pass at the bounds of  $Z$ , this is ensured by Equation (21). For an illustration, see Figure 2. Here,  $\Psi$  is included in the figure, so the overall distance for Equation (20) leads to  $2\bar{\Psi} + c$  to ensure that a robot can still move between the initial conditions or targets of the other robots.



**Figure 2.** Example of a given initial  $z_i(0)$ ,  $i \in \{1, 2\}$  and intermediate positions (here,  $z_2(n)$ ) with initial constraints (green) and constraints for intermediate states ( $n > 0$ , purple and orange) in a grid  $\mathcal{G}$  with  $a_{\max} = b_{\max} = 3$ .

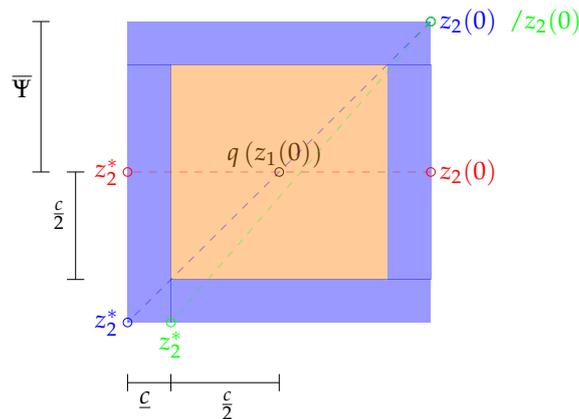
We like to point out that Assumption 3 is stricter as the target positions are included in the feasibility assumption and therefore directly induces Assumption 2 to hold. In order to obtain a sufficient prediction horizon length, which depends on the formulated cost function, a class of stage costs has to be defined:

**Definition 2** (Stage cost function). : Let  $\ell_p: Z \times U \rightarrow \mathbb{R}_{\geq 0}$  be a positive definite, continuous differentiable function with

$$\ell_p(z_p, u_p) := \left\| d_z(z_p, z_p^*) \right\|_2 + \lambda \left\| d_u(u_p, u_p^*) \right\|_2 \quad (22)$$

where  $d_z: Z \times Z \rightarrow \mathbb{R}_{\geq 0}$  and  $d_u: U \times U \rightarrow \mathbb{R}_{\geq 0}$  describe metrics with  $\ell_p(z_p^*, 0) = 0$ ,  $\ell_p(z_p, u_p) > 0$  for  $z_p \in Z$  and  $u_p \in U$  and where  $0 \leq \lambda \leq 1$ ,  $\lambda \in \mathbb{R}_{\geq 0}$  describes the fraction of the included penalty of the control.

For stage cost functions based on a normed distance, we have to evaluate the worst case of a necessary feasible detour caused by circumventing another robot’s cell in comparison to the direct path to ensure a sufficient prediction horizon length. Therefore, we consider the trajectory points (start and end point of the detour) of a robot at a cell to be circumvented. Then, as most pairs of start and end points of this detour are on the opposite sides of the cell, which has to be circumvented, these pairs can be classified as shown in Figure 3 by either taking the middle of a side length for start and (intermediate) targets or the diagonal length of an asymmetric pair.



**Figure 3.** Classification of start and end combinations of robot 2 to circumvent robot 1 to obtain the ratios of detour/direct way: middle of side length (red), diagonal (blue), and asymmetric (green).

We state the following theorem to obtain the worst-case positioning of the robots, i.e., we derive a maxima of the ratio of direct path to the necessary detour that a robot has to take:

**Theorem 2** (Worst-case positioning of robots). *Suppose a group of robots with the time discrete model in Equation (1) restricted by the state in Equation (3) and the control constraints in Equation (17) with feasible initial conditions and disjunct (intermediate) targets  $z_p(n), z_q(n), z_p(n + N), z_q(n + N) \in Z$  satisfying Assumption 3 with  $z_p(n + N) \neq z_q(n + N)$  and a minimum cell size  $\underline{c}$  obtained by Equation (9). For two robots  $p, q \in [1 : P]$ ,  $q$  keeps its position  $z_q(k) \equiv z_q$  with  $k \in [n : n + N]$ . Then, the worst-case of positioning of two robots is bounded by the maximum of the ratio of a necessary feasible detour ( $3(\frac{\underline{c}}{2} + \underline{c})$  for moving along the sides of the cell and  $\sqrt{(\frac{\underline{c}}{2} + \underline{c})^2 + \delta(\frac{\underline{c}}{2} + \underline{c})^2}$  for the hypotenuse to the target) to the direct path  $\|z_p(n) - z_p(n + N)\|_2$  (c.f.  $z_2$  (blue) in Figure 4b) by*

$$\frac{3(\frac{\underline{c}}{2} + \underline{c}) + \sqrt{(\frac{\underline{c}}{2} + \underline{c})^2 + \delta(\frac{\underline{c}}{2} + \underline{c})^2}}{2(\frac{\underline{c}}{2} + \underline{c}) + \delta(\frac{\underline{c}}{2} + \underline{c})} = \frac{3\bar{\Psi} + \sqrt{\bar{\Psi}^2 + \delta\bar{\Psi}^2}}{2\bar{\Psi} + \delta\bar{\Psi}} = \frac{3\bar{\Psi} + \bar{\Psi}\sqrt{(1 + \delta)^2}}{(2 + \delta)\bar{\Psi}}$$

with  $\delta \rightarrow 0$  leading to the maximum ratio

$$\lim_{\delta \rightarrow 0} \frac{3\bar{\Psi} + \bar{\Psi}\sqrt{(1 + \delta)^2}}{(2 + \delta)\bar{\Psi}} = \frac{4\bar{\Psi}}{2\bar{\Psi}} = 2.$$

This coincides with the shortest feasible target distance (opposite of both sides of the cell of  $z_p$ , c.f.  $z_2$  (red) in Figure 4b) with

$$\frac{4(\frac{\underline{c}}{2} + \underline{c})}{\|z_p(n) - z_p(n + N)\|_2} = \frac{4(\frac{\underline{c}}{2} + \underline{c})}{2(\frac{\underline{c}}{2} + \underline{c})} \equiv \frac{4\bar{\Psi}}{2\bar{\Psi}} = 2, \tag{23}$$

and therefore, the ratio of shortest feasible detour to direct path is bounded by

$$1 \leq \frac{3\bar{\Psi} + \bar{\Psi}\sqrt{(1+\delta)^2}}{(2+\delta)\bar{\Psi}} \leq \frac{4\bar{\Psi}}{2\bar{\Psi}} \leq 2.$$

**Proof.** Suppose, a larger ratio of detour to direct path exists. Then, we have to select a combination of start and end points on the side length of the cell. In total, four combinations can be classified, which are depicted in Figure 3. Setting the diagonal opposite edges (Figure 3, blue pair) would lead to a ratio of detour to direct way of

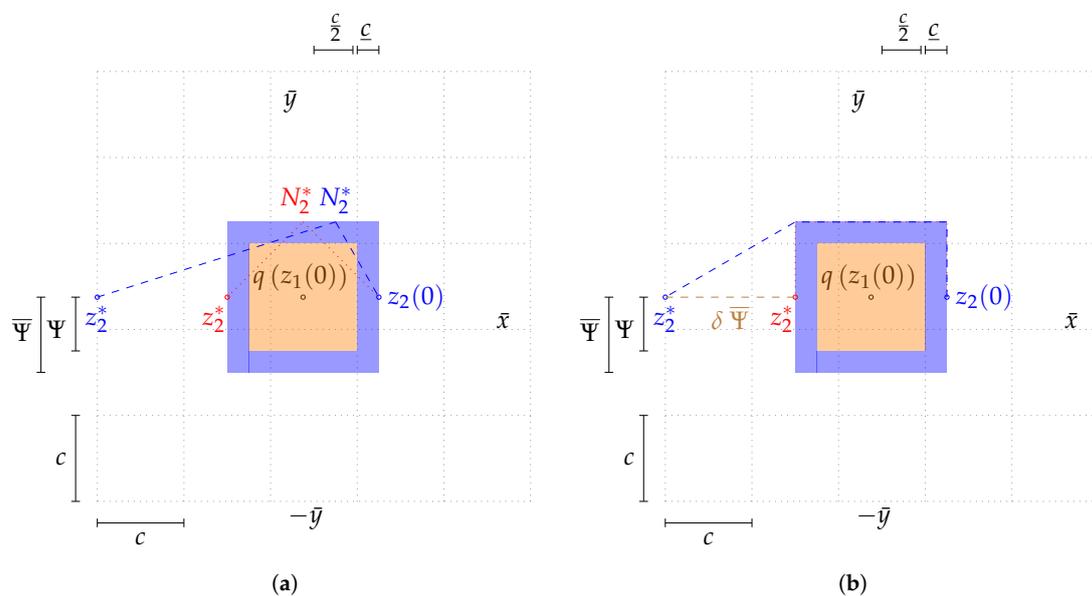
$$\frac{4\left(\frac{c}{2} + c\right)}{\|z_p(n) - z_p(n+N)\|_2} = \frac{4\left(\frac{c}{2} + c\right)}{\sqrt{4\left(\frac{c}{2} + c\right)^2 + 4\left(\frac{c}{2} + c\right)^2}} = \frac{4\bar{\Psi}}{\sqrt{8\bar{\Psi}}} = \sqrt{2},$$

and therefore, this is a contradiction to the hypothesis. If a nonsymmetric pair is chosen (Figure 3, green pair) with  $0 \leq \delta \leq 1$ , this lead to a ratio with

$$\frac{2\left(\frac{c}{2} + c\right) + 2\delta\left(\frac{c}{2} + c\right)}{\sqrt{4\left(\frac{c}{2} + c\right)^2 + 4\delta^2\left(\frac{c}{2} + c\right)^2}} = \frac{2\left(\bar{\Psi} + \delta\bar{\Psi}\right)}{2\sqrt{\bar{\Psi}^2 + \delta^2\bar{\Psi}^2}} \leq \sqrt{2}, \tag{24}$$

and therefore, all these combinations lead to a ratio of detour to direct tour of less than 2 and, therefore, a larger ratio than in Equation (23) does not exist, which completes the proof.  $\square$

Theorem 2 states the worst-case ratio, which has to be covered by a necessary detour in favour to the direct path, which also determines the ratio for the cost function to let the optimiser recognise a decrease of costs by circumventing a robot. This is illustrated in Figure 4a for two robots, where the quantised position of robot 1 is given as an occupied cell.



**Figure 4.** Illustration of the sufficient prediction horizon length  $N_2^*$  to recognise a decrease of costs by taking the detour (a) and feasible trajectories for robot 2 (b), with  $x_i(0)$  as the initial position,  $q(\cdot)$  as the quantised state, and  $x_i^*$  as the (intermediate) target for  $i \in [1 : 2]$ .

The sufficient horizon length  $N_p^*$ , necessary for the optimiser to recognise the decrease of costs, additionally depends on the position of the given target  $z_p^*$ , illustrated in Figure 4a for robot 2. Examples

for up to 4 robots ( $P \leq 4$ ) are presented in the Appendix. Based on this, we can derive a lower bound for the sufficient horizon length assuming the worst-case positioning from Theorem 2, where the ratio of necessary detour to shortest way is at maximum. An example of feasible trajectories is presented in Figure 4b, illustrating the trajectories depending on the target used for the maximum ratio by Theorem 2.

Note that, although the ratio is bounded, an additional difficulty to determine a lower bound on the prediction horizon length occurs as this depends also on the formulation of the cost function: The costs do not depend solely on the difference of current  $z_p(n)$  and target  $z_p^*$ . A positive definite cost function  $\ell_p$  defined by Definition 2 may include a penalty on the control with  $0 < \lambda \leq 1$  to avoid large control values or large differences in the control values as many cost functions aiming for a set point are defined for the euclidean space according to Reference [32] [Equation 3.3]. The open-loop costs for a robot retaining its position over the prediction horizon  $N$  are given by  $(N - 1) \ell_p(z_p(n; z_p^0), 0)$ . Then, if such a robot holds this position, the costs to circumvent the occupied cell have to be lower, i.e.,

$$(N - 1) \ell_p(z_p(n; z_p^0), 0) > \sum_{k=0}^{N-1} \ell_p(z_p(n), u_p(n)) \tag{25}$$

holds. Here, a prediction horizon length, which is long enough that a decrease of state costs ( $\lambda = 0$ ) is recognised by choosing a detour, is not necessarily long enough if control effort is also penalised ( $\lambda > 0$ ). Hence, a sufficient prediction horizon length has to incorporate the control effort, and therefore, the whole cost function has to be considered. Using Definition 2 and the assumption that a minimiser for the OCP in Equation (16) exists, the sufficient horizon length to recognise a decrease of the open-loop costs is stated in the following theorem:

**Theorem 3** (Sufficient prediction horizon length). *Suppose that two robots  $p, q$  are given with feasible initial conditions  $z_{p,q}^0 \equiv z_{p,q}(n)$  and disjunct (intermediate) targets  $z_p(K) \in Z$  with  $K \in \mathbb{N}_0, K < \infty$  satisfying Equations (2) and (3), minimising a cost function defined by Definition 2 with Assumptions 1 and 2 to hold and a feasible trajectory between  $z_p^0$  and  $z_p(K)$  exists bounded by Theorem 2. If*

$$N_p \geq N_p^* > 1 + \frac{\sum_{k=0}^{N_p^*-1} \ell_p(z_p(k; z_p^0), u_p^*(k))}{\ell_p(z_p(0; z_p^0), 0)} \tag{26}$$

*holds, then robot  $p$  is able to obtain a feasible control, which allows to decrease the costs according to Definition 2 and therefore is able to circumvent robot  $q$ .*

**Proof.** Based on Assumption 2 and Theorem 2, each robot  $p$  is equipped with a feasible (intermediate) initial condition and (intermediate) target, i.e., these are feasible and not occupied by other robots for the given time interval  $k \in [n : K]$ . By Theorem 2 a feasible trajectory exists between  $z_p(n)$  and  $z_p(n + N)$  for  $N \rightarrow \infty$  and bounded costs as the maximum ratio of necessary detour and direct path is bounded. As robot  $p$  is steered to  $z_p(N) = z_p(K)$  with  $K < \infty$  for a large horizon  $N$ , most of the open-loop cost values of the prediction converge to  $\ell_p(z_p(N), 0) \rightarrow 0$ . With the constrained control in Equation (3), the state  $z_p(K)$  is reachable by incorporating the impact of the control with

$$N_p^* \ell_p(z_p(0; z_p^0), 0) \leq 1 + \sum_{k=0}^{N_p^*-1} \ell_p(z_p(k; z_p^0), u_p^*(k)) \leq \max_{u \in U} \sum_{k=0}^{N_p^*-1} \ell_p(z_p(k; z_p^0), u),$$

while the upper bound is not necessarily optimal in terms of minimal control effort. Hence, as the states of  $z_p(n), z_p(K)$  satisfy Theorem 2 and a minimiser for the OCP in Equation (16) exists with a bounded maximal detour, a feasible trajectory exists for a finite horizon  $N$ . Therefore, the costs are bounded by the constrained control, which defines the required minimum cell size in Equation (9).  $\square$

The sufficient horizon length is illustrated in the following two examples depicting holonomic and non-holonomic robots, whereas we follow the definition that a robot is holonomic if the total degrees of freedom are equal to the controllable degrees of freedom. Here, we apply cost functions based on Reference [10], and for comparison reasons, the maximum norm constraint definition is used for both systems in Equation (13). The considered cellsize is  $c = 2.0$ , and the sampling size is  $T = 1.0$ .

**Example 1** (Holonomic example). *Let be two robots defined by*

$$z^+ = f(z, u) = z + u \text{ with } z_p(0) = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, z_p^* = \begin{pmatrix} -4 \\ 0 \end{pmatrix} \text{ and } z_q(0) = \begin{pmatrix} -2 \\ 0 \end{pmatrix}$$

satisfying the constraints in Equations (2) and (3) and the control bounded by  $u \in [-1, 1]^2$  with  $u = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}$ ,  $\|u\| \leq \sqrt{2}$  and stage costs

$$\ell_p(z_p, u_p) := \left\| \begin{pmatrix} (x_p - x_p^*)^2 \\ 5(y_p - y_p^*) \end{pmatrix} \right\|_2 + 0.2 \|u_p^2\|_2. \quad (27)$$

The costs of the open-loop prediction are presented in Table 1, illustrating the sufficient prediction horizon length to circumvent robot  $q$  with  $N = 7$ . The two columns at first show the costs for the open-loop prediction including the control impact with  $\lambda = 0.2$  and  $\lambda = 0$ , respectively. The last column represents the stage costs without imposing any control. For an insufficient horizon length, i.e.,  $N = 6$ , the solution of the optimiser reveals  $u_p = 0$ . However, to calculate the costs for taking the detour, the optimal control sequence is calculated with the sufficient horizon length  $N = 7$  and stripped by the last value.

**Table 1.** Open-loop costs with and without control impact.

$N$	$\sum_{k=0}^N \ell_p(z_p(k; z_p^0), u_p^*(k)), \lambda = 0.2$	$\sum_{k=0}^N \ell_p(z_p(k; z_p^0), u_p^*(k)), \lambda = 0$	$\sum_{k=0}^N \ell_p(z_p(k; z_p^0), 0)$
6	102.484	102.184	96.97
7	112.732	112.469	113.135

Therefore, if the control is included in the cost function, the prediction horizon has to be at least  $N \geq 7$  to recognise the decrease of costs in spite of the necessary detour. For an illustration, see Figure 5a.

**Example 2** (Non-holonomic example). *Based on Reference [33] an example of a non-holonomic robot defined in the 2D plane via*

$$z^+ = f(z, u) = \begin{pmatrix} x \\ y \\ \theta \end{pmatrix} + \begin{pmatrix} \cos(\theta) \\ \sin(\theta) \\ 0 \end{pmatrix} v + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \omega, \quad z_p(0) = \begin{pmatrix} 0 \\ 0 \\ \frac{\pi}{4} \end{pmatrix}, z_p^* = \begin{pmatrix} -4 \\ 0 \\ 0 \end{pmatrix} \text{ and } z_q(0) = \begin{pmatrix} -2 \\ 0 \\ 0 \end{pmatrix}$$

with  $\theta$  representing its orientation. Moreover, we suppose the given stage costs

$$\ell_p(z_p, u_p) := \left\| \begin{pmatrix} (x_p - x_p^*)^2 \\ 5(y_p - y_p^*) \\ (\theta_p - \theta_p^*)^2 \end{pmatrix} \right\|_2 + 0.2 \|u_p^2\|_2.$$

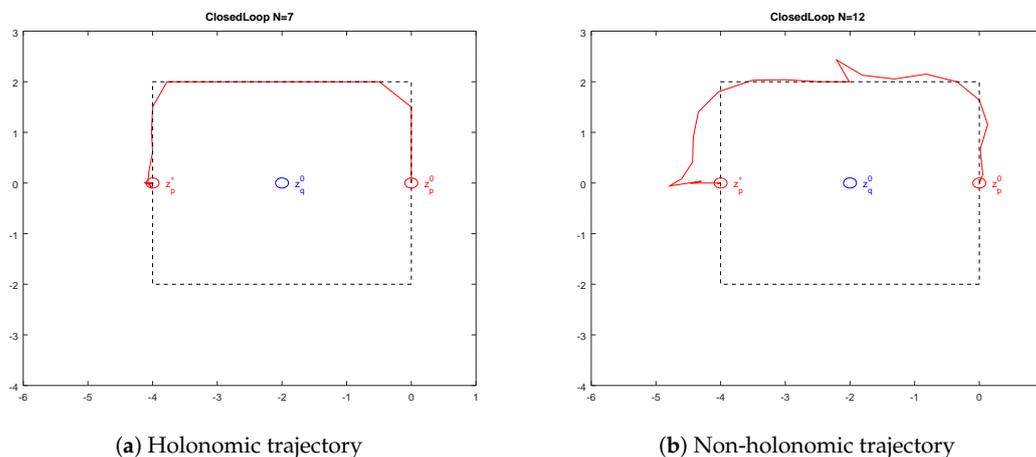
Due to the constraints of the kinematic model of the robot, the control might be higher than for the holonomic model if the robot has to turn around. The open-loop costs are given in Table 2, showing the costs with included control impact for  $\lambda = 0.2$  and  $\lambda = 0$ . To compute the control effort for taking the detour for the shorter

horizon, the optimal control sequence is computed with the longer horizon with the last value stripped as in the holonomic example.

**Table 2.** Open-loop costs with and without control impact.

$N$	$\sum_{k=0}^N \ell_p(z_p(k; z_p^0), u_p^*(k)), \lambda = 0.2$	$\sum_{k=0}^N \ell_p(z_p(k; z_p^0), u_p^*(k)), \lambda = 0$	$\sum_{k=0}^N \ell_p(z_p(k; z_p^0), 0)$
11	180.071	179.815	177.894
12	192.389	191.725	194.066

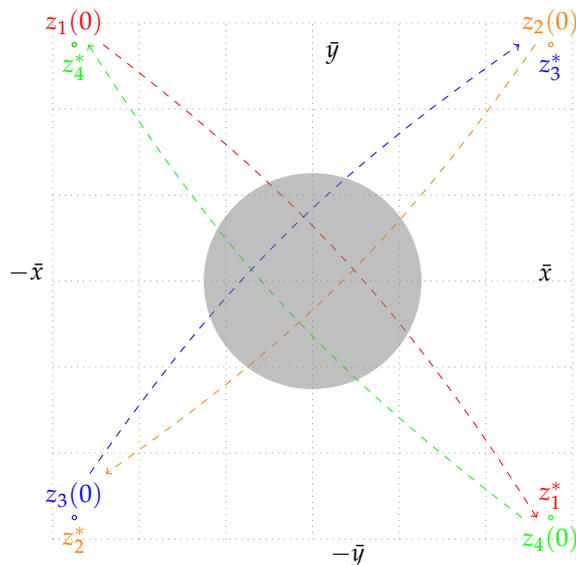
Here, the lowest sufficient horizon is  $N \geq 12$  to allow the optimiser to recognise the lower costs considering the detour with the additional control impact. For an illustration of the chosen trajectory, see Figure 5b. Therefore, both examples illustrate the necessary lower bound of the sufficient prediction horizon length to recognise a detour with the additional impact by penalising the control. The impact of the control values is too small to change the necessary horizon length but might have impact if a higher penalisation is chosen. A deep analysis about the design of the cost function and control effort analysis to stabilise such a non-holonomic robot is given in Reference [33].



**Figure 5.** Holonomic and non-holonomic trajectories of robot  $p$  with given initial conditions and targets  $z_p(0)$  and  $z_p^*$  and of robot  $q$  with state  $z_q^0$  to be circumvented with the cell constraint illustrated by the dashed rectangle.

#### 4.2. Convergence

Utilizing the geometry of our scenario, we show sufficient conditions for the convergence of robots to their given targets. Therefore, we utilise the assumption of feasible initial conditions and targets and the creation of a circle derived from the idea of a roundabout to let the robots use a circular path to ensure deadlock-free execution. An example for the conflicting zone of four robots with corresponding initial conditions and targets is presented in Figure 6.



**Figure 6.** Example for a collision avoidance conflict of four robots with given initial conditions  $z_i(0)$  and targets  $z_i^*$  for  $i \in \{1, 2, 3, 4\}$  with  $a_{\max} = b_{\max} = 3$  with conflicting area (gray).

**Assumption 4** (Feasible initial conditions and targets allowing boundary values). Consider a set of robots  $[1 : P]$  with the underlying model in Equation (1), the state in Equation (2), and the control constraints in Equation (3). For each robot  $p \in [1 : P]$  with  $\hat{z}_j \in \{z_j^0, z_j^*\}$ ,  $j \in \{p, q\}$ , we have that  $z_p^0, z_p^* \in Z$ , and for all  $p, q \in [1 : P]$  with  $p \neq q$ , we have that, for all  $\hat{z}_p$ , a non-empty set  $\mathbf{M}(\hat{z}_p) \subset Z$  exists such that

$$\mathbf{M}(\hat{z}_p) := \{(a, b) \mid (a, b) \in \mathcal{G}, \|\hat{z}_p - q((a, b))\|_2 \leq 2\bar{\Psi} + c\}$$

and

$$\hat{z}_p \notin \mathbf{M}(\hat{z}_q), \forall q \in [1 : P] \setminus \{p\}, p \neq q.$$

This assumption is weaker than the former one (Assumption 3) to allow for initial conditions and targets close to the boundary of the defined 2D plane that the robots are operating on. For the definition of the roundabout, the creation of the circle is presented in two versions: First, the size of the circle on the total number of robots and, second, a flexible radius is used depending on the number of conflicting robots only. Now, we show under which conditions the convergence of the robots to their targets can be guaranteed.

If the distance of the end of the predictions of two arbitrary chosen robots  $i, j \in [1 : P]$  satisfies

$$\|h(\mathcal{I}_i(n)(N)) - h(\mathcal{I}_j(n)(N))\|_2 \leq P\bar{\Psi}, \tag{28}$$

we define

$$(m_x, m_y) = \begin{pmatrix} 0.5(m_x^{\max} - m_x^{\min}) \\ 0.5(m_y^{\max} - m_y^{\min}) \end{pmatrix} \tag{29}$$

as centre of the circle in planar coordinates and radius  $r = \frac{P\bar{\Psi}}{2\pi}$ , where

$$\begin{aligned} (m_x^{max}, m_y^{max}) &= \max \{0.5 (h(\mathcal{I}_1(n)(N)) - h(\mathcal{I}_2(n)(N))), \dots, \\ &\quad 0.5 (h(\mathcal{I}_1(n)(N)) - h(\mathcal{I}_P(n)(N))), \dots, \\ &\quad 0.5 (h(\mathcal{I}_{P-1}(n)(N)) - h(\mathcal{I}_P(n)(N)))\} \\ (m_x^{min}, m_y^{min}) &= \min \{0.5 (h(\mathcal{I}_1(n)(N)) - h(\mathcal{I}_2(n)(N))), \dots, \\ &\quad 0.5 (h(\mathcal{I}_1(n)(N)) - h(\mathcal{I}_P(n)(N))), \dots, \\ &\quad 0.5 (h(\mathcal{I}_{P-1}(n)(N)) - h(\mathcal{I}_P(n)(N)))\} \end{aligned}$$

allows to state the equation of the circle. Hence,  $(x, y)^\top$  satisfying

$$(x - m_x)^2 + (y - m_y)^2 = \left(\frac{P\bar{\Psi}}{2\pi}\right)^2 =: r \tag{30}$$

represents a circle with  $x, y \in \mathbb{R}$ . To obtain the intersection between the trajectory and the circle, let  $(x_p(N), y_p(N))^\top = z_p(N)$  and  $(x_p(N-1), y_p(N-1))^\top = z_p(N-1)$ . This allows to state the linear equation with  $a^c = y_p(N) - y_p(N-1)$ ,  $b^c = x_p(N) - x_p(N-1)$ ,  $c^c = x_p(N)y_p(N-1) - x_p(N-1)y_p(N)$ , which can be inserted into Equation (30), and the intersection between circle and trajectory is obtained by

$$\begin{pmatrix} \bar{x}_{1,2} = m_x + \frac{a^c d^c \pm b^c \sqrt{r^2(a^c{}^2 + b^c{}^2)}}{a^c{}^2 + b^c{}^2} \\ \bar{y}_{1,2} = m_y + \frac{b^c d^c \mp a^c \sqrt{r^2(a^c{}^2 + b^c{}^2)}}{a^c{}^2 + b^c{}^2} \end{pmatrix}$$

To obtain a deadlock-free execution of the collision avoidance mechanism via the circle constraint, the following assumption is required:

**Assumption 5** (Target-point free circle). For a circle with centre  $(m_x, m_y)$  and radius  $r = \left(\frac{P\bar{\Psi}}{2\pi}\right)$ , suppose that, for all  $z_p^*$ ,  $p \in [1 : P]$ , the conditions  $\left| (m_x, m_y)^\top - z_p^* \right| > r$  and  $\left| (m_x, m_y)^\top - z_p^0 \right| > r$  hold.

This assumption ensures that neither an initial condition nor a target  $z_p^*$  lie inside the circle. Then, for each robot, the entry point is defined as the intercept point of the circle and the trajectory

$$z_p^s := \begin{pmatrix} x_p^s \\ y_p^s \end{pmatrix} = \begin{cases} (\bar{x}_1, \bar{y}_1)^\top, & \left\| z_p(N-1) - (\bar{x}_1, \bar{y}_1)^\top \right\|_2 < \left\| z_p(N-1) - (\bar{x}_2, \bar{y}_2)^\top \right\|_2 \\ (\bar{x}_2, \bar{y}_2)^\top, & \text{else} \end{cases} \tag{31}$$

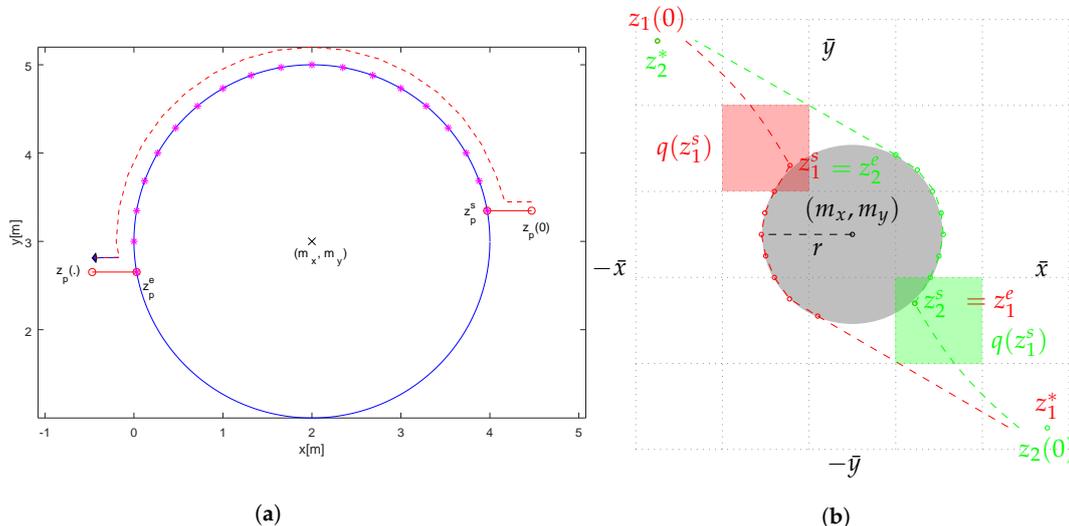
and the point where the robot leaves the circle is

$$z_p^e := \begin{pmatrix} x_p^e \\ y_p^e \end{pmatrix} = \begin{cases} (\bar{x}_2, \bar{y}_2)^\top, & \left\| z_p(N-1) - (\bar{x}_2, \bar{y}_2)^\top \right\|_2 < \left\| z_p(N-1) - (\bar{x}_1, \bar{y}_1)^\top \right\|_2 \\ (\bar{x}_1, \bar{y}_1)^\top, & \text{else} \end{cases} \tag{32}$$

As the radius of the circle is fixed with  $r = \frac{P\bar{\Psi}}{2\pi}$ , the feasibility of entry points on the circle for each participating robot is guaranteed by Proposition 2. Hence, to keep the robots on the circle, the control law is then changed to

$$u_p(n) = r \begin{pmatrix} \cos \left( \arccos \left( \frac{x_p(n) - m_x}{r} \right) + u_p^c \right) \\ \sin \left( \arccos \left( \frac{x_p(n) - m_x}{r} \right) + u_p^c \right) \end{pmatrix} \tag{33}$$

to force the robots on a circular trajectory with  $0 \leq u_p^c \leq \underline{c}$  describing the advance of the robot on the circle. As the radius depends on the number of robots, here,  $P$ , all robots may be driven on a common circle if needed. An illustration of the applied control law is given in Figure 7a.



**Figure 7.** Illustration of the circular control with centre  $(m_x, m_y)$ , and entry and exit points  $z_p^s, z_p^e$  for one robot (a) and two robots with start and end points  $z_1^s, z_2^s$  and  $z_1^e, z_2^e$  leaving earlier due to the tangential condition (b).

Each robot evaluates the distance to other robots to switch the control law to a circular curve to establish a shared roundabout and to, therefore, ensure the collision avoidance restrictions via Algorithm 4.

If the condition in Equation (28) holds, the algorithm evaluates for each time instant that the predictions of at least two robots undermine their distance. Then, for those robots, the control law is switched and the circular trajectory is carried out for all robots in  $\mathbf{Q}$ , i.e., all robots, which were added to the set  $\mathbf{Q}$ . This set  $\mathbf{Q}$  represents the robots, which satisfy Equation (28) with  $2 \leq |\mathbf{Q}| \leq P$ . The other robots are set to an immediate hold (Algorithm 4, line 12) and, hence, do not interfere with the created circle. The variable  $w_p$  ensures that the evaluation of the centre is carried out once and that the control law is switched back if it was circular before. If the exit  $z_p^e$  of the circle is reached or the target  $z_p^*$  is tangential reachable (see Algorithm 4, line 18), the remaining trajectory on the circle to  $z_p^e$  is skipped and the robot continues under the regime of the original control law, which is set back in Algorithm 4, line 19; see Figure 7b for an illustration. To integrate Algorithm 4 into the DMPC scheme, in Algorithm 2, we include after line 11

**Call Algorithm 4.**

Note that, from the circular control law established by Algorithm 4, the robots are still obeying the collision avoidance constraints between the other robots as the DMPC scheme in Algorithm 2 is still executed and Assumption 1 is ensured for  $u_p^c$  with  $\cos\left(\arccos\left(\frac{z_p^{(n)} - m_x}{r}\right) + u_p^c\right) = 0$  and  $\sin\left(\arccos\left(\frac{z_p^{(n)} - m_x}{r}\right) + u_p^c\right) = 0$ .

---

**Algorithm 4** Evaluation for switching control law of robot  $p$  to establish a collision avoidance circle

---

```

1: for all  $p \in [1 : P]$  do
2:   if  $\exists q \in [1 : P] \setminus \{p\}$  where Equation (28) holds for  $p, q$  then
3:     Add  $\mathbf{Q} := \mathbf{Q} \cup \{p, q\}$ 
4:   end if
5: end for
6: if  $\mathbf{Q} \neq \emptyset$  or  $w_p = true$  then
7:   if  $w_p = false$  then
8:     Set centre  $(m_x, m_y)$  for  $z_p$  with Equation (29) and  $z_p^s, z_p^e \forall p \in \mathbf{Q}$  with Equations (31) and
      (32)
9:   end if
10:  Set  $w_p := true$ 
11:  for all  $p \in [1 : P]$  and  $p \notin \mathbf{Q}$  do
12:    Keep  $u_p = \bar{u}_p \quad \forall k \in [n : n + N]$  according to Assumption 1
13:  end for
14:  if  $z_p^0 = z_p^s$  and  $p \in \mathbf{Q}$  then
15:    Set  $u_p$  according to Equation (33)
16:  end if
17: end if
18: if  $w_p = true$  and  $\left( (z_p^0 - z_p^*) \times (z_p^0 - (m_x, m_y)^\top) = 0 \right.$  or  $z_p^0 = z_p^e$  for  $p \in [1 : P]$ ) then
19:   Set  $u_p$  back to Equation (3)
20:   Remove  $\mathbf{Q} := \mathbf{Q} \setminus \{p\}$ 
21:   Set  $w_p := false$ 
22: end if
23: if  $\mathbf{Q} = \emptyset$  then
24:   for all  $p \in [1 : P]$  do
25:     Reset  $u_p$  according to Equation (3)
26:     Set  $w_p := false$ 
27:   end for
28: end if

```

---

**Theorem 4** (Collision avoidance via a fixed circular curve). *Suppose a given number of  $P$  robots, each defined by Equation (1), with restrictions on state and control in Equations (2) and (3) and with initial feasible conditions and targets fulfilling Assumption 4. If the distance between two robots is such that Equation (28) is fulfilled and Assumption 5 is satisfied for all robots  $p \in [1 : P]$ , then the calculation of a circle and execution of the DMPC scheme in Algorithm 2 using the switched control by Algorithm 4 will allow the robots to let them converge to their targets, i.e., to steer the system practically stable.*

**Proof.** From Assumption 4, the feasibility of the initial conditions ( $n = 0$ ) and targets for all  $P$  robots is guaranteed, which is weaker than Assumption 3 with additional allowance of points near to the bounds of the 2D plane. Then, with  $n > 0$  each robot  $p \in [1 : P]$  minimises the distance to the individual target  $z_p^*$ . Then, if the condition in Equation (28) is fulfilled for at least two robots, the centre  $(m_x, m_y)^\top$  of the circle for the conflicting robots is calculated with Equation (29) for the given radius  $r$  and the circular control law is applied for all conflicting robots according to Equation (33). All other robots are set to an immediate hold via  $\bar{u}_p$ . As the condition in Equation (28) holds for any arbitrary chosen robots, the circle is created, when at least two predicted states are closer according to the given condition, and therefore, any other robot is outside this circle fulfilling Assumption 5 and additionally

stops moving. With this assumption that no target is located inside the circle  $(z_p^* - (m_x, m_y)^\top)$ , each robot  $p \in \mathbf{Q}$  is able to calculate a feasible route via Algorithm 4 subject to the intersections  $z_p^s$  and  $z_p^e$ . Furthermore, the circumference of the circle is chosen such that all robots  $p \in [1 : P]$  could be integrated on the circular curve and retains the intermediate safety margin, i.e., the trajectories on the circle are feasible. As the exit point is on the opposite position of the circle for each robot, the cost function also decreases with closer distance. Close to the exit point, Algorithm 4 allows an earlier switch to the old control law in Equation (3) if a tangential movement to the original target  $z_p^*$  is possible under the condition that orthogonality of the tangent of the current position at the circle and the vector between current position and centre is ensured. Thus, the trajectory still ensures feasibility and allows each robot to converge closer to its target, which renders the system practically stable.  $\square$

In a second version, we keep the size of the circle flexible: Instead of using a fixed radius length from Assumption 5, the circle is established for at least two robots with

$$r = (x - m_x)^2 + (y - m_y)^2 = \left( \frac{|\mathbf{Q}| \bar{\Psi}}{2\pi} \right) \quad (34)$$

where  $\mathbf{Q}$  is the set of robots involved in the circle via Equation (28); therefore  $\mathbf{Q}$  reveals  $2 \leq |\mathbf{Q}| \leq P$ . Then, the circle is increased in Algorithm 5 as long as no other robot or target is inside the circle with  $\|z_q(k) - (m_x, m_y)^\top\|_2 > r$  for  $k \in [n : n + N]$  and  $\|z_q^* - (m_x, m_y)^\top\|_2 > r$ , respectively.

Here, the circle is increased in Algorithm 11, line 11 as long as no current position  $z_p^0$  or target  $z_p^*$  is inside the circle. Again, the variable  $w$  ensures that only once the circle is calculated and not changed further. A robot  $p$  not participating in this circle switches the control to  $u_p = 0$  according to the stop assumption as long as the circle exists. Then, after the robots following the circle switches their control law back according to the leaving condition in Algorithm 5, line 22, i.e.  $\mathbf{Q} = \emptyset$ , the remaining robots are unblocked, i.e., allowing a control  $u_p \neq 0$ . According to the previous fixed circle algorithm, the DMPC algorithm is modified by including in Algorithm 2

**Call Algorithm5**

after line 11.

---

**Algorithm 5** Flexible increase of the circular curve

---

```

1: for all  $p \in [1 : P]$  do
2:   if  $\exists q \in [1 : P] \setminus \{p\}$  where Equation (28) holds for  $p, q$  then
3:     Add  $Q := Q \cup \{p, q\}$ 
4:   end if
5: end for
6: if  $Q \neq \emptyset$  or  $w_p = true$  then
7:   Set  $k := |Q|$ 
8:   if  $w_p = false$  then
9:     Set centre  $(m_x, m_y)$  for  $z_p \forall p \in Q$  with Equation (29)
10:    Set  $w_p := true$ 
11:    while  $\forall p : \left\| z_p^* - (m_x, m_y)^\top \right\|_2 > \left( \frac{(k+1)\bar{\Psi}}{2\pi} \right)$  and  $\left\| z_p^0 - (m_x, m_y)^\top \right\|_2 > \left( \frac{(k+1)\bar{\Psi}}{2\pi} \right)$  and  $k \leq P$ 
12:      do
13:        Set  $r := \left( \frac{(k+1)\bar{\Psi}}{2\pi} \right)$ 
14:        Set  $k := k + 1$ 
15:      end while
16:      for all  $p \in [1 : P]$  and  $p \notin Q$  do
17:        Keep  $u_p = \bar{u}_p \quad \forall k \in [n : n + N]$  according to Assumption 1
18:      end for
19:      end if
20:      if  $z_p^0 = z_p^s, \quad p \in Q$  then
21:        Set  $u_p$  after Equation (33)
22:      end if
23:      if  $w_p = true$  and  $(z_p^0 - z_p^*) \times (z_p^0 - (m_x, m_y)^\top) = 0$  OR  $z_p^0 = z_p^e \quad \forall p \in Q$  then
24:        Set  $u_p$  according to Equation (3)
25:        Set  $w_p := false$ 
26:        Set  $Q = Q \setminus \{p\}$ 
27:      end if
28:      if  $Q = \emptyset$  then
29:        for all  $p \in [1 : P]$  do
30:          Reset  $u_p$  according to Equation (3)
31:        end for
32:      end if

```

---

**Theorem 5** (Collision avoidance via flexible circular curve). *For a given number of  $P$  robots with initial conditions fulfilling Assumption 4, if Equation (28) is fulfilled for at least two robots, the calculation of a circle based on Assumption 5 by using Equation (34) for Algorithm 5 lets the robots converge to their individual targets, which ensures practical stability.*

**Proof.** Consider again the Assumption 4 with feasible initial and target conditions, which ensures the intermediate safety margins and allows boundary values. Each robot calculates a feasible trajectory minimising the objective with subject to their target within a finite horizon. Then, if Equation (28) is fulfilled, the circle is established for at least two robots  $p, q \in Q, p, q \in [1 : P]$ , of which predictions are

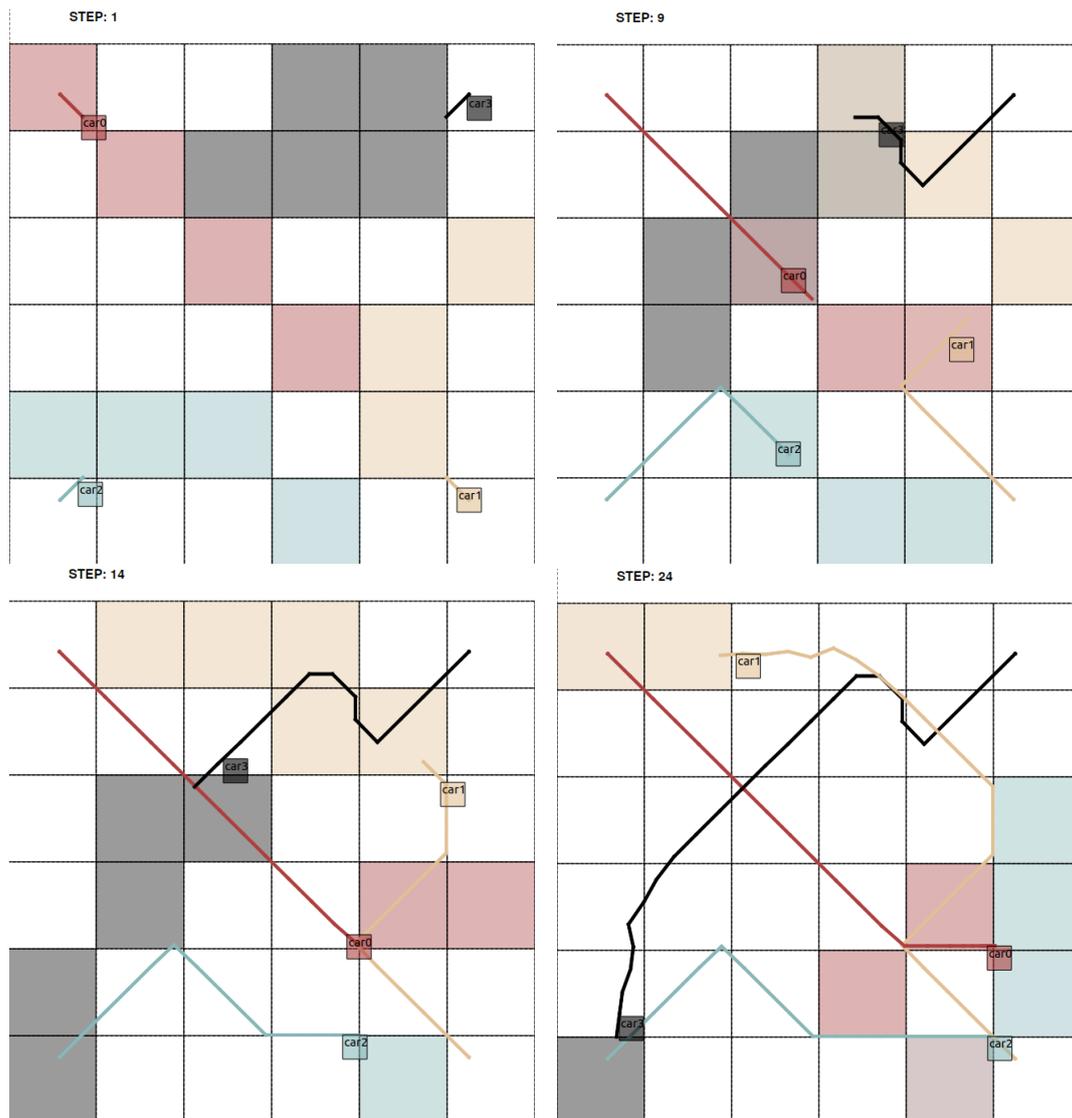
closest among all other robots. As this distance allows to establish a circle at least for these two robots, the increasing radius via Algorithm 5 allows a maximum size of the circle without incorporating the states of the other robots. Then, with switching the control law subject to Equation (33) and the subset of conflicting robots, the deadlock-free execution of this subset  $\mathbf{Q}$  is ensured. Then, with the robots following the circular curve under the circular control law, each robot  $i \in \mathbf{Q}$  is able to reach the opposite position and is able to leave on the same condition as in Theorem 4 when the target is tangential reachable. Moreover, as the exit point is closer to the target, the costs also decrease. As long as  $\mathbf{Q}$  is not empty, the other robots are blocked to prevent them from crossing the circle. Note that, after the circle is dissolved, a minimum distance to other robots which were not participating in the circle is guaranteed by Equation (28) as, for such a robot  $q \notin \mathbf{Q}$ , the distance is at least  $\|(m_x, m_y)^\top - z_q\|_2 > r$  with  $z_q \equiv z_q(k)$  for  $k \in [n : n + N]$ . Moreover, as the control is set to  $u_q = \bar{u}_p$  as long as  $\mathbf{Q} \neq \emptyset$ , after setting back to the original control law (Algorithm 5, line 29), a feasible solution exists due to Assumption 1. The same procedure to create the circle can be repeated to resolve following conflicts while the robots are approaching to their targets, hence being able to converge to their individual targets, i.e., the system achieves practical stability.  $\square$

## 5. Numerical Illustration

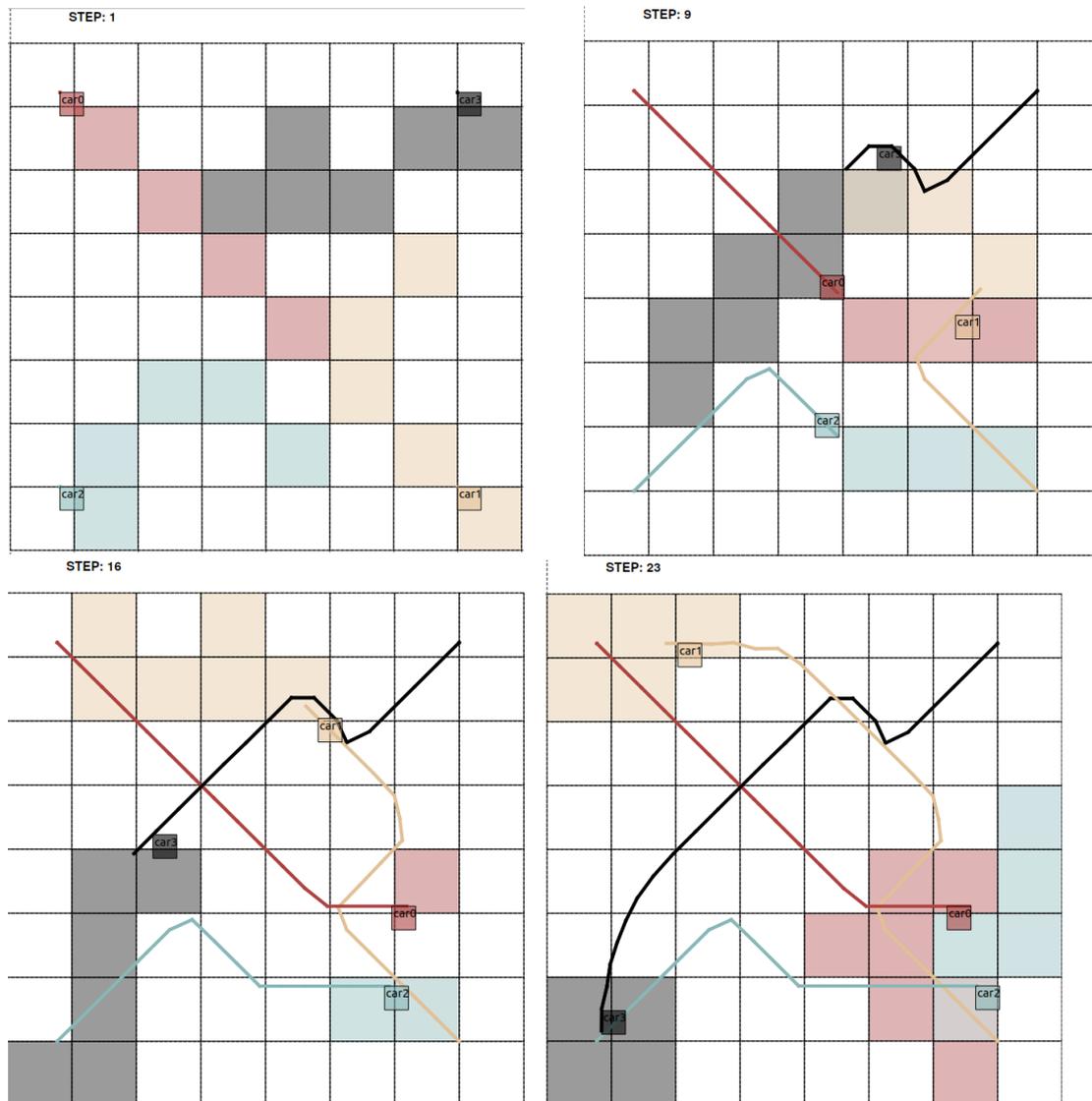
Detailed numerical results about the occupancy grid setting were presented in References [10,34] using, for the first, a fixed order of optimising and executing and, for the latter, a priority rule setting. Without enforcing explicitly the rule to reveal a circular curve, we like to point out one scenario with  $P = 4$  non-holonomic robots from Reference [34], where a minimum open-loop costs priority rule is used, i.e., the robot with minimum costs over the prediction horizon goes first. We selected here two scenarios using cell sizes  $c = 1.5$  and  $c = 2.0$  and with start and target positions of the robots opposite to each other in the edges of the operation space. Hence, they have to circumvent each other in the centre. In Figures 8 and 9, the snapshots of the different time instants are illustrated to show the evolution of time of the trajectories and the behaviour of the robots.

In the first scenario depicted in Figure 8 with cell size  $c = 2.0$ , the large cell size demands a large radius due to the collision avoidance constraints. Robot 0 (“car0”) goes at first and, hence, may start with a straight trajectory due to the fixed order and the initialisation phase in Algorithm 1. As all robots start with their initial prediction that they keep their position in the first time instant over the prediction horizon, i.e., ensuring feasibility, robot 0 has the centre of the operation space available to use this at first. The other robots (“car1-3”) have to incorporate this trajectory from robot 0 and have to choose detours shown by the coloured predictions, which forms a roundabout in a clockwise manner. The occupied cells, which are reserved by the predictions of the robots, are coloured accordingly to the robots. In later time instants, the trajectories show clearly that, although the order may disadvantage the last robots most, the arrival times are quite close due to the identical behaviour of the sidestepping robots.

With cell size  $c = 1.5$  shown in Figure 9, the robots form accordingly a smaller roundabout due to smaller cells and less necessary detours. Similar to the first scenario, robot 0 has the advantage of being enabled to take the centre of the operation space due to the fixed optimisation order. The trajectory especially of robot 3 “car3” shows that the detours are shorter as the robot crosses the centre in a more straight line than in the scenario before. Therefore, without setting implicit conditions on the behaviour of the collision avoidance in both scenarios, the optimiser leads to the pattern of a circular curve, which is also used in street traffic as roundabouts.



**Figure 8.** Holonomic example with 4 robots, prediction horizon length  $N = 12$ , and cell size  $c = 2.0$ : snapshots at time instants  $n = 1$  (left top),  $n = 9$  (right top),  $n = 14$  (left bottom), and  $n = 24$  (right bottom): Trajectories are drawn in continuous lines, and the predicted occupied cells are coloured in the same colour as the robot.



**Figure 9.** Holonomic example with 4 robots, horizon length  $N = 12$ , and cell size  $c = 1.5$ : snapshots at time instants  $n = 1$  (left top),  $n = 9$  (right top),  $n = 16$  (left bottom), and  $n = 23$  (right bottom): Trajectories are drawn in continuous lines, and the predicted occupied cells are coloured in the same colour as the robot.

## 6. Conclusions

In this paper, we derived sufficient conditions for a sufficient prediction horizon length and for convergence to achieve practical stability for a system of distributed robots with quantised communication based on an occupancy grid. We used the properties of the occupancy grid to determine the sufficient prediction horizon length, which is crucial for the optimiser to detect the decrease of costs by taking a detour. The occupancy grid allows to determine the maximum ratio of feasible detour and direct path, which allows to derive such a sufficient prediction horizon length. Moreover, we utilised the idea from a roundabout to establish a control law, which describes a pattern obtained by the numeric results to show convergence for the overall system. This control law was implemented in two versions: First, a fixed radius was used to allow for all robots participating immediately in the circle. Second, this control law was extended to use a flexible radius size to reduce the necessary space to establish the roundabout.

Future considerations include the incorporation of a dynamic optimisation order (priority rules) of the robots and restrictions to traffic scenarios where the freedom of the agents (then cars) is more restricted, e.g., they have to keep within lanes. This may allow easier derivation of conditions to keep such a system deadlock-free, i.e., to steer all distributed agents to their equilibrium.

**Author Contributions:** Conceptualization, T.S. and J.P.; Methodology, T.S.; Software, T.S.; Validation, T.S.; Formal Analysis, T.S. and J.P.; Writing—Original Draft Preparation, T.S.; Writing—Review & Editing, T.S. and J.P.; Visualization, T.S.; Supervision, J.P. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding. The APC was funded by the Staats- und Universitätsbibliothek Bremen (SuUB), Germany.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Monostori, L.; Valckenaers, P.; Dolgui, A.; Panetto, H.; Brdys, M.; Csáji, B.C. Cooperative control in production and logistics. *Ann. Rev. Control* **2015**, *39*, 12–29. [[CrossRef](#)]
2. Mehrez, M.W.; Mann, G.K.I.; Gosine, R.G. An Optimization Based Approach for Relative Localization and Relative Tracking Control in Multi-Robot Systems. *J. Intell. Robot. Syst.* **2017**, *85*, 385–408. [[CrossRef](#)]
3. Espinoza, L.J.; Sánchez, L.A.; Osorio, M. Exploring unknown environments with mobile robots using SRT-Radial. In Proceedings of the IEEE International Conference on Intelligent Robots and Systems, San Diego, CA, USA, 29 October–2 November 2007; pp. 2089–2094.
4. Saffarian, M.; Fahimi, F. Non-Iterative nonlinear model predictive approach applied to the control of helicopters group formation. *Robot. Autono. Syst.* **2009**, *57*, 749–757. [[CrossRef](#)]
5. Venkat, A.; Rawlings, J.; Wright, S. Stability and optimality of distributed model predictive control. In Proceedings of the 44th IEEE Conference on Decision and Control, Seville, Spain, 12–15 December 2005; pp. 6680–6685.
6. Worthmann, K.; Kellett, C.M.; Braun, P.; Grüne, L.; Weller, S.R. Distributed and Decentralized Control of Residential Energy Systems Incorporating Battery Storage. *IEEE Trans. Smart Grid* **2015**, *6*, 1914–1923. [[CrossRef](#)]
7. Christofides, P.D.; Scattolini, R.; Muñoz de la Peña, D.; Liu, J. Distributed model predictive control: A tutorial review and future research directions. *Comput. Chem. Eng.* **2013**, *51*, 21–41. [[CrossRef](#)]
8. Raffard, R.L.; Tomlin, C.J.; Boyd, S.P. Distributed optimization for cooperative agents: application to formation flight. In Proceedings of the 43rd IEEE Conference on Decision and Control (CDC), Nassau, Bahamas, 14–17 December 2004; Volume 3, pp. 2453–2459.
9. Mehrez, M.W.; Sprodowski, T.; Worthmann, K.; Mann, G.K.I.; Gosine, R.G.; Sagawa, J.K.; Pannek, J. Occupancy Grid based Distributed Model Predictive Control of Mobile Robots. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada, 24–28 September 2017; pp. 4842–4847.
10. Sprodowski, T.; Mehrez, M.W.; Worthmann, K.; Mann, G.K.I.; Gosine, R.G.; Sagawa, J.K.; Pannek, J. Differential Communication with Distributed Model Predictive Control of Mobile Robots based on an Occupancy Grid. *Inf. Sci.* **2018**, *453*, 426–441. [[CrossRef](#)]
11. Richards, A.; How, J. Decentralized model predictive control of cooperating UAVs. In Proceedings of the 43th IEEE Conference on Decision and Control, Nassau, Bahamas, 14–17 December 2004; Volume 4, pp. 4286–4291.
12. Müller, M.A.; Allgöwer, F. Economic and Distributed Model Predictive Control: Recent Developments in Optimization-Based Control. *SICE J. Control Meas. Syst. Integr.* **2017**, *10*, 39–52.
13. Jia, D.; Krogh, B. Min-max feedback model predictive control for distributed control with communication. *Proc. Am. Control Conf.* **2002**, *6*, 4507–4512.
14. Müller, M.A.; Reble, M.; Allgöwer, F. Cooperative control of dynamically decoupled systems via distributed model predictive control. *Int. J. Robust Nonlinear Control* **2012**, *22*, 1376–1397. [[CrossRef](#)]
15. Wang, C.; Ong, C.J. Distributed model predictive control of dynamically decoupled systems with coupled cost. *Automatica* **2010**, *46*, 2053–2058. [[CrossRef](#)]

16. Seyboth, G.S.; Dimarogonas, D.V.; Johansson, K.H.; Frasca, P.; Allgöwer, F. On robust synchronization of heterogeneous linear multi-agent systems with static couplings. *Automatica* **2015**, *53*, 392–399. [[CrossRef](#)]
17. Giselsson, P. A generalized distributed accelerated gradient method for distributed model predictive control with iteration complexity bounds. In Proceedings of the 2013 American Control Conference (ACC), Washington, DC, USA, 17–19 June 2013.
18. Giselsson, P.; Rantzer, A. On feasibility, stability and performance in distributed model predictive control. *IEEE Trans. Autom. Control* **2014**, *59*, 1031–1036. [[CrossRef](#)]
19. de Oliveira, L.B.; Camponogara, E. Multi-agent model predictive control of signaling split in urban traffic networks. *Transp. Res. Part C Emerg. Technol.* **2010**, *18*, 120–139. [[CrossRef](#)]
20. Farina, M.; Scattolini, R. An output feedback distributed predictive control algorithm. In Proceedings of the IEEE Conference on Decision and Control and European Control Conference, Orlando, FL, USA, 12–15 December 2011; pp. 8139–8144.
21. Grüne, L.; Worthmann, K. A distributed NMPC scheme without stabilizing terminal constraints. In *Distributed Decision Making and Control*; Springer: London, UK, 2012; pp. 261–287.
22. Pannek, J. Parallelizing a state exchange strategy for noncooperative distributed NMPC. *Syst. Control Lett.* **2013**, *62*, 29–36. [[CrossRef](#)]
23. Huang, H.; Yu, C.; Gusrialdi, A.; Hirche, S. Topology Design for Distributed Formation Control towards Optimal Convergence Rate. In Proceedings of the American Control Conference, Montréal, QC, Canada, 27–29 June 2012; pp. 3895–3900.
24. Pu, Y.; Zeilinger, M.N.; Jones, C.N. Quantization design for distributed optimization with time-varying parameters. In Proceedings of the 54th IEEE Conference on Decision and Control (CDC), Osaka, Japan, 15–18 December 2015; pp. 2037–2042.
25. Mhaskar, P.; El-Farra, N.H.; Christofides, P.D. Stabilization of nonlinear systems with state and control constraints using Lyapunov-based predictive control. *Syst. Control Lett.* **2006**, *55*, 650–659. [[CrossRef](#)]
26. Liu, J.; Chen, X.; Mun, D. Sequential and Iterative Architectures for Distributed Model Predictive Control of Nonlinear Process Systems *AIChE J.* **2010**, *56*, 2137–2149.
27. Hermans, R.M.; Lazar, M.; Jokić, A. Almost Decentralized Lyapunov-based Nonlinear Model Predictive Control. In Proceedings of the American Control Conference, Baltimore, MA, USA, 30 June–2 July 2010; pp. 3932–3938.
28. Liu, C.; Gao, J.; Xu, D. Lyapunov-based Model Predictive Control for Tracking of Nonholonomic Mobile Robots under Input Constraints. *Int. J. Control Autom. Syst.* **2017**, *15*, 2313–2319. [[CrossRef](#)]
29. Körkel, S.; Qu, H.; Rucker, G.; Sager, S. Derivative Based vs. Derivative Free Optimization Methods for Nonlinear Optimum Experimental Design. In *Current Trends in High Performance Computing and Its Applications, Proceedings of the International Conference on High Performance Computing and Applications, Sorrento, Italy, 21–23 September 2005*; Springer: Heidelberg, Germany, 2005; pp. 339–344.
30. Richards, A.; How, J. A decentralized algorithm for robust constrained model predictive control. In Proceedings of the American Control Conference (ACC), Boston, MA, USA, 30 June–2 July 2004; pp. 4261–4266.
31. Richards, A.; How, J.P. Robust distributed model predictive control. *Int. J. Control* **2007**, *80*, 1517–1531. [[CrossRef](#)]
32. Grüne, L.; Pannek, J. *Nonlinear Model Predictive Control: Theory and Algorithms*; Communications and Control Engineering; Springer: London, UK, 2017.
33. Worthmann, K.; Mehrez, M.W.; Zanon, M.; Mann, G.K.I.; Gosine, R.G.; Diehl, M. Model Predictive Control of Nonholonomic Mobile Robots Without Stabilizing Constraints and Costs. *IEEE Trans. Control Syst. Technol.* **2016**, *24*, 1394–1406. [[CrossRef](#)]
34. Sprodowski, T.; Mehrez, M.W.; Pannek, J. Priority-based DMPC with an Occupancy Grid for Mobile Systems. *Int. J. Control* **2020**, 1–29, doi:10.1080/00207179.2019.1707291. [[CrossRef](#)]



This article is printed under terms of Creative Commons CC-BY-NC-ND <https://creativecommons.org/licenses/by-nc-nd/4.0/> with all rights ©MDPI.  
Original publication is available via <https://doi.org/10.3390/app10031007>.

# Frequency Based Model Predictive Control of a Manufacturing System

Tobias Sprodowski \* Juliana Keiko Sagawa \*\* Jürgen Pannek \*

\* *University of Bremen, Department of Production Engineering,  
Hochschulring 20, 28359 Bremen, Germany (e-mail:  
[spr,pan]@biba.uni-bremen.de)*

\*\* *Department of Production Engineering, Federal University of São  
Carlos, Brazil (e-mail: juliana@dep.ufscar.br)*

---

**Abstract:** Manufacturing scenarios have to incorporate many dynamic influences such as urgent orders or unavailability of resources. To better address this, tools of control theory and dynamic modelling should be employed. In this paper, we present a novel multi-product dynamic model inspired by an existing bond graph model for manufacturing systems, and apply a Model Predictive Controller (MPC) to steer the machines to a reference state to fulfil a given product demand. In previous related works, the capacity adjustment of the machines is based on local information only and the existing model can only represent a fixed manufacturing configuration. Thus, the contribution of this approach is twofold: it allows applying optimal control methods, generating a centralized solution based on global information; it depicts the machines as autonomous elements, enabling to represent more flexible configurations of production systems. We show in simulations that an appropriate cost function and a suitable prediction horizon length are important factors for the control of the system.

*Keywords:* Model Predictive Control, Bond graphs, Operational Research, Logistics and Planning

---

## 1. INTRODUCTION

In today's manufacturing systems, many product varieties and disturbing influences, as availability of material or tools, have to be taken into account. This leads to a complex problem where a solution has to fulfil a product mix, achieve the requested production rates, and allow the reaction to disturbances like breakdowns of machines. These requirements demand flexible systems. Various control techniques were applied to control such systems like scheduling policies, ordering policies, as in Qiu et al. (2017), or (meta-)heuristics, utilised by Abedinnia et al. (2017).

Some approaches were derived from the application of classical control theory and optimal control to manufacturing systems. While in first models only single workstation problems were considered, cf. Ratering and Duffie (2003), these models were quickly extended to multiple products and workstations (see, e.g., Kim and Duffie (2006)). Considering an optimal control policy, Ma and Koren (2004) developed a step control approach, which divides an stochastic optimal control problem into linear deterministic subproblems. Several kinds of offline-controllers were defined to include multiple states (processing, breakdowns) of workstations in a flow shop.

Model Predictive Control (MPC), which is adopted in the production control area as *Rolling Planning Horizon* (see Lu et al. (2015)), has also been applied to manufacturing systems. According to it, the system is modelled as an optimal control problem that is solved at each time instant over a given prediction horizon, possibly featuring nonlinear dynamics and nonlinear constraints (see, e.g., Grüne and Pannek (2017) for details). Considering applications in production planning and control, Tzafestas et al. (1997) introduced MPC for production inventory control and evaluated the applicability of this method for a production-inventory model in examples. For supply chain management, Fu et al. (2012) used MPC to reduce the bullwhip effect in a four-stage supply chain. Moreover, Chunyue et al. (2003) used a piecewise linearised production model.

To consider a multi-product manufacturing scenario, we study a real case modelled via a bond graph by Sagawa and Nagano (2015), convert it to an optimal control problem (OCP) and propose a respective model predictive controller. Based on this formulation, the problem is solved over a finite horizon, and the solutions are coordinated centrally and delegated to the machines at each iteration. The underlying bondgraph model, which depicts the intricate material flow among the machines, is converted to an OCP considering the constraints imposed by the material flow/production routings. To steer the system to an equilibrium, we impose steady-state working frequencies of the machines, which are adopted from an a-priori defined product mix. In previous publications con-

---

\* The first author thanks the University of Bremen for supporting this work by the BremenIDEA Out scholarship for a research stay in Brazil. The second author thanks the São Paulo Research Foundation (FAPESP) for the support to attend this conference [grant #2017/24716-6].

cerning the modelled system, the presented solutions are not optimal and the controllers steer the system based on local information only. In contrast to that, our approach allows to generate an optimal centralized solution based on global information. In addition, it allows representing the machines as autonomous elements, which enables to model more flexible configurations of production systems. The paper is organised as follows: In Section 2 we will derive the system model and formulate the optimal control problem. In Section 3 we show simulations, concluding the paper in Section 4.

## 2. PROBLEM SETTING

We consider a unidirectional job shop system based on a real-case scenario by Sagawa and Nagano (2015), producing polypropylene bags considered for packaging. The system consists of several stages with machines that are responsible for various processing steps. The input is given as one type of raw material (polypropylene pellets). As the product mix consists of nine types of polypropylene bags, after each stage, the manufactured intermediate products follow a different production flow. Considering the given a priori product mix, the types of bags are differentiated, if they have to be, e.g., laminated, printed or cut. Therefore, not all bags flow through all stages and may leave the system at different stages depending on the given product demand. The dynamic model of this system was defined in Sagawa and Nagano (2015) by utilising the bond graph syntax based on an adaptation of the electrical interpretation of bond graphs to manufacturing systems proposed by Ferney (2000). This modelling technique offers advantages regarding its generality, modularity and easiness of generation of the state model. A detailed overview about modelling abilities of the bond graph technique is given in Borutzky (2010). Here, the electrical interpretation is applied, which adopts Kirchhoff's voltage (*1-junction*) and current law (*0-junction*) regarding effort  $e$  and flow  $f$  to diverge and converge energy flows. In the manufacturing domain, these flows are interpreted as material flows or material rates.

In our approach, the machines are distributed and connected via the material flow, which creates a coupling or dependency among them, as will be presented in the following subsection.

### 2.1 Workstation Model

Each machine  $p \in P \subset \mathbb{N}$  in this model consists of a processing unit  $R: R_p = \frac{1}{U_p}$  with  $U_p$  as the processing frequency of machine  $p$ , a stock  $C_p$ , and a source of effort  $S_e^p$ . The first two elements are based on an analogy with a resistor and a capacitor from the bond graph definitions, respectively, and the source works as a coupling interface linking the stored material to the processing unit. Fig. 1 illustrates this model of a workstation. The input flow of material that feeds the stock of a workstation is defined as  $f_d^p$ , the intermediate flow from stock to processing unit as  $f_u^p$ , and the resulting output as  $f_y^p$ . The constitutive equation of an ideal capacitor  $e_C^p = (1/C_p)q_p$  establishes the relationship between the generalised bond graph variables of displacement  $q$  and effort  $e$ . In the

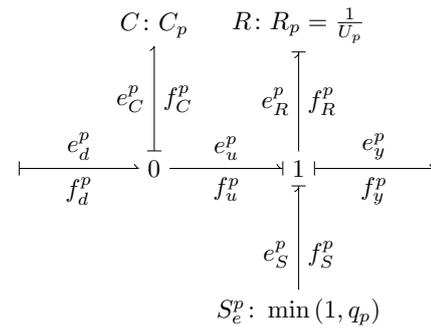


Fig. 1. Scheme of a workstation composed by a stock with capacity  $C_p$  and a processing unit with  $R_p = \frac{1}{U_p}$

electrical domain, it represents the relationship between the stored electrical charge and the applied potential difference, respectively. In the manufacturing domain, it determines the production volume  $q$  that is stored as a function of an effort, which in this case is interpreted as a coupling variable between the stock (buffer) and the succeeding machine (Ferney (2000)). The processing unit of a workstation is described by a 1-port resistor which follows  $e_R^p = R_p f_R^p \rightarrow e_R^p = \frac{1}{U_p} f_R^p$ . Stock and processing unit are connected by a coupling interface  $S_e^p$  which is a “switch” to represent material availability. Therefore, the effort to run the machine results from the provided material and the effort  $e_S^p$  to provide it. For more details please refer to Ferney (2000). The rate of material storage or consumption in a given intermediate stock can be calculated as the difference between the input flow from preceding workstations  $f_d^p$  and output flow  $f_u^p$  to the processing unit, that is,

$$\dot{q}_p = f_C^p = f_d^p - f_u^p. \quad (1)$$

Then, to calculate the buffer level, we integrate this rate over time  $t$ , that is,

$$\begin{aligned} q_p(t) &= q_p(t_0) + \int_{t_0}^{t_0+T} f_C^p(t) dt \\ &= q_p(t_0) + \int_{t_0}^{t_0+T} (f_d^p(t) - f_u^p(t)) dt, \end{aligned} \quad (2)$$

where  $T$  is the time step, and  $t_0$  the start time. As the machines are depending on a minimal material input flow (material unit) to start the process, the effort  $e_S^p$  is approximated by  $\min(1, q_p)$ , as with less than one material unit, the process is constrained by the fraction of the buffer level. Considering the properties of the 1-junction, which imposes equality of flows in the node ( $f_u^p = f_R^p = f_S^p = f_y^p$ ), and conservation of efforts (i.e. sum of inputs is equal to sum of outputs,  $e_R^p = e_u^p + e_S^p - e_y^p$ ), it is possible to express the output flow of a workstation as

$$f_y^p = f_R^p = U_p e_R^p = U_p \left( \frac{q_p}{C_p} + \min(1, q_p) - e_y^p \right).$$

Then, if the stock capacity is unbounded ( $C_p \rightarrow \infty$ ), the quotient  $q_p/C_p$  tends to 0. Considering that, this workstation will be coupled to the  $p + 1$  workstation downstream, also with unbounded capacity, then the effort  $e_y^p$  will be driven by the quotient  $q_{p+1}/C_{p+1}$ . This leads to the output flow

$$f_u^p = f_y^p = f_R^p = U_p \min(1, q_p),$$

and to the material storage rate, considering (1), of

$$\dot{q}_p = f_d^p - U_p \min(1, q_p). \quad (3)$$

As the machines are connected via the material flow, the material flow is split up (diverged) to feed several machines or converges into one machine. Therefore, for converging flows resulting in one flow  $f_p$ , we obtain

$$f_p = \sum_{k=1}^{|K_p|} f_k, \quad (4)$$

with  $K_p$  as the set of preceding flows immediately converging before workstation  $p$ . If  $p$  is fed by a partial flow diverged from the flow  $f_i$ , then we have

$$f_p = w_p f_i, \quad \text{with } \sum_{m=1}^{|M_p|} w_m = 1 \quad (5)$$

with  $M_p$  describing the set of immediately diverging flows before workstation  $p$ . A third situation, a combination of a converging/diverging junction in one stage may occur in the following cases: First, the output flows of some workstations are converged and immediately diverged in the same stage. Second, the output flow of a workstation may feed downstream workstations at several stages, e.g., if a fraction of the output flow of one workstation  $p-2$  is feeding its downstream workstation  $p-1$  and both, workstation  $p-2$  and  $p-1$  are feeding another workstation  $p$ . The combination of these converging/diverging situations is formulated as

$$f_p = \sum_{k=1}^{|K_p|} w_p v_k f_k \quad \text{with } k \in [1 : |K_p|], \sum_{i=1}^{|M_p|} w_i = 1 \quad (6)$$

$$v_k := \begin{cases} 1, & M_1 \cap K_p = \emptyset, \dots, M_{p-1} \cap K_p = \emptyset \\ w_j, & j \in \{M_1 \cap K_p, \dots, M_{p-1} \cap K_p\}. \end{cases} \quad (7)$$

Without  $w_p$  and  $v_i$ , (6) just describes a convergent junction as stated in (4). With the weight  $w_p$ , a possible diverging of flows before workstation  $p$  is incorporated, where the weights are defined in the diverging set  $M_p$  and  $w_p$  defines the fraction of  $f_k$ , which serves workstation  $p$ . Considering the second case, there would be two flows converging before workstation  $p$ , one coming from  $p-1$  and the other from  $p-2$ , so  $|K_p| = 2$ . On the other hand, there are two flows diverging before workstation  $p-1$  (coming from workstation  $p-2$ ), i.e.  $|M_{p-1}| = 2$ . In this case, one of the diverging flows before workstation  $p-1$  is actually the converging flow before workstation  $p$ , i.e.  $M_{p-1} \cap K_p$  is not empty. This intersection reveals the correct fraction  $v_k := w_j$ , which describes the second case of (7). If a converging flow comes completely from a given workstation, without diverging, this leads the first case of (7), i.e.  $v_i = 1$ . In order to define these couplings between the output flow of a given workstation and connected input flows of downstream workstations, it is necessary to firstly identify the set of workstations succeeding workstation  $p$ . Thus, we introduce the following algorithm, which yields such a set  $P_{p,succ}$  evaluating all connected workstations downstream, having  $f_y^p$  of workstation  $p$  as input:

---

**Algorithm 1** Find successive machines

---

```

1: Given a workstation  $p$ 
2: Set  $P_{p,succ} = \emptyset$ 
3: for  $k = p + 1$  to  $P$  do
4:   if  $f_y^p \in K_k$  according to (4)  $\vee f_y^p \in M_k$  according
     to (5) then
5:     Append workstation  $k$  to  $P_{p,succ}$ 
6:   end if
7: end for
8: Return  $P_{p,succ}$ 

```

---

Here, we additionally define  $P_{succ} := P_{j,succ}, j \in \{1, \dots, P\}$ . In the adopted approach, the machines are represented as a distributed systems. This allows to model more flexible production configurations instead of using an incidence matrix to represent the flow routings among the machines, which would be a more conventional approach.

## 2.2 Costs and optimal control problem

To formulate our optimal control problem, we first introduce a time discretization inducing an ordered set of time instances  $n = t_n$  such that  $t_n < t_{n+1}$ . Moreover, we adopt two cost functions, the first one based on the  $L_1$  norm, the latter one adopting a semi-quadratic approach. The first one leads to

$$\ell_{U_p}(q_p, U_p) = |U_p - U_p^*| + \lambda \Delta |U_p| + \mu q_p \quad (8)$$

regarding the current processing frequency and  $U_p^*$  the target frequency, where  $\Delta |U_p|$  represents the difference between the current and previous control, i.e.  $\Delta |U_p| = |U_p(n) - U_p(n-1)|$ ,  $0 < \lambda, \mu \leq 1$ . Then, the gradient is given by

$$\nabla \ell_{U_p} = \frac{U_p - U_p^*}{|U_p - U_p^*|} + \lambda \frac{\Delta U_p}{|\Delta U_p|} \quad \text{for } |U_p - U_p^*| \neq 0, \Delta U_p \neq 0. \quad (9)$$

The semi-quadratic objective is defined as

$$\ell_{U_p}(q_p, U_p) = (U_p - U_p^*)^2 + \lambda |\Delta U_p| + \mu q_p. \quad (10)$$

where the gradient is given by

$$\nabla \ell_{U_p} = 2(U_p - U_p^*) + \lambda \frac{\Delta U_p}{|\Delta U_p|} \quad \text{for } \Delta U_p \neq 0.$$

In both cases, we add a penalty on the control to avoid scattering behaviour and on the stocks to avoid a high level.

The dynamics from (3) is utilised to formulate a *centralised control problem*. Considering (3), which implicitly includes the condition of the existence of the predecessor material flow, we additionally add a constraint to avoid negative stock levels and compose the Optimal Control Problem (OCP) based on the cost function (8) or (10) over the prediction horizon  $N$ , which reads:

$$\underset{\mathbf{u}^*}{\operatorname{argmin}} J^N(\mathbf{u}^*; q^0) := \sum_{p=1}^P \sum_{k=0}^{N-1} \ell_{U_p}(q_p(k), U_p(k)) \quad (11)$$

with subject to the constraints

$$f_d^p(k) = \sum_{j=1}^{|K_p|} w_p v_j f_{y_j}(k), j \in [1 : K_p] \quad (12)$$

$$g_p^k := -q_p(k) \leq 0 \quad (13)$$

$$q_p(k) = q_p(0) + \sum_{k=0}^{N-1} [f_d^p(k) - U_p(k) \min(1, q_p(k))] \quad (14)$$

$$f_y^p(k+1) = U_p(k) \min(1, q_p(k)) \quad (15)$$

$$0 \leq U_p(k) \leq \overline{U}_p \quad (16)$$

$$\sum_{j \in P_{succ}} w_j v_p f_{dj}^p(k+1) = f_y^p(k+1) \quad (17)$$

where the optimal control frequency  $\mathbf{u}^*$  is denoted as  $\mathbf{u}^* := (u_1(0)^*, \dots, u_1(N-1)^*, \dots, u_P(0)^*, u_P(N-1)^*)$  and  $\sum_{j=1}^{|P_{succ}|} w_j = 1$ . (12)-(13) are defined for  $k \in [0 : N]$ , and (14)-(17) for  $k \in [0 : N-1]$ . The constraints refer to: the input flow of a given workstation coming from the predecessor workstations (12) based on (6), the non-negativity of the stock (13), the stock level (14), the output flow of the workstation resulting from the processing frequency (15), lower and upper bounds for the processing frequencies (16) and the output flow distributed for the successive workstations, guaranteeing that the output flow matches the input flows of the successive workstations (17).

### 3. SIMULATIONS

We conducted the simulations based on the given OCP (11). To this end, we used the processing frequencies presented in Sagawa and Nagano (2015), which were calculated to attend a certain product mix in the steady state (medium term). The implementation was carried out in C++ using the optimisation package NLOpt (Johnson (2004)). The parameters were set to  $\lambda = 0.2$  and  $\mu = 0.05$  to penalise a very high stock level. A prediction horizon of  $N = 8$  and  $N = 20$  was chosen to be able to incorporate all stages of the production system and to reduce an overshoot in the stock levels. The SLSQP method was used, (see Kraft (1994)), which is a gradient-based sequential quadratic programming approach. First, we present the results for minimising the  $L_1$  norm cost function (8) for both prediction horizons  $N = \{8, 20\}$  and for the semi-quadratic cost function (10), respectively. As we are aiming to see how the system converges with initial empty stocks, the time evolution of the simulation stops after 100 steps. As performance criteria, to compare the different objectives, we considered the calculated working frequencies obtained by solving the OCP (11) and the stock levels via (2).

The results for the  $L_1$  norm based cost-function (8) are illustrated in the Figures 2 and 3. As the  $L_1$  norm steers the system to the desired equilibrium fast, this adjustment leads to an overshoot in the stock levels and therefore an oscillation in the downstream machines. Although the frequencies seem to be converging with small oscillations, the stock levels change a lot, also due to the intrinsic flow. Considering  $N = 20$  for the  $L_1$  norm based cost-function, the results are shown in Figures 4 and 5. The longer prediction horizon leads even to more instability considering stock levels and frequency oscillations. This may occur because the process is partially ill-conditioned (see Pannocchia and Rawlings (2001) for details). For instance, this occurs for workstations 7 – 10, which are working in parallel and feeding the same subsequent workstations

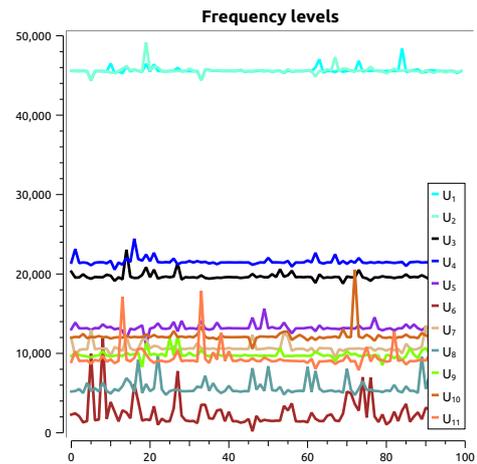


Fig. 2.  $U_{1-11}$  based on (8) with  $N = 8$

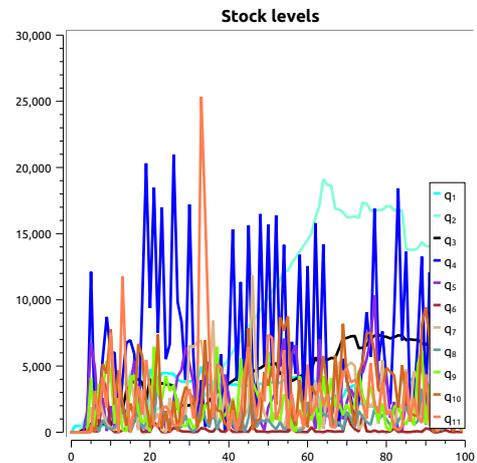


Fig. 3.  $q_{1-11}$  based on (8) with  $N = 8$

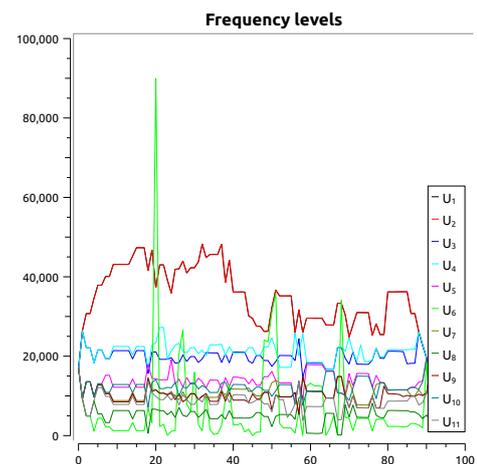


Fig. 4.  $U_{1-11}$  based on (8) with  $N = 20$

and product families: Various combinations of processing frequencies to control those workstations lead to the same output, as this is merged into a convergent junction, and also lead to the same costs. In combination with a long prediction horizon, this may lead to instability due to

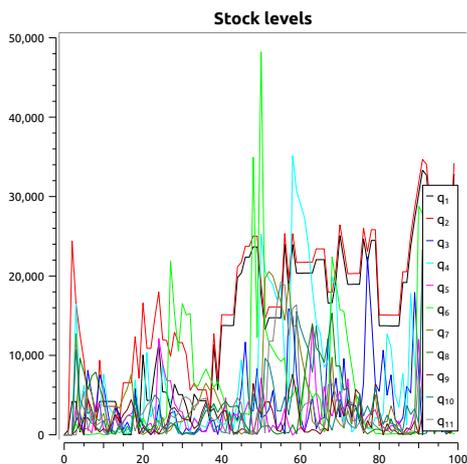


Fig. 5.  $q_{1-11}$  based on (2) with  $N = 20$

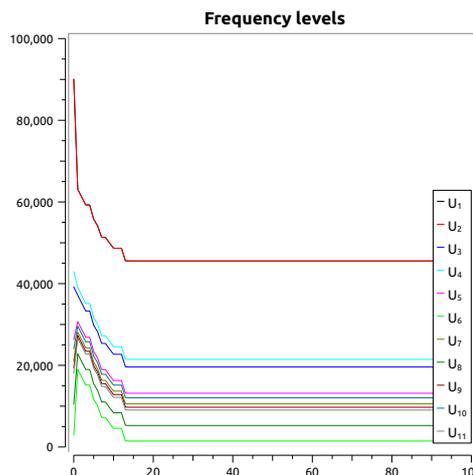


Fig. 8.  $U_{1-11}$  based on (10) with  $N = 20$

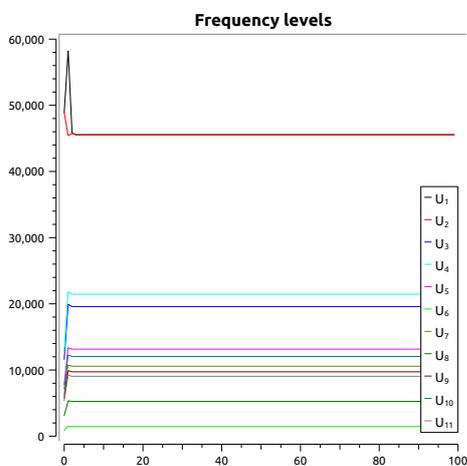


Fig. 6.  $U_{1-11}$  based on (10) with  $N = 8$

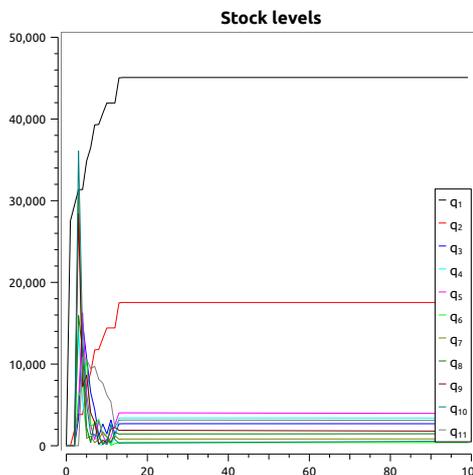


Fig. 9.  $q_{1-11}$  based on (10) with  $N = 20$

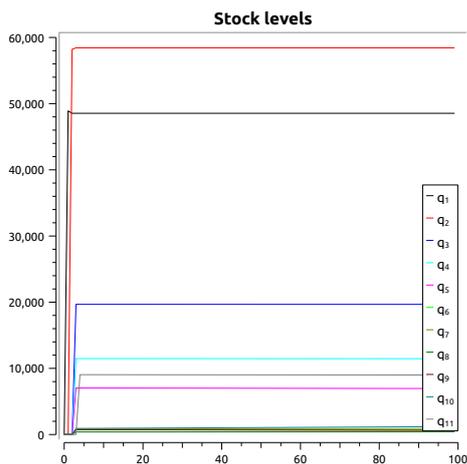


Fig. 7.  $q_{1-11}$  based on (10) with  $N = 8$

many combinatorial possibilities regarding a feasible solution (see Rossiter (2003) for details).

Considering the semi-quadratic cost function, results are depicted in Figures 6 and 7 for  $N = 8$  and in Figures 8 and 9 for  $N = 20$ . For  $N = 8$ , the controller steers

the workstations to the equilibrium very fast as a large distance from the equilibrium is incurring high costs. The frequency levels converge quickly while the stock level decreases slowly due to the small penalty. For  $N = 20$ , the equilibrium is reached later. Again, with the longer prediction horizon, the system tends to an overshoot as in the  $L_1$  norm case, but has converged around  $n = 15$ . Furthermore, the overshoot is also caused by the tendency to decrease the high costs as this is penalised higher for longer horizons. In accordance to the results of Rossiter (2003), we can conclude that the correct choice of a prediction horizon is important, especially, if the system tends to be ill-conditioned. Nevertheless, we observe that an appropriate choice of the cost function is also important to steer the system to a desired state especially with eventual conflicting objectives, see Marler and Arora (2004).

The results of Fig. 7 and 9 may help to answer the relevant question of how the level of work in process should be adjusted in each part of the manufacturing system, so that the operation is stable and a given product mix, in the medium term, is attended. Since the first two workstations (responsible for extrusion and weaving operations, respec-

tively) feed all the remaining operations, it makes sense that the stock levels  $q_1$  and  $q_2$  are higher than the remaining ones. Thus,  $q_1$  and  $q_2$  work as safety stocks. In addition, this proposed formulation, as an OCP problem, allows to obtain a centralised solution, overcoming a drawback of previous works. In the approaches presented by Sagawa and Freitag (2016) and Sagawa et al. (2017), based on classical control theory, there was one controller for each workstation and no linkage between the controllers, i.e. each controller would adjust the processing frequencies based on local information only, preventing an overall optimization.

#### 4. CONCLUSION

In this paper, we considered a real-case manufacturing shop floor, based on a bond graph model, and converted it to an optimal control problem to apply a model predictive controller. The controller is able to steer the manufacturing system using a quadratic cost function. Furthermore, the overshoot considering processing frequencies and stock levels is kept small for a sufficient long prediction horizon. We have shown in simulations, that a too long prediction horizon may slow the controller to steer such a system to a steady-state. The proposed modelling approach provides the opportunity of representing the workstations as a multi-agent system. Thus, further research should compare this centralized approach to a distributed setting. Furthermore, suitability of different cost functions should be considered, which is still an open topic in the field of dynamic modelling and control.

#### REFERENCES

- Abedinnia, H., Glock, C.H., Schneider, M., and Grosse, E.H. (2017). Machine scheduling problems in production: A tertiary study. *Computers & Industrial Engineering*. doi:10.1016/j.cie.2017.06.026.
- Borutzky, W. (2010). *Bond Graph Methodology*. Springer London, London. doi:10.1007/978-1-84882-882-7.
- Chunyue, S., Hui, W., and Ping, L. (2003). A receding Optimization Control Policy for Production Systems with Quadratic Inventory Costs. *IFAC Advanced Control of Chemical Processes*, 37(1), 713–717. doi:10.1016/S1474-6670(17)38817-1.
- Ferney, M. (2000). Modelling and controlling product manufacturing systems using bond-graphs and state equations: Continuous systems and discrete systems which can be represented by continuous models. *Production Planning & Control*, 11(1), 7–19. doi:10.1080/095372800232441.
- Fu, D., Dutta, A., Ionescu, C.M., and De Keyser, R. (2012). Reducing the bullwhip effect in supply chain management by applying a model predictive control ordering policy. In *14th IFAC Symposium on Information Control Problems in Manufacturing*, volume 14, 481–486. IFAC, Bucharest, Romania. doi:10.3182/20120523-3-RO-2023.00075.
- Grüne, L. and Pannek, J. (2017). *Nonlinear Model Predictive Control: Theory and Algorithms*. Communication and Control Engineering. Springer, London. doi:10.1007/978-3-319-46024-6.
- Johnson, S. (2004). The NLOpt nonlinear-optimization package.
- Kim, J.H. and Duffie, N.A. (2006). Performance of coupled closed-loop workstation capacity controls in a multi-workstation production system. *CIRP Annals - Manufacturing Technology*, 55(1), 449–452. doi:10.1016/S0007-8506(07)60456-9.
- Kraft, D. (1994). Algorithm 733; TOMP—Fortran modules for optimal control calculations. *ACM Transactions on Mathematical Software*, 20(3), 262–281. doi:10.1145/192115.192124.
- Lu, S., Su, H., Wang, Y., Xie, L., and Zhang, Q. (2015). Multi-product multi-stage production planning with lead time on a rolling horizon basis. *IFAC-PapersOnLine*, 28(8), 1162–1167. doi:10.1016/j.ifacol.2015.09.125.
- Ma, Y.H.K. and Koren, Y. (2004). Operation of Manufacturing Systems with Work-in-process Inventory and Production Control. *CIRP Annals - Manufacturing Technology*, 53(1), 361–365. doi:10.1016/S0007-8506(07)60717-3.
- Marler, R.T. and Arora, J.S. (2004). Survey of multi-objective optimization methods for engineering. *Structural and Multidisciplinary Optimization*, 395(6), 369–395. doi:10.1007/s00158-003-0368-6.
- Pannocchia, G. and Rawlings, J.B. (2001). Robustness of MPC and Disturbance Models for Multivariable Ill-conditioned Processes. Technical report.
- Qiu, R., Sun, M., and Lim, Y.F. (2017). Optimizing (s, S) policies for multi-period inventory models with demand distribution uncertainty: Robust dynamic programming approaches. *European Journal of Operational Research*, 261(3), 880–892. doi:10.1016/j.ejor.2017.02.027.
- Ratering, A.M. and Duffie, N.A. (2003). Design and Analysis of a Closed-Loop Single-Workstation PPC System. *Annals of the CIRP*, 52(1), 355–358. doi:10.1016/S0007-8506(07)60600-3.
- Rossiter, J. (2003). *Model-Based Predictive Control: A Practical Approach*. CRC Press.
- Sagawa, J.K. and Freitag, M. (2016). A Simulation Model for the Closed-Loop Control of a Multi-Workstation Production System. In *9th EUROSIM Congress on Modelling and Simulation*, September. Oulu, Finland. doi:10.1109/EUROSIM.2016.2.
- Sagawa, J.K. and Nagano, M.S. (2015). Modeling the dynamics of a multi-product manufacturing system: A real case application. *European Journal of Operational Research*, 244(2), 624–636. doi:10.1016/j.ejor.2015.01.017.
- Sagawa, J.K., Nagano, M.S., and Speranza Neto, M. (2017). A closed-loop model of a multi-station and multi-product manufacturing system using bond graphs and hybrid controllers. *European Journal of Operational Research*, 258(2), 677–691. doi:10.1016/j.ejor.2016.08.056.
- Tzafestas, S., Kapsiotis, G., and Kyriannakis, E. (1997). Model-based predictive control for generalized production planning problems. *Computers in Industry*, 34(1994), 201–210. doi:10.1016/S0166-3615(97)00055-9.

This article is printed under terms of Creative Commons CC-BY-NC-ND  
<https://creativecommons.org/licenses/by-nc-nd/4.0/> with all rights ©IFAC.

Original publication is available via <https://doi.org/10.1016/j.ifacol.2018.04.012>.

# A multi-product job shop scenario utilising Model Predictive Control

Tobias Sprodowski<sup>a,e,\*</sup>, Juliana Keiko Sagawa<sup>b,c</sup>, Arthur Sarro Maluf<sup>b,d</sup>,  
Michael Freitag<sup>a,e,g</sup>, Jürgen Pannek<sup>a,e,f</sup>

<sup>a</sup>University of Bremen, Faculty of Production Engineering,  
Badgasteiner Straße 1, 28359 Bremen, Germany

<sup>b</sup>Department of Production Engineering, Federal University of São Carlos,  
Rod. Washington Luís - Km 235, CEP: 13565-905, São Carlos, São Paulo - Brazil

<sup>c</sup>e-mail: juliana@dep.ufscar.br

<sup>d</sup>e-mail: maluf@ufscar.br

<sup>e</sup>BIBA - Bremer Institut für Produktion und Logistik GmbH at the University of Bremen,  
Hochschulring 20, 28359 Bremen, Germany

<sup>f</sup>e-mail: pan@biba.uni-bremen.de

<sup>g</sup>e-mail: fre@biba.uni-bremen.de

\*Corresponding author, e-mail: spr@biba.uni-bremen.de, fon: +49 421 218-50097

Multi-product manufacturing scenarios today have to face many challenges considering external factors such as availability of resources or attending product demands and internal factors such as adjustment of buffer levels or utilisation of workstations. In this paper, a multi-product job shop consisting of workstations coupled by an unidirectional material flow with different production routings is considered. The existing model based on the bond graph technique is adapted to an optimal control problem, allowing the applicability of the Model Predictive Control scheme. Concerning performance criteria, two different objective functions are defined: the first aims for predefined processing frequencies of the workstations and the second one takes into account product demands. Both approaches were examined in simulations showing that a steady state is achieved in terms of stable buffer levels and processing frequencies.

**Keywords:** Model Predictive Control; Production Control; Optimal Control; Capacity Adjustment;

## 1 Introduction

Decision-making in production scenarios has to take into account multiple products/parts, workstations and tools, as well as a myriad of information, e.g. product customisations, production routings, production and supplier lead times, capacity and storage constraints, disturbances etc. It is also well known that the pursued performance goals usually involve trade-offs (**Bechte1988**). Although cost minimization is a prevalent managerial goal, the demand for short delivery times compels to find a balance between workstation utilisation levels and acceptable throughput times (**Wiendahl2007**). This balance, on its turn, is directly related to the definition of work in process levels (**Henrich2004**). These are defined as the difference between the input and output flow rates of a workstation, also incorporating failures and disturbances (**Duffie2009**).

Different methods from various research disciplines have been utilized to formulate mathematical models for production systems, e.g. mathematical programming, heuristics (**Bonivento2008**) and meta-heuristics for production scheduling (**Abedinnia2017**; **Beheshti2015**), queuing systems (**Marsudi2013**), and so on. With incorporation of disturbances, i.e. the dynamics of the production systems, control methods, which are widespread in system theory, offer an adequate range of tools (**Wiendahl2000**; **Duffie2014**) to stabilise or steer such an overall system to a defined goal.

In this paper, a multi-product scenario is considered (**Sagawa2015a**), which involves different workstations where various products are manufactured in an unidirectional job shop. Since the products are customised, the respective production routings differ. Each workstation consists of a buffer, which may be connected upstream to another workstation, and a processing unit, filling the downstream workstations. Each workstation has its own production rate or processing frequency, partially depending on the provided material, while these rates/frequencies have to be adjusted to lead the overall system to a steady state. The goals of controlling such a production system are to fill the buffer of the workstations to an adequate level for safety reasons and to fulfil the customer's demand considering a defined product mix. While the original model was defined using the formulation of bond graphs based on an adaptation from **Ferney2000**, the dynamics and control of the system are formulated as an optimal control problem (OCP), cf. **Sprodowski2018**.

Several production-inventory models have been formulated as OCPs, as it will be presented in the literature review section. Optimal solutions were found for problems that take into account a variety of conditions and factors, and these solutions were proven to be optimal. However, there are assumptions in the models which hinder their application to real life problems, e.g. the consideration of single-product or single-workstation systems, or of a discrete set of capacity adjustments, such as either full capacity or null capacity. These limitations will be discussed in the next section. Moreover, in several OCPs, the shop floor is modelled as a Flexible Manufacturing System (FMS), where a given workstation can process all the operations needed to deliver the final products. This configuration does not fit conventional production systems, where the machines are more specialised: the workstations have different functions and the parts are manufactured according to specific routings.

The main contribution of this paper is to present an alternative formulation for the OCP aiming to overcome some limitations of the existing models: the limitations regarding single-product and/or single-workstation models, and the type of production flow. Although trying

to keep the formalism of the analytical formulation, the primary focus is set on the application perspective.

In the proposed model, each workstation applies a control policy, which is calculated following a centralised Model Predictive Control (MPC) scheme. The scheme relies on the formulation of an OCP minimising a cost function with respect to a given model and constraints over a fixed prediction horizon. Then, a sequence of optimal control values is obtained and the first control value is applied. The procedure is repeated until a given target condition is met.

As mentioned, the original model revisited in this paper is based on bond graphs (**Sagawa2017**). Thus, besides the motivation mentioned (i.e. overcoming practical limitations of applying optimal control to different multi-product multi-station production systems), this integration between bond graphs and Model Predictive Control devoted to the modelling of production systems is also novel. Bond graphs can be used to model dynamic systems from different fields and physical domains (and also hybrid systems) by means of an integrated graphical representation. The approach presents advantageous features such as simplicity of representation and understanding, generality and modularity (**Komurgoz2019861**; **Srinivasarao2020**). On the other hand, the obtained dynamic model and the choice of control system for it are independent from each other. Since the bond graph technique generates state space models, one natural and usual choice is the use of full-state feedback control, such as in **Sagawa2019**. There are prescriptive rules and procedures for tuning this kind of controllers; however, they are not optimal. Thus, improving the quality of the controllers for complex multi-product multi-station production systems is also one of our motivations for proposing this approach. This integration between bond graphs and MPC for production systems is novel and is not direct, that is, it requires remodelling of the system to apply the MPC. The focus of this work is set on this remodelling.

In this paper, a centralised control scheme was applied. However, the presented mathematical modelling allows considering each production station as an independent entity/agent, and this allows a future extension of this work towards decentralized control architectures, which may be applied to cyber physical systems in the context of Industry 4.0 (**Cheng2018**). This is an additional motivation of the presented modelling.

In the MPC scheme proposed, the optimisation evaluates two performance criteria, which follow two different control policies: Firstly, considering the operational level and based on an a-priori defined product demand, the optimal working frequency for each workstation is tracked by solving an OCP in each time instant by gradient-based and gradient-free algorithms. As the outputs (material flows) are directly affected by the control inputs, both gradient-based and gradient-free algorithms are able to converge to the target frequencies after a short initialisation phase.

Secondly, as a mixture of tactical and operational level, the product demands of the product mix are used to calculate the individual needed production volume for each workstation. This provides to the optimiser more freedom to choose the processing frequencies of the workstations according to the overall product mix. Here, as the control inputs are delayed regarding the measured outputs, the MPC has to be adjusted to a specific prediction horizon to render the system practically stable. Both objectives are utilised in simulations to examine the performance criteria regarding buffer level and stability of processing frequencies as the first criterion is important to keep the workstations running and the latter to reduce

the necessity of reconfigurations.

**Notation:** We abbreviate a set  $\{0, 1, \dots, N\}$  with  $[0 : N]$ ,  $\mathbb{N}$  and  $\mathbb{R}$  as the set of natural and real numbers.

## 2 State of the art

In this section, we cover the scope of models based on dynamic programming and optimisation for controlling manufacturing systems. These models are capable to control both a discrete or continuous formulation, depending on the necessity to render the underlying application to the defined equilibrium. In combination with the formulation as an OCP, this allows the applicability of MPC in production control.

Historically, dynamic programming was applied to the basic problem of inventory control in **Arrow1958** and, since then, its use became ubiquitous. Production and inventory problems can be formulated as stochastic OCPs, hence they can be seen as dynamic problems that involve uncertainties (**bertsekas1995dynamic**). The very first problem comprised a single-echelon and single-product system where the state variable is the inventory level at the beginning of a time period, the control variable is the purchase or production and the disturbance variable is the stochastic demand. The solution of this fundamental problem yields the classical base-stock or order-up-to policy,  $(s, S)$ , which is still a standard applied in the operations management area (**Qiu2017**). Several extensions were built upon this basic knowledge block, including: multi-echelon single-product systems (**Sethi1997**), single-echelon multi-product systems (**Cattani2011**), problems considering setup costs (**Allahverdi2015**), infinite horizon, time-variant demands (**Afzalabadi2016**), extra information about demand forecasts (**Hartzel2017**) and unreliable workstations (**Shi2014**). For a more detailed review, see **Sarimveis2008**.

First results considering a discrete order function were stated in **Lippmann1969**. Assuming, that inventory costs are sub-additive, an optimal purchase policy is derived for one time period incorporating production, including demanded, but not fully-used resources, holding and setup costs as step function, which incorporates quantised capacities, e.g. transportation vehicles.

In order to extend the inventory problems towards multi-product and multi-workstation models, some authors **Sethi1992** presented a model of a FMS with  $m$  workstations producing  $n$  parts or products, where the products can be processed in any of the workstations. In the model of **Sethi1998**, the total production rate at time  $t$  is given by  $u_i(\cdot) = \sum_{j=1}^m u_{ij} \leq k, i = 1, \dots, n$ , with  $u_{ij}$  describing the production rate of product  $i$  on workstation  $j$  bounded by production capacity  $k$ . This concept of FMS is similar to the definition of an open shop system used in the traditional field of Operations Scheduling. For different real-time scheduling policies the authors provide properties for feasibility under the condition that the workstation workload is not matching the maximum level. **Sethi1998** dealt with a multi-product system with single or parallel workstations, and convex cost of holding and backlogging. The workstations have two states, up and down. They used a vanishing discount approach to address the average cost problem, and built upon the methodology proposed by **Presman1995** for optimal control of flow shops. The latter provided a verification theorem and the existence of an optimal feedback control based on

the Hamilton-Jacobi-Bellman equation in terms of directional derivatives (HJBDD). Based on this idea, **Sethi1998** obtained a solution of the HJBDD equation by using convex analysis and properties of viscosity solutions and directional derivatives, for the multi-product model deriving a optimal control policy.

In **Gharbi2003**, the workstations were subject to breakdowns and repairs and its production rate may assume three values, namely maximal production rate, demand rate and zero for each product. The analytical formalism was used to obtain the structure of the solution, i.e. the structure of a hedging point policy, relying on the fact that the value function is the solution for the associated HJB equations of the considered problem. The aforementioned authors showed that the complexity of a two-workstation five-product manufacturing system with different states prevents the implementation of a numerical algorithm to obtain the optimal solution. Thus, based on the structure of the hedging point policy, a parametrized near-optimal production policy was proposed, and its parameters were optimized by means of experimental design (DOE) and surface response methodology.

Other solution methods utilising a rolling planning horizon, broadly applied in production-inventory systems, are covered by metaheuristics. The problem setting for these scenarios is formulated as a mixed-integer job assignment problem, where solution candidates are generated by using either a genetic algorithm for a lot-sizing problem in flow shops (**Mohammadi2011**) or applying the simulated annealing algorithm to calculate an initial feasible solution in a rolling planning horizon manner (**Torkaman2017**). **Mohammadi2011** solve the integrated problem of lot sizing and scheduling in a permutational flow shop configuration. Their model is discrete, and they present a centralized solution. In contrast, the focus of our application (and of the papers reviewed in this section) is set on lot sizing in production-inventory systems, we consider an unidirectional job shop configuration, and the base of our model is continuous (production flow). We also present a centralized solution, but the presented formulation also allows the implementation of a decentralised solution. The formulation of the OCP in our setting is stated as a problem based on a continuous differential cost function and constraints due to continuous and variable lot sizes. This enables us to utilise exact methods, which were otherwise hard to apply due to a higher computational effort if combinatorial issues, e.g. stemming from fixed lot sizes, have to be incorporated. Thus, in particular MPC or receding horizon control (RHC) could be applied here providing a suboptimal solution over a finite prediction horizon length, which use, in a rolling horizon manner, the previous solution from the preceding time instant as a warm start. For an analysis in detail about suboptimality and prediction horizon length, see **Grune2017**.

While in control theory a variety of possibilities to derive control laws exists, one technique to integrate an OCP is the MPC Scheme, which has been widely adopted and implemented in industrial applications (**QinBadgwell2003**; **Xi2013**). First applications utilising the MPC scheme considered chemical systems incorporating slow changing dynamic (**Rivera1992**). **Ivanov2012** surveyed the approaches to formulate the Supply Chain Management (SCM) problem as an OCP. The authors emphasised the importance of availability of full information among the supply chain participants to keep such a system stable. They concluded that central formulations of such problems are broadly explored, while distributed systems are still to be explored to allow decentralised decision making.

**Sarimveis2008** provided a broader review, covering different control theory approaches, varying from transfer function formulations (i.e. classical control theory) to adaptations of

MPC, where an underlying OCP is utilised. Although the formulation of an OCP allows flexibility in the choice of control methods, still the challenge is to obtain a solution, able to steer such an OCP to an equilibrium or setpoint. **Kouvaritakis1996** provided properties for unstable systems by limiting the control input a-priori and tight those input constraints for future predicted steps to retain stability and furthermore reduce the computational load.

Considering manufacturing scenarios, **Tzafestas1997** introduced MPC for production planning processes. The authors examined inventory scenarios and marketing decision processes to strike out the improved performance following a production or inventory objective by minimising the production effort. Focusing on the bullwhip effect in SCM, **Fu2012** utilised MPC to reduce this undesired behaviour. Applied to a linearised, discrete four-stage example, the authors showed that the centralised MPC reduces this effect by tracking a reference.

Considering applications in production control especially for scheduling and planning problems, **Gimeno1997** applied the MPC scheme for a batch multi-purpose plant with four stages, which is flexible to perform several tasks. The problem was formulated as a Mixed Integer Linear Problem (MILP) which obtains an optimal order of the operation tasks regarding a short term policy. Incorporating failure-prone workstations, **Rui2007** applied MPC for a multi-product single workstation using a hedging point control policy. The authors emphasised the possibility of a sub-optimal inventory control due to the finite set of products, which increases the probability of similarity of some parts of the production sequences. Then, with an imposed feedback, the control input may be updated to calculate a new policy if a failure occurs. **Cataldo2015** applied an online-optimisation MPC scheme for an multi-stage plant. The control was imposed on the working speed of this plant by considering a minimal consumption of energy and fulfilling a minimum production requirement. The working states of the workstations, which include also the shifting of the parts, the energy consumption, and physical constraints e.g. the maximum allowable power, were implemented in a Mixed Logical Dynamic Model to allow binary or logical states. **Chunyue2003** implemented, for a multi-product system, MPC regarding quadratic inventory control to minimise storage time for short-term products. To reduce the computational burden, the problem was split up into a static problem considering terminal setup costs and production rate, and a dynamic problem as a piecewise deterministic process considering real-time production and failure rates based on the age of the system.

Using a semi-conductor benchmark example from Intel using two product types, **Jang2013** applied a centralised MPC control approach targeting for an a-priori WIP level for each workstation.

The discussed contributions reveal a need for alternative formulations of multiple-product multiple-workstation systems, aiming to approximate the models to applications in real systems and to cope with the curse of dimensionality of the problems regarding multi-products and configurations of many workstations as mentioned by **Gharbi2003** and **Sarimveis2008**. Concerning an applicable solution, analytical approaches may be unable to obtain a feasible solution due to the complexity of the overall problem, which leads to customised heuristic approaches.

These aspects are important if applicability is emphasised, instead of the improvement of the method itself. Here, the applicability is prioritised and the aim is to contribute in this direction, extending/broadening the assumptions of the models, as stated in the introduction. Moreover, the formulation of production systems as FMS presented in this section does not

apply for the majority of conventional production systems, where the workstations have specific functions, and the shop floor has either a flow shop or a job shop configuration. The model proposed by **Presman1995** provides an extension of this restriction, since it considers a n-workstation job shop, but limits to a one single-product manufacturing.

Following MPC approaches in manufacturing scenarios, most approaches focus on single products (using production rates) or single workstation models, as discussed. Other approaches aim at a separation of offline and online problems to reduce the computational burden. The focus of our problem is to depict a real-case multi-product job shop scenario with focus on the integration of the production flow, relaxed by the assumption of unlimited buffers. The control scheme is based on MPC, solving the underlying OCP representing the model of the workstations. Regarding comparability to other control theory approaches, e.g. P(ID) controllers, a numerical comparison with these approaches is not considered as the choice is driven by different reasons: While on the one hand, P(ID) controller approaches have to be tailored with respect to the given model and a given objective, their advantage come by a reduced computational effort and high effectiveness with optimised parameters. On the other hand, solving an OCP in the MPC scheme demands increased computational effort, but allows for higher flexibility as the objective function may be modified easily while the underlying model has not to be adapted.

The formulation of the OCP proposed in this paper is based on a bond graph model of a production system. The bond graph method is based on the characterization of the power exchange in the system (**KarnoppD.MargolisD.L.Rosenberg2006**), and power is defined as the multiplication of the generalized variables of effort (e) and flow (f) (**Borutzky2010**). Momentum and displacement are also generalized variables resulting from the integration of effort and flow. The idea of the bond graph draws back to the 1950s on describing the dynamics of physical systems (**KarnoppD.MargolisD.L.Rosenberg2006**), able to cover all kinds of systems which can be reduced to energy domain based systems, such as electrical, mechanical, hydraulic or thermodynamic based systems (and its combinations, such as mechatronic systems). The modelling consists of dividing systems into increasingly simpler subsystems, constituted by basic components. These components present the phenomenological aspects in the system: a) supply, b) storage, c) dissipation, d) transfer or conversion of energy. For each of these elements, the corresponding bond graph elements are: a) sources of flow and effort, b) capacitor/compliance and inertia; c) resistor; d) 0 and 1-junctions, transformer, gyrator. The components are connected by bonds through energy ports. These bonds represent a pair of an effort and flow.

The bond graph technique allows for modularity to encapsulate or aggregate systems into hierarchical levels. This modularity allows modelling manufacturing systems with different numbers of machines and different configurations, such as parallel machines, simple and parallel flow shops and job shops which do not need to be balanced. All these systems are composed of the same basic entities; thus, the mathematical representation of different models does not depend on the formulation of a different transfer function for each particular system, it requires only different combinations of the basic entities. In terms of mathematical modelling, the state representation can be obtained by ordinary algebraic manipulation of the constitutive equations of basic entities, and not by the deduction of differential equations and their conversion to transfer functions, for each particular case (**Sagawa2017**). Their use to the representation of production/manufacturing systems is a novel application. After

**Ferney2000**, proposed bond graph subsystems to represent basic manufacturing entities, few subsequent works have explored this research stream (cf. **Sagawa2015a**, **Sagawa2017**, **Sagawa2018825**, **Sagawa2019**).

In this paper, we limit the model to directed bond graphs as no reentry-order flows to preceding workstations is considered. The participants can be either energy sources or sinks, stores (capacitors), dissipators (resistors) or power couplers. To interpret these terms for the manufacturing system, the flow is regarded as material flow, and the effort represents the leading or not-leading property of a workstation, feeding the successive stock of a succeeding one, i.e. the coupling of a workstation depends on its predecessors' material input. If a buffer of a workstation is running empty, no material is available from its predecessors, i.e the effort is zero.

As mentioned in the introduction, one contribution of this paper is to develop the integration of bond graphs and Optimal Control for the representation and optimization of production systems. The first step towards this direction was taken by **Sprodowski2018**.

### 3 Problem setting

In this section, we recap the underlying workstation model regarding the notation in **Sprodowski2018**. The proposed approach here consists of the underlying bond graph model and builds on the proposed algorithms and hence, reveals a modular 3-tier architecture: First, we define in the ensuing Section 3.1 the bond graph model with the workstations as independent entities, where the successive and predecessive workstations in terms of depending material flow are evaluated by each workstations in Alg. 1 and 2. Second, to obtain a valid demand and material flow also in terms of future time predictions, the distance to the source flow for each workstation is calculated via Alg. 3 in Section 3.2. To propagate the demand correctly, Alg. 4 defines the backward propagation, which originally stems from the target demands of each product routing. In Alg. 5 the forward propagation is calculated to aggregate the correct demand for each workstation. At last, in Section 3.3 the OCPs are formulated for the frequency control in (18) and in (25) for the demand. The latter, based on the demand-based costs defined in (13)–(17), utilises the aforementioned algorithms to determine the demand for each workstation.

#### 3.1 Bond graph model

An uni-directional, multi-stage job shop is considered based on **Sagawa2015a**, which was modelled utilizing the bond graph approach modified for manufacturing systems formulated by **Ferney2000**, deriving the basic components from the electrical domain with voltage (effort  $e$ ), current (flow  $f$ ), linkage flux (momentum  $p$ ) and charge (displacement  $q$ ). To distribute energy between elements, two types of nodes are used: Firstly, *0-junctions*, which impose equal efforts around the junction and conservation of flow (Kirchhoff's current law):

$$\begin{aligned} e_1 &= \dots = e_i \\ f_1 &= f_2 + \dots + f_i, \end{aligned} \tag{1}$$

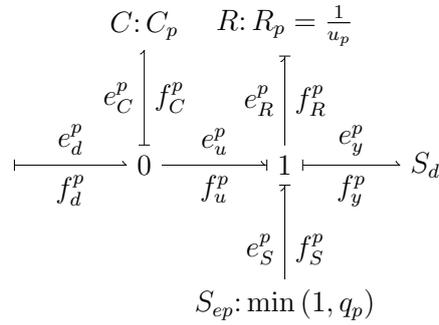


Figure 1: **Scheme of a workstation** composed by a stock with capacitance  $C_p$  and a processing unit with  $R_p = \frac{1}{u_p}$

with  $f, e \in \mathbb{R}^{\geq 0}$ . Secondly, *1-junctions* (Kirchhoff's voltage law), which impose equal flows and conservation of efforts, are defined as:

$$\begin{aligned} f_1 &= \dots = f_i \\ e_1 - \dots - e_i &= 0. \end{aligned} \quad (2)$$

Following **Borutzky2010**, a 1-port energy storage is defined as linear capacitor, which reveals  $q = Ce$  and is utilized as buffer of material, while machines are delineated by 1-port resistors **Borutzky2010**, which follows the linear characteristic  $e = Rf \rightarrow e = \frac{1}{u}f$ . According to **Borutzky2010**, the source flow is given by the incoming material flow and the resulting production of the several products are the sink flows, therefore, this system can be classified without the loss of generality as a multiport system.

Each workstation  $p \in P \subsetneq \mathbb{N}$  consists of a capacitor  $C_p$  (buffer), a resistor (with processing unit  $R_p = \frac{1}{u_p}$ ), and a source of effort  $S_{ep}$ , describing the effort to pull the material from the buffer to the processing unit if material is available ( $\min(1, q_p)$ , see Fig. 1). Input and output flows are given as  $f_d^p$  and  $f_y^p$ , the latter feeding subsequent workstations or the product outcome  $S_d$ . Based on the principles of the 0-junction (see Eq. 1) and on the fundamental relationships established among the bond graph generalized variables, the difference between the input  $f_d^p$  and output flows  $f_y^p$  of buffer  $C_p$  for a workstation  $p$  is given as the flow rate

$$\dot{q}_p = f_C^p = f_d^p - f_y^p, \quad (3)$$

and the effort for the capacitor is defined as

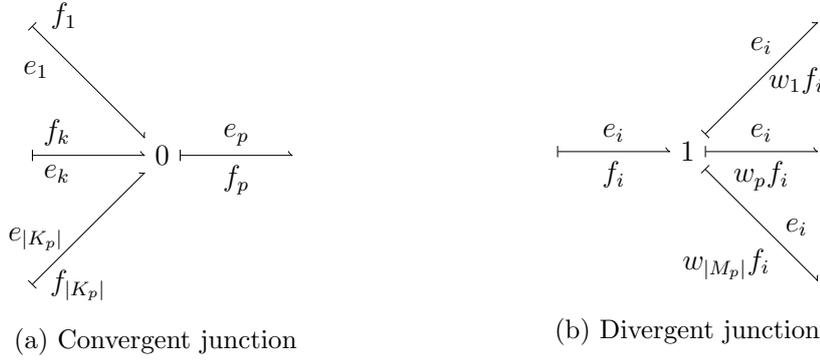
$$e_d^p = \frac{q_p}{C_p}. \quad (4)$$

As the *1-junction* (2) retains the equality of all input and output flows we obtain

$$f_u^p = f_R^p = f_y^p = f_S^p,$$

where the flow for the resistor composing the working unit is defined as

$$f_R^p = u_p e_R^p \quad (5)$$

Figure 2: **Types of junctions**

with  $u_p$  as the processing frequency revealing the first performance criterion. According to the property of the *1-junction*, the effort for a workstation results to

$$e_R^p = e_u^p + e_S^p - e_y^p \quad (6)$$

with  $e_S^p = \min(1, q_p)$ , as the effort is bounded by  $0 < e < 1$ , allowing a fraction of material according to **Ferney2000**. Inserting this and (4)–(5) into (6) leads to

$$f_R^p = u_p \left( \frac{q_p}{C_p} + \min(1, q_p) - e_y^p \right).$$

In this model, storage of the buffers is unbounded, i.e.  $C_p \rightarrow \infty$  which leads to  $e_{dj} = 0$ , cf. (4). As the output effort of a workstation is the input of the successive workstation, we obtain  $e_y^p = e_{dj}$ ,  $j \in P \setminus \{p\}$  with  $j$  as the successive downstream workstation. Hence, it can be concluded that

$$f_y^p = f_R^p = u_p \min(1, q_p).$$

Together with (3) we obtain

$$\dot{q}_p = f_d^p - u_p \min(1, q_p), \quad (7)$$

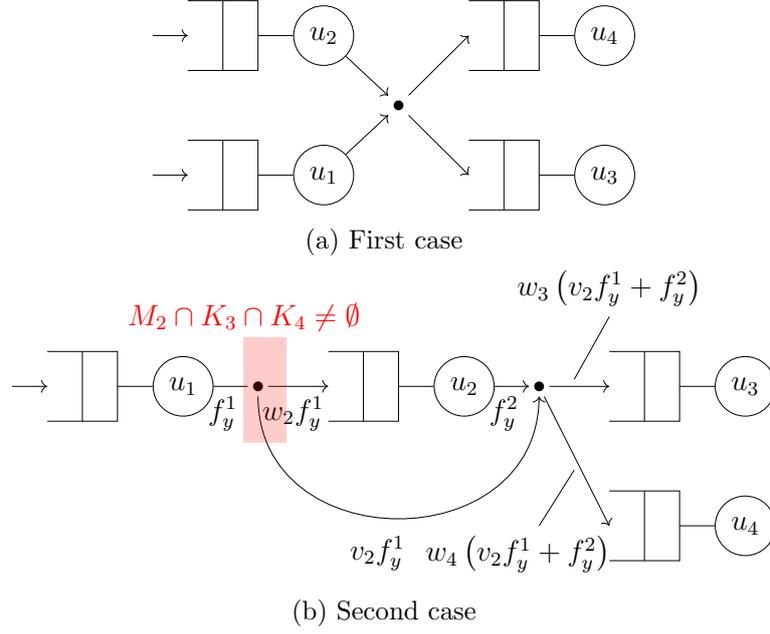
which represents the rate of material storage or consumption in a workstation  $p$ .

Here, convergent and divergent junctions are used to merge or to split up material flows among workstations: The convergent junction (Fig. 2a) describes the union of flows from converging to one flow  $f_p$  before workstation  $p$  with

$$f_p = \sum_{k=1}^{|K_p|} f_k \quad (8)$$

where  $K_p$  represents the set of cumulating flows before workstation  $p$ . A divergent junction, however, corresponds to a split of the material flow, cf. Fig. 2b, and is defined as

$$f_p = w_p f_i, \quad , \text{with} \sum_{m=1}^{|M_p|} w_m = 1 \quad (9)$$

Figure 3: **Convergent-divergent junction**

with  $M_p$  describing the set of immediately diverging flows before workstation  $p$ , and  $0 < w_p \leq 1$  defining the fraction of the diverged flow that feeds workstation  $p$ .

A third situation, a combination of a converging/diverging junction in one stage may occur in the following cases: First, the output flows of some workstations are merged and immediately split in the same stage, illustrated in Fig. 3a with a reduced schematic depicting the preceding buffers and connections among the workstations.

Second, the output flow of a workstation may feed downstream workstations at several stages, see Fig. 3b: For instance, a fraction of the output flow  $f_y^1$  of workstation 1 is feeding its downstream workstation 2, and both, workstation 1 and 2 are feeding workstations 3 and 4 with  $v_2 w_3 f_y^1 + w_3 f_y^2$  and  $v_2 w_4 f_y^1 + w_4 f_y^2$ , respectively. The fraction  $v_2$  describes the diverging flow of  $f_y^1$ , which feeds both workstations 3 and 4.

The combination of these converging/diverging situations is formulated as

$$f_p = \sum_{k=1}^{|K_p|} w_p v_k f_k \text{ with } k \in [1 : |K_p|], \sum_{i=1}^{|M_p|} w_i = 1 \quad (10)$$

$$v_k := \begin{cases} 1, & M_1 \cap K_p = \emptyset, \dots, M_{p-1} \cap K_p = \emptyset \\ w_j, & j \in \{M_1 \cap K_p, \dots, M_{p-1} \cap K_p\}. \end{cases} \quad (11)$$

With (10) and (11), it is possible to model an asymmetric junction of material flow, where a workstation  $p$  receives a material flow of preceding workstations in different stages: In the first (symmetric) case of (11), all the preceding workstations of  $p$  are on the same stage, that is, no direct preceding workstation  $p - i$ ,  $0 < i < p$  is concurrently feeding other preceding workstations of  $p$  (c.f Fig. 3a), and the diverging flow before workstation  $p$  is defined by fraction  $w_p$ , as  $v_k = 1$ . This corresponds to a regular divergent junction.

For the second (asymmetric) case, if an upstream workstation  $p - i, 0 < i < p$  connected to  $p$  feeds another preceding workstation  $k, p - i + 1 < k < p$ , which also feeds  $p$ , the diverging output flow of workstation  $p - i$  has to be considered: If the diverging flow of this workstation  $p - i$  is included in the set of diverging flows  $M_j$  of the intermediate workstation  $k$  as well as in the converging set  $K_p$  of workstation  $p$  ( $M_k \cap K_p$ ),  $v_k := w_j$  reveals the correct fraction of the bypassed material flow of  $p - i$  on this junction. C.f. Fig. 3b for an example: workstation 1 as  $p - i$  is feeding 2 - 4, 2 as  $k$  is feeding 3 and 4, which could be  $p$  at any one time. Then,  $v_2 f_y^1$  of workstation 1 is incorporated in  $K_3, K_4$ , and  $M_1$ , indicated by the red marking. Thus, this allows to retrace the direct ratio of material flow from this feeding workstation  $p - i$  for the considered workstation  $p$  also incorporating the weight  $w_p$ , while  $w_p < 1$  indicates the diverge before workstation  $p$ .

### 3.2 Successors, Predecessors & Demands

To evaluate flows after workstations and to be able to evaluate flows without successors, i.e. resulting in a finished product,  $\bar{K}_p$  is defined as the set of converging flows after workstation  $p$ , and  $\bar{M}_p$  the set of diverging flows after workstation  $p$ . Regarding the routing of the material flow, the dependencies among the workstations have to be analysed, as subsequent workstations connected to one workstation upstream are dependent on its output flow. In other words, it is necessary to establish the product routings and workstations for a given product mix. Therefore, to allow an examination in both directions (predecessors and successors), the following two algorithms are calculating the sets: The first one calculates the set of predecessors of workstation  $p$ , denoted as  $P_{p,pred}$ , i.e. workstations, which are feeding  $p$ . The latter is calculating the set of successors of  $p$ , denoted as  $P_{p,succ}$ , i.e. workstations, which are feeded by  $p$ . Both algorithms are initialised with a given workstation  $p$  and create the sets by examining the diverging and converging set of each workstation. Alg. 1 examines

---

#### Algorithm 1 Find successive machines

---

**Require:** workstation  $p$ ,  $f_y^p$  of  $p$

- 1: **Set**  $P_{p,succ} = \emptyset$
  - 2: **for**  $k = 1$  to  $P$ ,  $k \neq p$  **do**
  - 3:     **if**  $f_y^p \in K_k$  according to (8)  $\vee f_y^p \in M_k$  according to (9) **then**
  - 4:         **Append**  $P_{p,succ} \cup k$
  - 5:     **end if**
  - 6: **end for**
  - 7: **return**  $P_{p,succ}$
- 

the output  $f_y^p$  of workstation  $p$ : if  $f_y^p$  is included in the converging (or diverging) set of flows  $K_k$  (respectively  $M_k$ ) of another workstation  $k$ ,  $k$  is a direct successor of  $p$  and is added to the set of successive workstations ( $P_{p,succ}$ ). Equivalently, this is also done to evaluate the set of preceding workstations of  $p$  in Alg. 2 utilising the input  $f_d^p$  of workstation  $p$ .

To incorporate delays to provide the material for a specific workstation, the distance  $d_p$  of the latter in relation to the source (i.e. provider of the raw material) is calculated via the recursive Alg. 3. This algorithm is initialised by each workstation with  $d_p = 0$ . First, the set of predecessors is obtained by Alg. 2. Then, after incrementing the distance by one, the

---

**Algorithm 2** Find preceding machines

---

**Require:** workstation  $p$ 

- 1: **Set**  $P_{p,pred} = \emptyset$
  - 2: **for**  $k = 1$  to  $P$ ,  $k \neq p$  **do**
  - 3:     **if**  $f_d^p \in \bar{K}_k$  according to (8)  $\vee f_d^p \in \bar{M}_k$  according to (9) **then**
  - 4:         **Append**  $P_{p,pred} \cup k$
  - 5:     **end if**
  - 6: **end for**
  - 7: **return**  $P_{p,pred}$
- 

---

**Algorithm 3** Calculate maximum (**minimum**) distance to source

---

**Require:** workstation  $p$ , distance  $d_p$ 

- 1: **Calculate**  $P_{p,pred}$  via Alg. 2
  - 2: **for all**  $j \in P_{p,pred}$  **do**
  - 3:     **Set**  $d_p = d_p + 1$
  - 4:     **Obtain**  $d_j$  by calling Alg. 3
  - 5:     **if**  $d_j > (<)d_p$  **then**
  - 6:         **Set**  $d_p = d_j$
  - 7:     **end if**
  - 8: **end for**
  - 9: **return**  $d_p$
- 

procedure is recursively called for all predecessors. If no predecessors are available, i.e. the source is reached, then the calculated distance is returned to the initiating workstation  $p$ .

For the second performance criterion, i.e. the product demand,  $M$  products are considered and the average demand rate  $f_{sfj}^*$  for each product  $j \in M$ , c.f. (Sagawa2015a). The final outputs of the system, i.e. the flows  $f_{sfj}$  for  $j \in [1 : M]$ , after the last workstation of the product routing should meet the target demand rates  $f_{sfj}^*$ . These flows correspond to the output of the finished products leaving the manufacturing system. The controller of workstation  $p$  is able to adjust its output flow and therefore the succeeding flows, which are either converging or diverging to the successive workstations. As the workstations without connections to direct product outcomes ( $f_y^p = f_{sfj}$ ) cannot evaluate their local demand directly, based on those target flows, a different approach has to be followed: A *backward-forward propagation* based on the target flows allows to calculate the intermediate demand  $f_{sp}$  for each workstation  $p$  stated in Alg. 4, see for an example Fig. 5. At first, for a manufacturing system consisting of connected workstations and target demands (i.e. target flows), the maximum distance  $d_{max}$  (i.e. number of stages) between the workstations and the material provider (source) is evaluated, as the approach is examining the internal demand on each stage for all workstations, i.e. obtaining a set  $P_f$  of workstations with the same distance to the source, that is in the given example workstation 5 and product demand defined by  $f_{sf1}^*$  and  $f_{sf2}^*$  with  $f_{s5} = 10.000$  in Fig. 5. The variable  $d_{rec}$  stores the actual stage, which is under examination.

In Alg. 4, this demand is calculated in the first loop in lines 5 ff. Here, a set  $C_p$  is constructed for each workstation on the examined stage to incorporate the set of successive

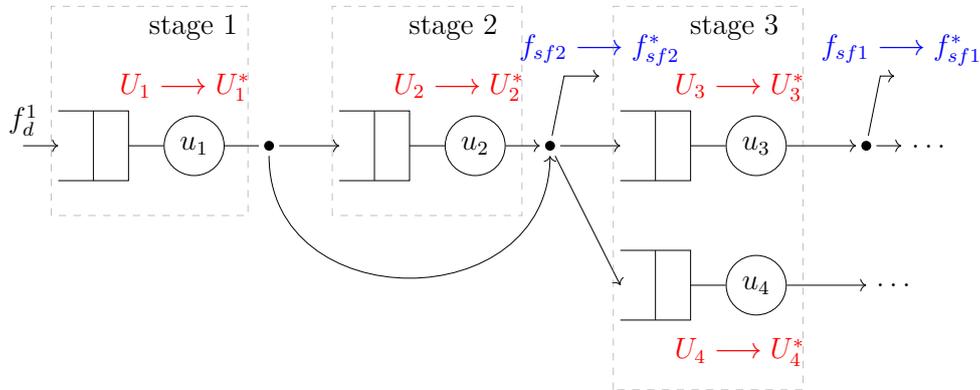
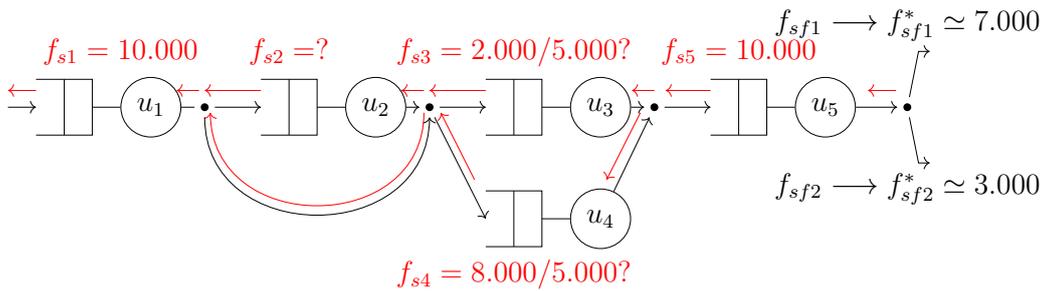
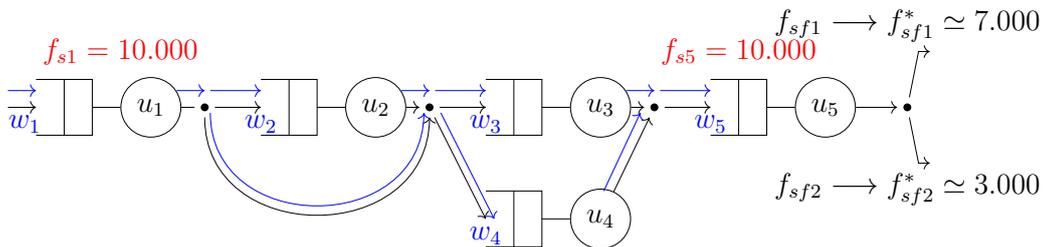


Figure 4: **Extract of the manufacturing system** consisting of four workstations in three stages (dashed boxes) with different product routings ( $f_{sf1}, f_{sf2}$ ) and control policies: tracking processing frequencies (red) or tracking product outcome (blue)



(a) **Backward Propagation:** At first, the demand is propagated backwards, allowing all combinations of distributions to the preceding workstations. In the algorithm, the equal distribution is chosen.



(b) **Forward Propagation:** The correct weights ( $w_i, i = 1, \dots, 5$ ) are applied to match the correct percentage of workload.

Figure 5: **Illustrated backward (red) and forward (blue) propagation example** with two product routings ( $f_{sf1}, f_{sf2}$ ).

**Algorithm 4** Backward propagation of the demand**Require:** target flows  $f_{sfj}^*$  for  $j \in [1 : M]$  and  $P$  workstations

---

```

1: Evaluate  $d_{\max}$  based on Alg. 3
2: Set  $d_{rec} := -1$ 
3: for  $d = d_{\max}$  to 0 do
4:   Obtain  $P_f := P_f \cup p$ , where  $d_p = d$  with Alg. 3
5:   for all  $p \in P_f$  do
6:     Set  $C_p := |\bar{K}_p \cup \bar{M}_p|$ 
7:     Calculate  $P_{p,succ}$  by Alg. 1
8:     Set  $f_{sp} = f_{sp} + \frac{1}{K_p} \sum_{j \in C_p} f_{sfj}^*$ 
9:     Set  $f_{sp} = f_{sp} + \frac{1}{K_p} \sum_{j \in P_{p,succ}} f_{sj}$ 
10:   end for
11:   Set  $d_{cur} := 0$ 
12:   for all  $p \in P_f$  do
13:     Calculate  $P_{p,pred}$  via Alg. 2
14:     for all  $j \in P_{p,pred}$  do
15:       if  $\bar{M}_j \neq M_p$  then
16:         Set  $d_{cur} := 1$ 
17:         if  $d_{rec} = -1$  then
18:           Set  $d_{rec} := d$ 
19:         end if
20:       end if
21:     end for
22:   end for
23:   if  $d_{cur} = 0$  then
24:     if  $d_{rec} \neq -1$  then
25:       Set  $d_{rec}$  by calling Alg. 5 with  $P_f$  and  $d_{rec}$ 
26:     else
27:       Set  $f_{sp} = w_p \sum_{j \in P_f} f_{sj}$ 
28:     end if
29:   end if
30: end for

```

---

target demands due to the target flows  $f_{sfj}^*$  or due to the demand of successive workstations. In the first iteration, this will exclusively be the target flows, defined by the product demands. Here, the coefficient  $\frac{1}{K_p}$  is imposed to split up the demand in a first step equally to avoid adding up the demand multiple times by mistake, but it ignores at first the percentages of the routing matrix, as a determination is not possible on this stage. In the example in Fig. 5a, the product demand for workstation 2 can only be determined if the bypass of the material flow of workstation 1 is considered, which will be done on the stage where workstation 2 is placed.

Then, if a set of diverging flows following a preceding workstation differs from the diverging flows before the current workstation (here,  $M_2 \neq \bar{M}_3$ ) (Alg. 4, lines 15), the affected stage is stored in  $d_{rec}$  to define the necessary stage for the forward propagation of the correct demand in a later step. If for the next iteration, i.e. the previous stage, the set of diverging flows differs again, which is evaluated in each iteration (line 14), the forward propagation (line 15) is prevented to avoid invalid propagation values. Now, with each following previous stage, if a forward propagation is necessary due to  $d_{rec} \neq 1$  and the current stage is consistent in terms of identical sets of diverging flows, which is ensured by  $d_{cur} = 1$ , the forward propagation is able to start by calling in line 25 Alg. 5.

Therein, if all diverging flows  $\bar{M}_j$  following the preceding workstations fits to the the set of diverging flows  $M_p$  before the current examined workstation  $p$ , which is the stage of workstation 2 in the illustrated example,  $d_{eq}$  is set to 1 and the forward propagation is executed. In practical terms, the origin of the diverged material flow is joined here, this flow will be provided by workstation 1, where the subsequent set of diverging flows  $\bar{M}_1$  equals to  $M_2$ , the diverging set of flows before workstation 2. Then, if this condition is met (Alg. 5, line 10), the forward propagation is executed up to the stage  $d_{rec}$  (from the stage of workstation 2 to the stage of workstations 3 and 4, c.f. Fig. 5b).

To avoid again a double summation by mistake, Alg. 6 is called recursively beforehand to recess these demand variables. Alg. 5 sets up the correct distribution according to the weights  $w_p$  according to each workstation  $p$ . If the distribution of demand is spread only among parallel workstations (by removing the bypass between workstation 1 and 3, 4, respectively), in Alg. 4 the simple case (line 27) is called, which allows immediately to distribute the summed, determined demand according to the defined weights  $w_p$  for each workstation  $p$ .

### 3.3 Optimal Control Problem

In this section, the OCP is introduced utilising two different objectives based on the actual processing frequencies of the workstations based on a state of pre-calculated processing frequencies as first performance criterion and on the other hand regarding the demand aiming for a product mix using backward and forward propagation as second performance criterion. Both objective functions are stated by using the L1-norm on the one hand and the quadratic form on the other hand as this allows to evaluate the performance of the overall system, when an equilibrium is reached. Additionally, the quadratic form is used in many formulations as a standard approach, as with the positive-definite form of the objective function convexity is ensured, which allows to most optimisers to achieve good results.

The OCP is solved by a centralised MPC scheme, obeyed by each workstation. It means that a central solution with availability of full information of the workstations (capacity, pro-

---

**Algorithm 5** Forward propagation

---

**Require:** set of workstations  $P_f$ ,  $d_{rec}$ 

```

1: Set  $d_{eq} := 1$ 
2: for all  $p \in P_f$  do
3:   Calculate  $P_{p,pred}$  via Alg. 2
4:   for all  $j \in P_{p,pred}$  do
5:     if  $\bar{M}_j \neq M_p$  then
6:       Set  $d_{eq} := 0$ 
7:     end if
8:   end for
9: end for
10: if  $d_{eq} = 1$  then
11:   Obtain  $P_{acc} := \bigcup_{j \in P_{p,pred}} j, \forall p \in P_f$ 
12:   Set  $f_{acc} := \bigcup_{k \in P_{p,succ}} f_{sk}, \forall p \in P_{acc}$ 
13:   Call Alg. 6
14:   Call Alg. 7 for each workstation  $p \in P_f$  with  $d_{rec}, f_{acc}$ 
15:   Set  $d_{rec} := -1$ 
16: end if
17: return  $d_{rec}$ 

```

---



---

**Algorithm 6** Delete temporary evaluated demand recursively

---

**Require:** workstation  $p$  and recursive distance  $d_{rec}$ 

```

1: Set  $f_{sp} := 0$ 
2: if  $d_p \leq d_{rec} \wedge P_{p,succ} \neq \emptyset$  then
3:   for all  $j \in P_{p,succ}$  do
4:     Call Alg. 6 with  $j$  and  $d_{rec}$ 
5:   end for
6: end if

```

---



---

**Algorithm 7** Forward propagation of the demand

---

**Require:** workstation  $p$ , recursive distance  $d_{rec}$  and cumulated flow  $f_{acc}$ 

```

1: for all  $p \in P_f$  do
2:   Set  $f_{sp} := w_p f_{acc}$ 
3:   if  $d_p \leq d_{rec} \wedge P_{p,succ} \neq \emptyset$  then
4:     for all  $j \in P_{p,succ}$  do
5:       Call Alg. 7 with  $j, d_{rec}$  and  $f_{acc}$ 
6:     end for
7:   end if
8: end for

```

---

cessing frequencies and material flow) is calculated by solving an OCP, i.e. by minimising an objective function subjected to material flow constraints. This is motivated by the necessity that, if the target frequencies are tracked, the workstations are driven to a target frequency in order to ensure that subsequent workstations are provided with sufficient material flows. If the product demand outcomes are tracked, a given product mix should be obeyed.

To fulfil both targets, the stage costs (current costs of the overall system) are defined in two ways: First, as a frequency-based minimising cost function and, secondly, as a demand-based cost function based on an *a-priori* defined product mix. These two approaches aim to attend similar goals (i.e. fulfilling the demand in terms of volume and mix), see for an example Fig. 4: The first issue is to match a fixed setpoint, which was calculated beforehand, see (Sagawa2015a), in order to attend the average medium-term demand for all product families. The second approach allows the optimiser more degrees of freedom, c.f. (Engell2007) to choose an optimal frequency instead of maintaining a fixed frequency for all workstations.

Here, a L1-norm cost function and quadratic cost function are utilised, motivated by Zheng1993, as the usage of the L1-norm may minimise the worst-case error of the sum, which may lead to a more robust behaviour. The stage costs based on the processing frequency are defined utilising the L1-norm with

$$\ell_{u_p}(q_p, u_p) = |u_p - u_p^*| + \lambda \Delta |u_p| + \mu q_p, \quad (12)$$

where  $\Delta |u_p|$  is the absolute difference of the frequency for the current and previous time instants, and  $0 < \mu, \lambda \leq 1$ , and  $q_p$  is the level of the buffer preceding workstation  $p$ , with  $\mu, \lambda$  to penalise high buffer levels and huge changes for the control values. Regarding a *quadratic* cost function, which may drive the systems faster to their reference states,

$$\ell_{u_p}(q_p, u_p) = \frac{1}{2} (u_p - u_p^*)^2 + \frac{1}{2} \lambda (\Delta u_p)^2 + \frac{1}{2} \mu q_p^2 \quad (13)$$

is given as this is suitable for many optimisation algorithms due to twice-differentiability.

Using the backward and forward propagation scheme from the previous Section 3.2 we are able to calculate the intermediate internal demand  $f_{sp}$  for each workstation. Therefore, the stage costs function based on the product demands utilising the L1-Norm is defined as

$$\ell_{f_p}(q_p, \overline{M}_p, u_p) = \sum_{j \in \overline{M}_p}^{| \overline{M}_p |} |f_{sfj} - f_{sfj}^*| + |f_y^p - f_{sp}| + \lambda \Delta |u_p| + \mu q_p \quad (14)$$

with  $0 < \lambda, \mu \leq 1$  and  $\Delta |u_p|$  as the absolute difference of the working frequency of the current and previous time instants.  $| \overline{M}_p | \leq M$  depicts the set of target flows that are direct successors of  $p$ , i.e. they are provided by the output  $f_y^p$ . Also for the demand based cost function, a quadratic version similar to the previous one based on reference processing frequencies is formulated:

$$\ell_{f_p}(q_p, \overline{M}_p, u_p) = \frac{1}{2} \left( \sum_{j \in \overline{M}_p}^{| \overline{M}_p |} (f_{sfj} - f_{sfj}^*)^2 + (f_{sp} - f_y^p)^2 + \lambda \Delta (u_p)^2 + \mu q_p^2 \right). \quad (15)$$

Considering the distances between the current workstation and the measured final outputs, a delay has to be taken into account between the imposed control input and measured output

due to the necessary time until the product outcome is changed: Increasing the processing frequency of a workstation needs at minimum the delay by the number of stages between this workstation and the measured product outcome. Besides that, an increase of the overall material flow, if necessary, is only imposed at the source. This delay between source and current workstation is then incorporated and therefore extend the costs function (14) to

$$\ell_{f_p}(q_p(n), \bar{M}_p, u_p(n)) = \sum_{j \in \bar{M}_p}^{| \bar{M}_p |} |f_{sfj}(n + d_j) - f_{sfj}^*| + |f_y^p(n) - f_{sp}(n + d_p)| + \lambda \Delta |u_p(n)| + \mu q_p(n) \quad (16)$$

where  $d_j$  defines the number of stages between source and the final output  $f_{sfj}$  corresponding to a finished product and  $d_p$  the number of stages between source and current workstation  $p$ .

Therefore, the prediction horizon has to be at least  $N > \max_{j \in M} \{|f_y^1 - f_{sfj}|\}$  to take into account the number of stages between the final output and the source output  $f_y^1$ . Adapting this for the quadratic cost function (15), this leads to

$$\ell_{f_p}(q_p(n), \bar{M}_p, u_p(n)) = \frac{1}{2} \left( \sum_{j \in \bar{M}_p}^{| \bar{M}_p |} (f_{sfj}(n + d_j) - f_{sfj}^*)^2 + (f_y^p(n) - f_{sp}(n + d_p))^2 \right) + \lambda \Delta (u_p(n))^2 + \mu q_p(n)^2. \quad (17)$$

Considering (7), which implicitly includes the condition of the existence of the predecessor material flow, the centralised OCP based on the cost function (12) is composed over the prediction horizon  $N$ :

$$\underset{\mathbf{u}^*}{\operatorname{argmin}} J^N(\mathbf{u}; q^0) := \sum_{p=1}^P \sum_{k=0}^{N-1} \ell_{u_p}(q_p(k), u_p(k)) \quad (18)$$

subject to the constraints

$$f_d^p(k) = \sum_{j=1}^{|K_p|} w_p v_j f_{yj}(k), j \in [1 : K_p], \quad k \in [0 : N] \quad (19)$$

$$g_p^k := -q_p(k) \leq 0, \quad k \in [0 : N] \quad (20)$$

$$q_p(k) = q_p(0) + \sum_{k=0}^{N-1} [f_d^p(k) - u_p(k) \min(1, q_p(k))], \quad k \in [0 : N - 1] \quad (21)$$

$$f_{yp}(k + 1) = u_p(k) \min(1, q_p(k)), \quad k \in [0 : N - 1] \quad (22)$$

$$0 \leq u_p(k) \leq \bar{u}_p, \quad k \in [0 : N - 1] \quad (23)$$

$$f_{yp}(k + 1) = \sum_{j \in P_{succ}}^{|P_{succ}|} w_j v_p f_{dj}(k + 1), \quad k \in [0 : N - 1] \quad (24)$$

where  $q^0 := \{q_1^0 := q_1(n), \dots, q_p^0 := q_p(n)\}$ . The constraints refer to: the input flow of a given workstation  $p$  coming from the predecessor workstations (19) based on (10), the non-negativity of the stock (20), the buffer level (21) of a given workstation based on (7), the

output flow of the workstation resulting from the processing frequency (22), lower and upper bounds for the processing frequencies (23), and the output flow of a given workstation  $p$  distributed to the successive workstations, guaranteeing that the output flow matches the input flows of the successive workstations (24). The optimal control sequence is then defined as

$$\mathbf{u}^* := \left\{ \underbrace{[\lceil u_1^*(0) \rceil, \dots, \lceil u_1^*(N-1) \rceil]}_{\mathbf{u}_1^*}, \underbrace{[u_2^*(0), \dots, \lceil u_2^*(N-1) \rceil]}_{\mathbf{u}_2^*}, \dots, \underbrace{[u_P^*(0), \dots, \lceil u_P^*(N-1) \rceil]}_{\mathbf{u}_P^*} \right\}.$$

Applying the product-demand based cost function (14), the cost function for the OCP is then calculated and assembled as

$$\operatorname{argmin}_{\mathbf{u}^*} J^N(\mathbf{u}, \bar{M}; q^0) := \sum_{p=1}^P \sum_{k=0}^{N-1} \ell_{f_p}(q_p(k), \bar{M}_p, u_p(k)) \quad (25)$$

subject to (19)-(24) and  $M := \{M_1, \dots, M_P\}$ . The corresponding value function (open-loop costs) for the OCP (18) is then obtained as

$$V^N(q^0) = J^N(\mathbf{u}^*; q^0) \quad (26)$$

and for the demand driven OCP (25) as

$$V^N(\bar{M}_p; q^0) = J^N(\mathbf{u}^*, \bar{M}; q^0). \quad (27)$$

## 4 Numerical simulations

The simulations were conducted in C++ by using the optimisation package NLOpt (**Nlopt**). As the cost functions allow gradient-based (SLSQP, **Kraft1994**) optimisation algorithms with support of boundary and inequality constraints, different simulations were carried out to examine the properties and the behaviour of the centralised control for both cost functions, i.e. frequency-based and demand-based. In addition, a gradient-free algorithm was also evaluated, namely the COBYLA based on linear approximation (**Powell1998**).

To compare the performance, the following indicators were considered:

- open-loop costs
  - regarding target frequencies (26), minimising (18) utilising the L1-norm (12) or quadratic stage costs (13), or
  - regarding the product demand mix (27), minimising (25), utilising the L1-norm (16) or quadratic stage costs (17);
- process frequency of workstation  $p$ :  $u_p(n)$ ;
- buffer level of workstation  $p$ :  $q_p(n)$ ;

with  $\lambda = 0.2$  and  $\mu = 0.05$  to penalise high differences in frequencies and high buffer levels, respectively.

In particular, the gradient-free algorithm does not offer advantages in terms of speed, convergence or computational time. But nevertheless, if e.g. non-differentiable constraints might be used (on/off behaviour of workstations), COBYLA could be still applied. Furthermore, other cost functions might be employed based on the infinity norm, which demands a gradient-free algorithm to be utilised. The effectiveness of both algorithms is shown, when the equilibrium, a steady state, could be achieved with either keeping the frequencies of the workstations or fulfilling the given demand.

As the prediction horizon length for the MPC scheme influences the speed of convergence to reach a steady state, and hence, the quality of solution, the performance was evaluated by conducting the simulations with  $N = 8$  and  $N = 20$  for frequency-based cost functions and  $N = 8$  and  $N = 30$  for demand-based cost functions. We chose these candidates as their results provide enough difference to demonstrate the influence of the prediction horizon length. As a longer horizon had only a small impact on the objective considering the processing frequencies, the horizon is kept to  $N = 20$ .

On the other hand, for the objective function considering the product demands, the horizon length has to be carefully chosen: As the system tends to ill-conditioning, which might occur due to different processing frequency configurations of parallel workstations, the horizon length has to be a multiple of the number of stages (cf. **Rossiter2003**) due to the delay between measurement and control input. Hence, the horizon length was set to  $N = 30$ , as the different product routings use between two and six stages.

#### 4.1 Frequency-based

The OCP (18) tracking the processing frequencies was evaluated for gradient-based and gradient-free optimisers, utilising L1-stage costs (12) or quadratic stage costs (13).

##### 4.1.1 Gradient-based optimisation

Here, a modified version, the Sequential Least-Squares QP (SLSQP) was used, which transform the OCP into a two-point boundary value problem via piecewise interpolation for cost function, state and control constraints. The approach uses a discretisation, which was already provided by the formulation of the OCP (18), and partial derivatives of the objectives to convert the OCP into a common non-linear program (NLP). While in a previous version (**Sprodowski2018**) a lower accuracy of tolerance was used ( $\delta f = 0.0001$  instead of  $\delta f = 0.00001$ ) and an older version of NLopt was used (version 2.4.0 instead of 2.5.0), improved results were retrieved with the latest version and also a higher accuracy of the optimal value. The results are shown in Fig. 6a.

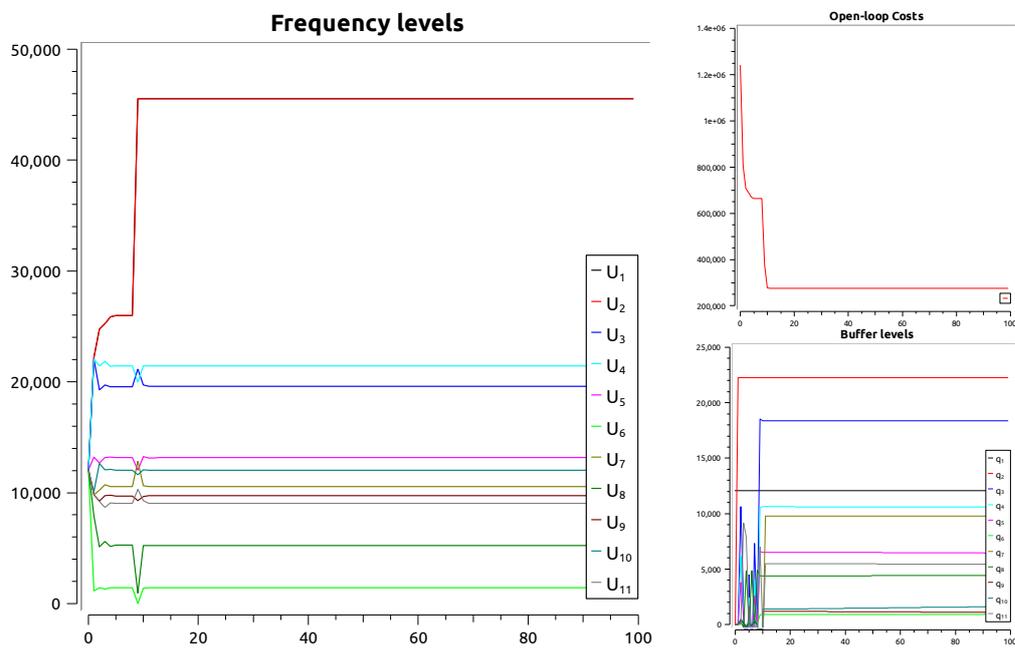
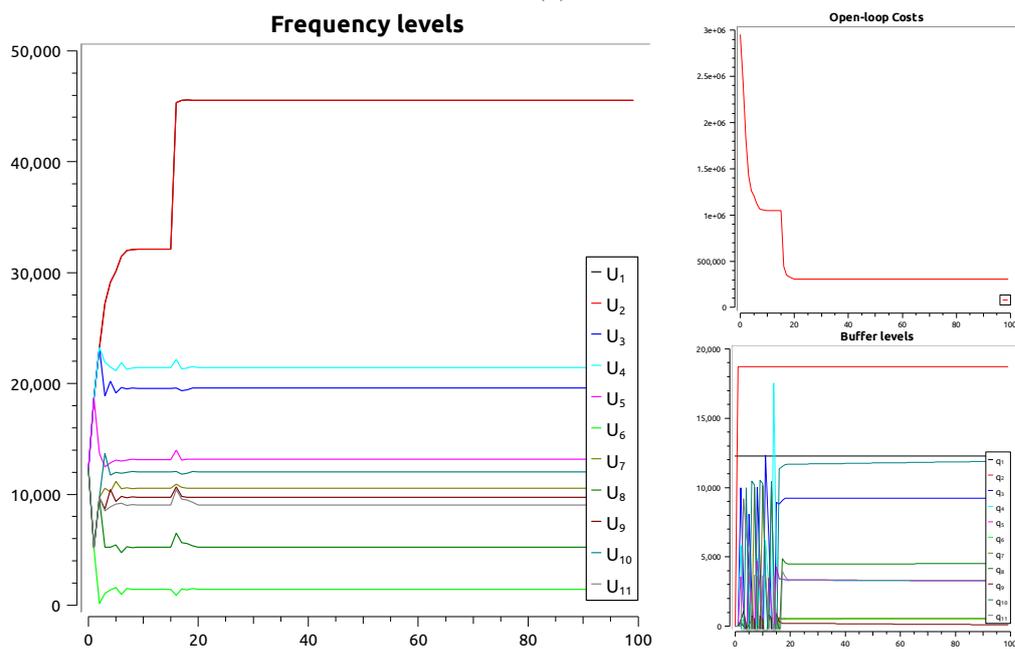
(a)  $N = 8$ (b)  $N = 20$ 

Figure 6: Frequency levels  $u_{1-11}$  (left), Buffer levels  $q_{1-11}$  (right bottom) and Open-loop costs (26) (right top), tracking processing frequencies (18) using SLSQP and L1-norm as stage costs (12)

After a short initialisation phase the frequencies were stable around the aimed steady-

state frequencies. The open-loop costs were reduced quickly, while a small decrease in the frequency curves occurred due to material supply shortage. For the workstations in later stages, the delay of material supply induced a short slowdown of the processing frequencies (see workstation 11 in Fig. 6a (left)), due to the buffer level stabilisation the interruption was only on short term. For a comparison to a longer horizon, the results for  $N = 20$  are illustrated in Fig. 6b.

As the optimiser was equipped with a longer horizon, the cost functions decreased slower than for  $N = 8$  (cf. Fig. 6b and 6a, right top). Also the steady-state frequencies were reached more slowly and induced a longer initialisation phase in comparison to the shorter horizon shown beforehand. An advantage in this setup is the reduced utilisation of the buffer due to a slower asymptotic approximation of the processing frequencies of the initial workstations, which holds especially for the stock of workstation 2.

Applying the SLSQP optimisation method with the quadratic cost function equipped with a short horizon  $N = 8$  led to a fast approximation to the reference processing frequencies (Fig. 7a) and faster stabilisation of the buffer levels, but a smoother reduction of the open-loop costs without plateaus (Fig. 7a (right top)).

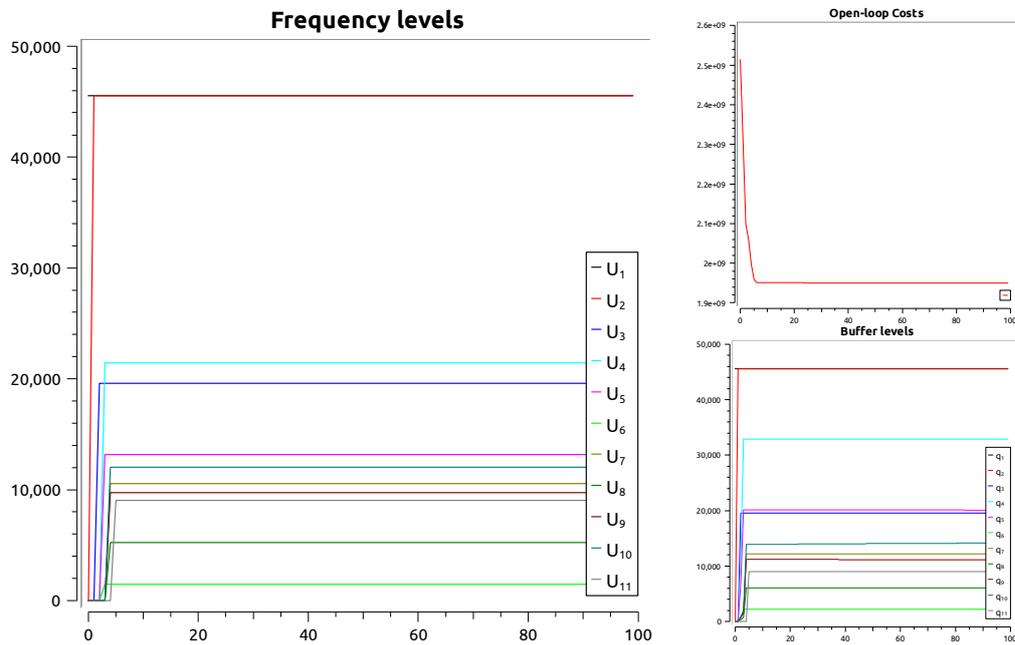
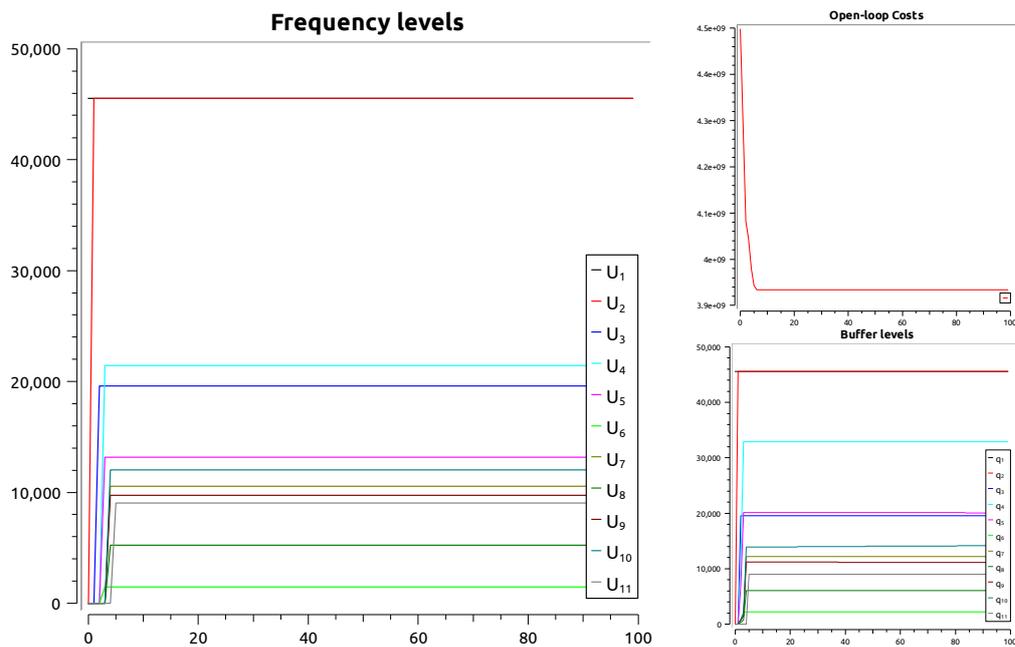
(a)  $N = 8$ (b)  $N = 20$ 

Figure 7: Frequency levels  $u_{1-11}$  (left), Buffer levels  $q_{1-11}$  (right bottom), and Open-loop costs (26) (right top), tracking processing frequencies (18) using SLSQP and quadratic stages costs (13)

For the quadratic costs using the SLSQP-method, results for the longer prediction horizon

are illustrated in Fig. 7b.

Considering different horizon lengths, this seems to be a negligible parameter considering the quadratic cost function in opposite to the L1-norm beforehand. With both horizon lengths, a fast convergence was achieved. Comparing the results to the semi-quadratic function (Eq. 10), used in **Sprodowski2018**, shows that the overshoot did not occur with the “full” quadratic function, which was used here.

#### 4.1.2 Gradient-free algorithms

Considering gradient-free algorithms, COBYLA was used, which approximates the problem linearly with respect to the given constraints. The results, obtained for the L1-norm cost function (12) with horizon length  $N = 8$ , are shown in Fig. 8a.

It is important to note that the runtime is significantly larger than with gradient-based algorithms (30-50 sec vs. 2-5 sec. per iteration), which should be taken into account if real-time requirements were considered. As the frequency levels were reached without any disturbance or overshooting, the cost function decreased monotonously. As the frequency values were slightly below the steady-state frequencies, for workstations in later stages, the buffer levels were slightly reduced.

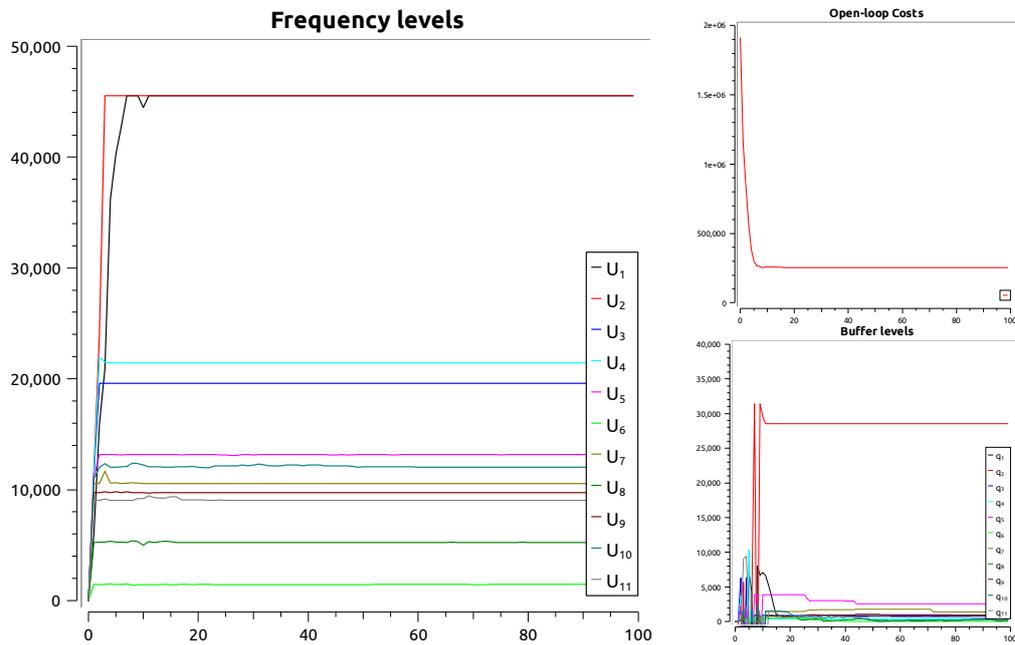
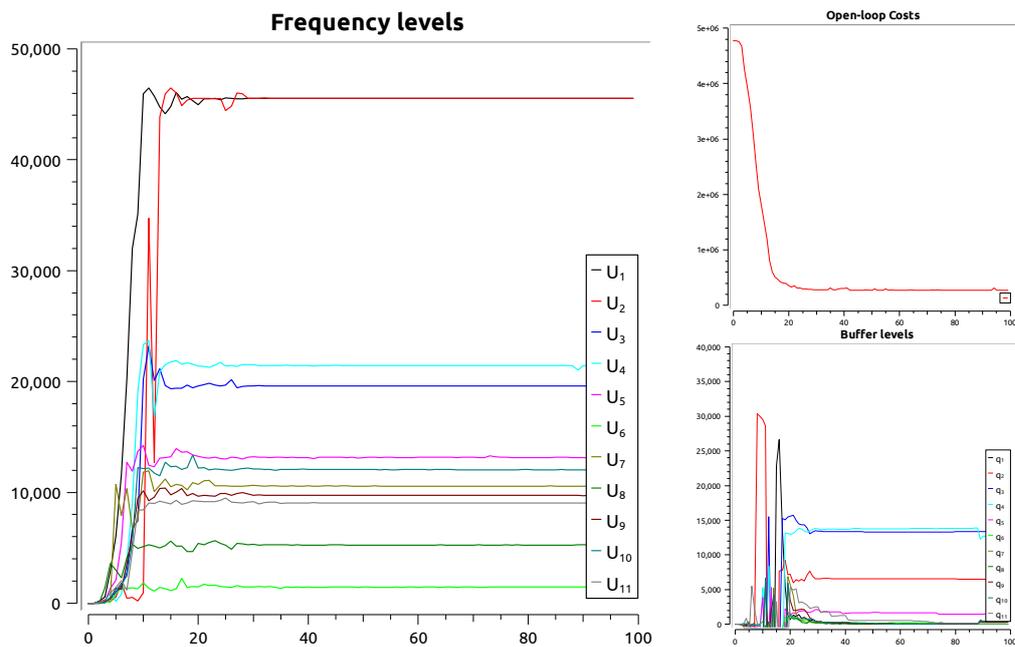
(a)  $N = 8$ (b)  $N = 20$ 

Figure 8: Frequency levels  $u_{1-11}$  (left), Buffer levels  $q_{1-11}$  (right bottom), and Open-loop costs (26) (right top), tracking processing frequencies (18) using COBYLA and L1-norm stage costs (12)

Also for a longer prediction horizon, with  $N = 20$ , the COBYLA algorithm was applied

with L1-norm stage costs, and the results are depicted in Fig. 8b.

For the quadratic costs, the results for  $N = 8$  are illustrated in Fig. 9a.

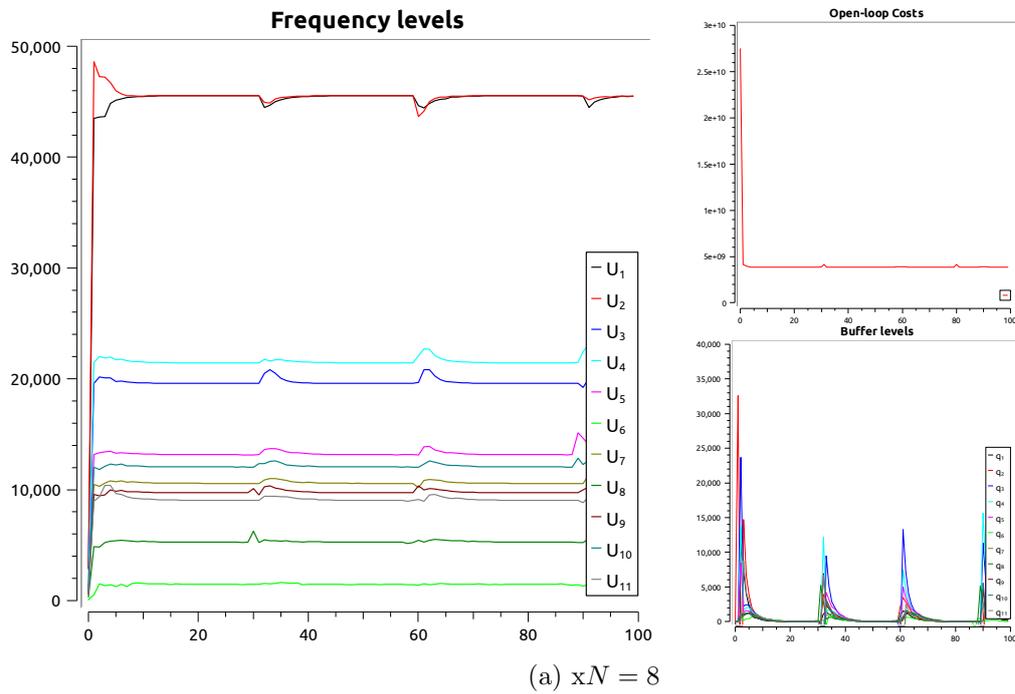
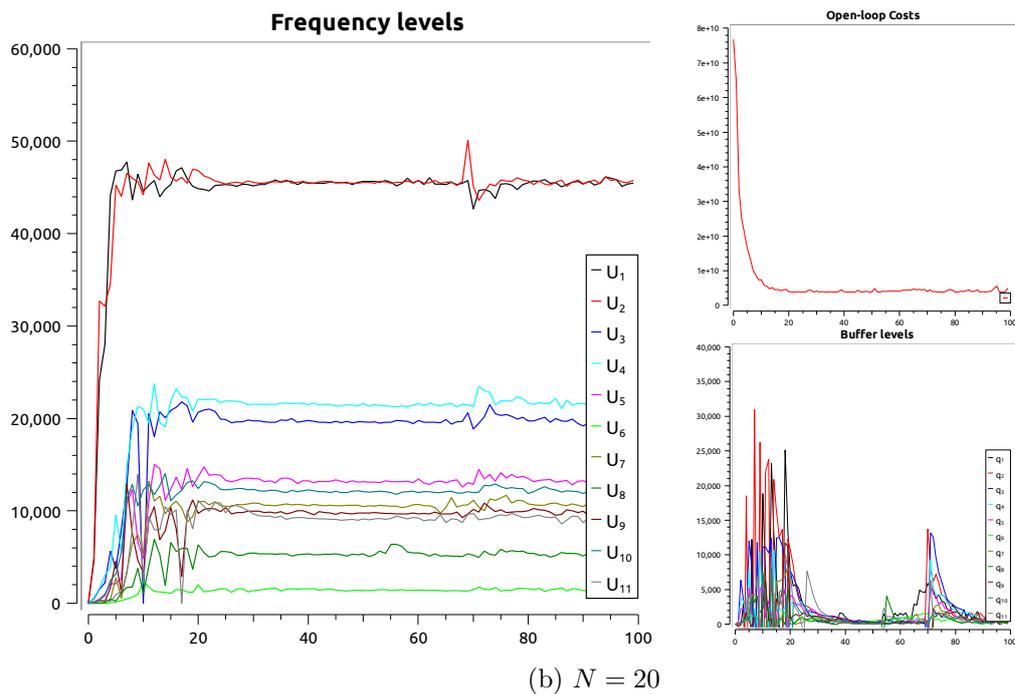
(a)  $xN = 8$ (b)  $N = 20$ 

Figure 9: Frequency levels  $u_{1-11}$  (left), Buffer levels  $q_{1-11}$  (right bottom), and Open-loop costs (26) (right top), tracking processing frequencies (18) using COBYLA and quadratic stage costs (13)

The steady-state frequencies were achieved faster than for the gradient-based SLSQP

optimiser with the quadratic cost function. While the buffer levels were reduced, the workstations were running short in material supply, which led to a periodic behaviour of the frequencies, cf. Fig. 9a (left). Hence, as the solution diverges from the steady state and the workstations have to decrease the frequencies, the open-loop costs (Fig. 9a (right top)) are also rising slightly. For a longer horizon, results are shown in Fig. 9b.

Considering a longer horizon with quadratic stage costs, the periodic material shortage decreasing the processing frequencies occurs randomly, while for the shorter horizon this happens periodically. But for both horizon lengths, oscillations of the workstation frequencies occurred over the full simulation period.

#### 4.2 Demand-based

The OCP (25) was equipped with a quadratic cost function (15) incorporating the time-delay (17). The demand-based version has two limitations: First, the SLSQP-approach was able to provide stable solutions only. Second, using the L1-norm cost function (14), the optimiser was unable to converge to a steady state, which led to oscillating frequencies and buffer levels. To incorporate the delay induced by the material flow via the stages and to attenuate the maximum distance between the first workstation providing the material flow and the resulting output concerning the product mix, a longer prediction horizon had to be used.

For the L1-norm, the results are presented using a long prediction horizon ( $N = 30$ ) in Fig. 10.

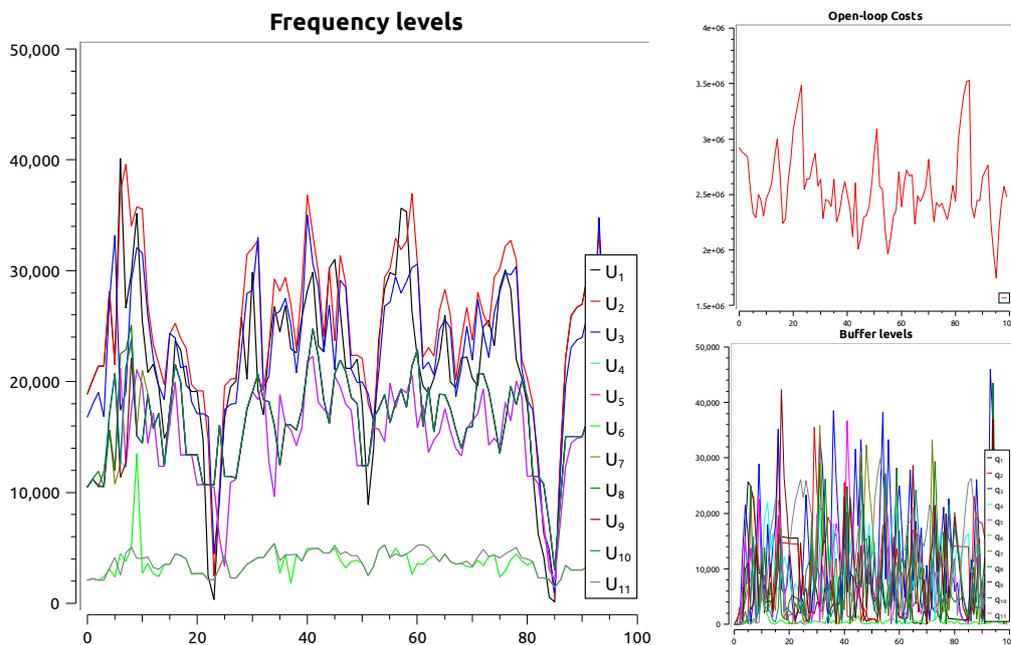
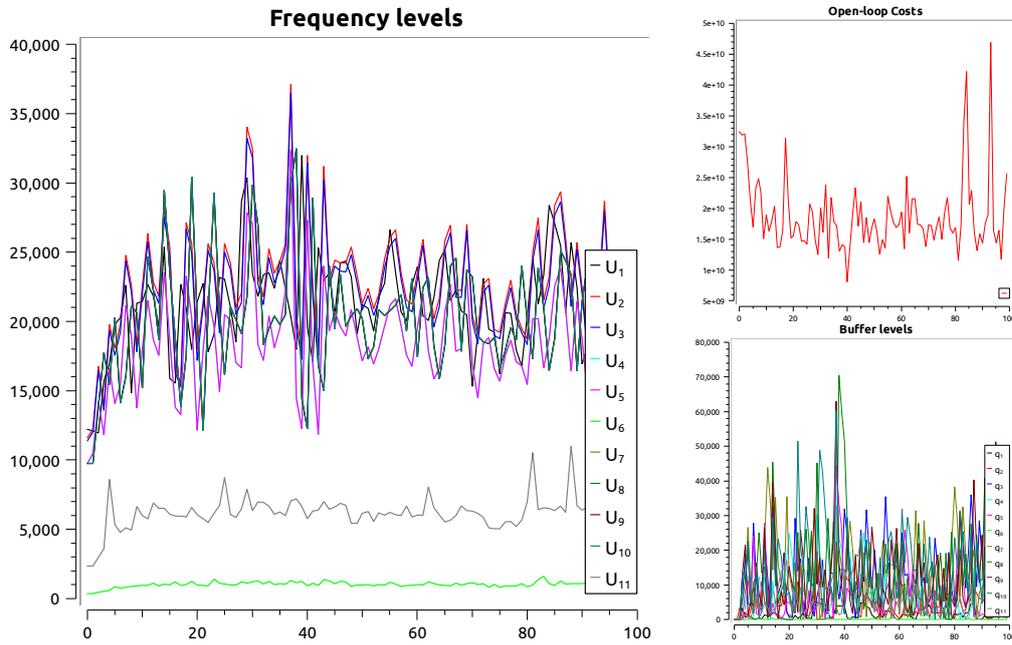
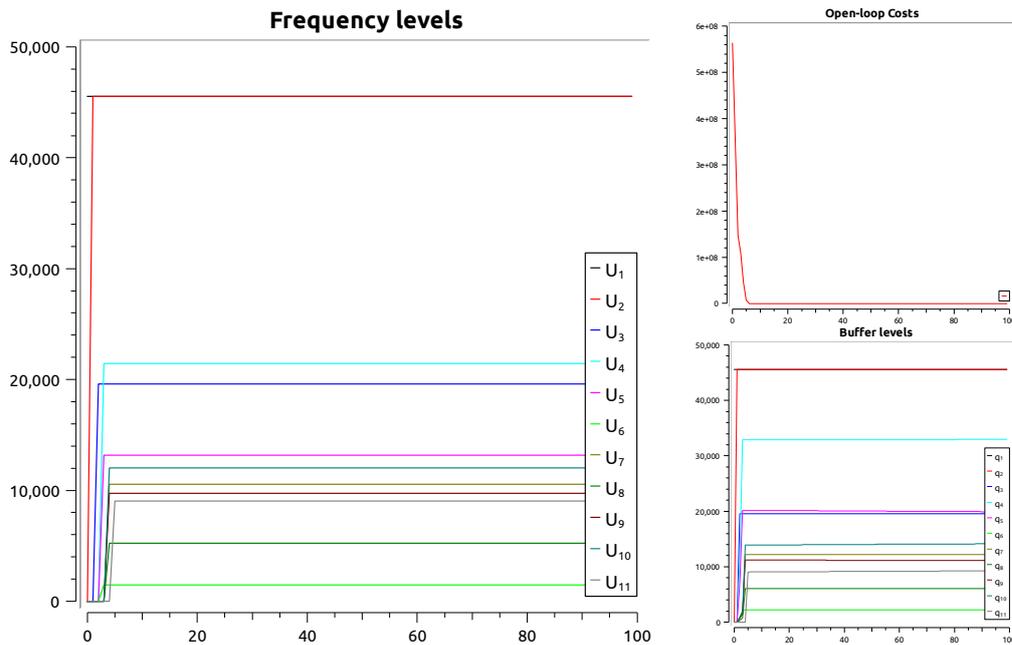


Figure 10: Frequency levels  $u_{1-11}$  (left), Buffer levels  $q_{1-11}$  (right bottom), and Open-loop costs (27) (right top), tracking product mix demand (25) using SLSQP and L1-norm stage costs (16) with  $N = 30$

Both buffer levels and frequencies of the workstations oscillated and led to unsatisfying results. For the quadratic cost function using  $N = 12$ , see Fig. 11a, and for  $N = 30$  see Fig. 11b.



(a)  $N = 12$



(b)  $N = 30$

Figure 11: Frequency levels  $u_{1-11}$  (left), Buffer levels  $q_{1-11}$  (right top), and Open-loop costs (27) (right top), tracking product mix demand (25) using SLSQP and quadratic stage costs (15)

For a short horizon (see Fig 11a), neither the frequencies of the workstations reached a

steady state, nor the open-loop costs decreased monotonously. A reason for this oscillating behaviour was the entangled material flow on different levels, which cannot be handled by a too short prediction horizon as the optimiser cannot monitor the imposed control inputs together with the resulting product outcomes at the end of the manufacturing process. Beyond that, even a long prediction horizon is not sufficient, as this is not equal to a multiple of the number of stages, i.e. an unaligned prediction horizon length. This also led to an ill-conditioning process, as many configurations for the workstations resulted to the same function values of the cost function. Finally, much better results were achieved with a longer horizon  $N = 30$ , which is presented in Fig. 11b, as this was the lowest common multiple concerning the number of stages.

Analogously to the frequency-based cost functions (12) and (13), the steady-state was reached quickly. Looking at the number of stages associated to each product flow, according to **Rossiter2003** the prediction horizon has to be matched by a multiple of the number of stages to reduce the probability of ill-conditioning. Furthermore, an appropriate prediction horizon aligned the imposed control inputs to the measured product outcomes, which created an enclosed model state for the optimiser to evaluate the influence of various control inputs on the product outcomes over the full prediction horizon via the cost function.

### 4.3 Discussion of the simulations

In these simulations, results were presented concerning an objective aiming at steady-state working frequencies and an objective concerning fulfilling product demands. Regarding steady state working frequencies for the workstations, the SLSQP-approach either with a short or long prediction horizon was able to keep a certain buffer level and constant working frequencies (e.g. Figs. 6a and 6b). The curves of processing frequencies and buffer levels were smoother for the quadratic norm than for the L1-norm, in the sense that the curves for the L1-norm tended to present higher variability (high frequency oscillations, at the beginning of operation, for instance) and steps. Overall, the gradient-based optimisation led to better performance concerning a less oscillating buffer level and constant working frequencies.

The COBYLA algorithm was also able to keep the frequencies for the L1-norm cost function, while for the quadratic cost function the buffer levels were lower but led to a supply shortage. An overview representing the performance criteria is summarised in Table 1. For the product-demand based function, the SLSQP optimisation was able to find appropriate solutions by selecting an appropriate prediction horizon, i.e. a multiple of the number of stages.

A classical trade-off in managerial terms is revealed by the comparison of Figs. 6a and 7a: with the L1-norm, the reference processing frequencies were achieved with lower levels of WIP (maximum around 22,000 on Fig. 6a), but with high WIP variability at the beginning of the system's operation, while for the quadratic norm, the WIP levels are higher (maximum around 45,000 on Fig. 7a) but the variability is low/nonexistent. In fact, the adjustment of the frequencies based on the L1-norm may have led to the starvation of some stations, when the levels of some buffers became null. The deleterious effects of high input variability to the stations are discussed by **Hopp2011**. Thus, in some cases, it is relevant to have smooth operations at the expense of lower buffer levels, as in the case depicted in Fig. 7a.

One incessant goal of production managers is to smoothen operations, so that the opera-

	Method	N	Target	Frequencies	Buffer levels	Buffer variab.
1	SLSQP (F)	8	L1	achieved	moderate	intermediate
2	SLSQP (F)	20	L1	achieved	lower	high
3	SLSQP (F)	8	Q	achieved	high	low
4	SLSQP (F)	20	Q	achieved	high	low
5	COBYLA (F)	8	L1	achieved	moderate	high
6	COBYLA (F)	20	L1	achieved	moderate	high
7	COBYLA (F)	8	Q	mostly achieved	low	high
8	COBYLA (F)	20	Q	not achievable	low	high
9	SLSQP (D)	12/30	L1	not achievable	high	high
10	SLSQP (D)	12	Q	not achievable	high	high
11	SLSQP (D)	30	Q	achievable	high	low

Table 1: Results for all methods tracking either frequency (F) or product demands (D) regarding the type of given objective (L1-norm or quadratic (Q))

tional costs stay low, although this pursuit should not unreasonably compromise the flexibility or speed to attend customers, depending of the business competitive priorities, as discussed by **Hayes2004**.

For the advantageous functioning of a system, there must be a critical level of WIP in it; with WIP below this level, i.e. temporary starvation, the system may not achieve its maximum throughput rate (**Hopp2011**). On the other hand, the aforementioned authors also argue that the most important feature of a pull system is to have a WIP cap based on endogenous information, which may be adopted here in the given case from Fig. 7a).

Considering the technical perspective, the gradient-based approach (COBYLA) undermined this critical level especially for the quadratic objective (see Fig. 9a). Here, the frequencies decreased periodically as the buffers were running empty. For the L1-norm, the critical WIP level was not violated as for both horizon lengths (see Fig. 8a and 8b) the frequencies stay constant. Nevertheless, the smoother achievement of the target frequencies with a longer horizon (see Fig. 8b) led to lower buffer levels according to the SLSQP approach (see Figs. 6a and 6b).

For the demand-based approach, it is possible to note that the processing frequencies matched to those, which were set as reference for the frequency-based approach. These latter references, on their turn, adopted from **Sagawa2015a** showing that the MPC model is consistent and that the forward and backward strategy of demand propagation is appropriate.

In other words, the comparison of both approaches functions as an internal validation of the proposed MPC model (including the frequency- and demand based versions). Furthermore, the demand-based approach provides more flexibility to support decisions: different demands for the product families can be included into the model, and the impact of these changes can be evaluated. The frequency-based approach, on the contrary, is not suitable to cope with volatile demands, since these parameters (demands) are not explicit in this version of the model. In addition, the demand-based MPC model overcomes one of the limitations of the original model presented in the literature (**Sagawa2015a**; **Sagawa2017**), in which

the demands are also not modelled as explicit parameters.

Last, the values of processing frequencies are not infinite applicable: in fact, these adjustments should be implemented by means of overtime, extra shifts, subcontracting, acquisition of new/extra resources or downtime of the machines. As known, there are technological constraints that prevent the direct or “physical” adjustment of the processing frequencies (or processing times) of the stations to any arbitrary values, that is, these times depend on the features of the product being manufactured, on the machine employed, among many other factors. Thus, if the processing frequencies should be temporary increased in 2% in relation to the reference frequencies, this means, for instance, that 2% of overtime should be implemented, over a given period. If the processing frequency is temporary below the reference, this means that machine downtimes should be introduced.

## 5 Conclusion

In this paper, an OCP for a real-case application was proposed based on a bond graph model. The motivation was to have a realistic view to the applicability of such schemes as many schemes or models are handling small or example scenarios as discussed in the literature review. The MPC scheme was applied with two different objectives tracking on the one hand constant working frequencies of the workstations and on the other hand tracking a given product demand mix. Considering operational goals concerning constant frequencies or an adequate WIP level, the results allow a reasonable decision support to assess the most appropriate control method to the chosen goal, which should be fulfilled.

For both objectives, the gradient-based cost functions were able to reach a given equilibrium for reference working frequencies, and on the other hand to fulfil a given product demand. The gradient-free targets tackled the working frequencies only with some restrictions considering a steady-state but lower buffer levels.

Further considerations should evaluate a distributed or decentralised control to evaluate a comparison to this centralised approach. Considering the failure of workstations, reference levels for the stock to ensure robustness with the availability of material should be addressed. Furthermore, the adjustment of penalising a change in the working frequency and the impact of the stock penalties may still be evaluated.

## Acknowledgments

The second and third author would like to thank the *São Paulo Research Foundation (FAPESP)* for the financial support for this research (grant number 2019/12023-1). The corresponding author would like to thank the University of Bremen, which supported via the programme *Bremen IDEA out* to conduct part of this research in Brazil.

This article is printed under terms of Creative Commons CC-BY-NC-ND  
<https://creativecommons.org/licenses/by-nc-nd/4.0/> with all rights ©Elsevier.  
Original publication is available via <https://doi.org/10.1016/j.eswa.2020.113734>.

# B

## Software

The software, especially simulations of the DMPC system, generation of graphical plots, which are implemented in C++/Qt, is hosted at <https://github.com/SirTobias/DMPCRobotSimulation>.