

Autonomous Operation of a Reconfigurable Multi-Robot System for Planetary Space Missions

by

Thomas Mirko Roehr

Dissertation
zur Erlangung des Grades eines
Doktors der Ingenieurwissenschaften
- Dr.-Ing. -

Vorgelegt im Fachbereich 3 (Mathematik & Informatik)
der Universität Bremen
März 2019

Datum des Promotionskolloquiums: 19. Juni 2019

Gutachter:

Prof. Dr. Dr. h.c. Frank Kirchner (Universität Bremen)

Prof. Dr. Joachim Hertzberg (Universität Osnabrück)

Abstract

Reconfigurable robots can physically merge and form new types of composite systems. This ability leads to additional degrees of freedom for robot operations especially when dynamically composed robotic systems offer capabilities that none of the individual systems have. Research in the area of reconfigurable multi-robot systems has mainly been focused on swarm-based robots and thereby to systems with a high degree of modularity but a heavily restricted set of capabilities. In contrast, this thesis deals with heterogeneous robot teams comprising individually capable robots which are also modular and reconfigurable. In particular, the autonomous application of such reconfigurable multi-robot systems to enhance robotic space exploration missions is investigated.

Exploiting the flexibility of a reconfigurable multi-robot system requires an appropriate system model and reasoner. Hence, this thesis introduces a special organisation model. This model accounts for the key characteristics of reconfigurable robots which are constrained by the availability and compatibility of hardware interfaces. A newly introduced mapping function between resource structures and functional properties permits to characterise dynamically created agent compositions. Since a combinatorial challenge lies in the identification of feasible and functionally suitable agents, this thesis further suggests bounding strategies to reason efficiently with composite robotic systems.

This thesis proposes a mission planning algorithm which permits to exploit the flexibility of reconfigurable multi-robot systems. The implemented planner builds upon the developed organisation model so that multi-robot missions can be specified by high-level functionality constraints which are resolved to suitable combinations of robots. Furthermore, the planner synchronises robot activities over time and characterises plans according to three objectives: efficacy, efficiency and safety. The planner's evaluation demonstrates an optimisation of an exemplary space mission.

This research is based on the parallel development of theoretical concepts and practical solutions while working with three reconfigurable multi-robot teams. The operation of a reconfigurable robotic team comes with practical constraints. Therefore, this thesis composes and evaluates an operational infrastructure which can serve as reference implementation. The identification and combination of applicable state-of-the-art technologies result in a distributed and dynamically extensible communication infrastructure which can maintain the properties of reconfigurable multi-robot systems.

Field tests covering semi-autonomous and autonomous operation have been performed to characterise multi-robot missions and validate the autonomous control approach for reconfigurable multi-robot systems. The practical evaluation identified critical constraints and design elements for a successful application of reconfigurable multi-robot systems. Furthermore, the experiments point to improvements for the organisation model.

This thesis is a holistic approach to automate reconfigurable multi-robot systems. It identifies theoretical as well as practical challenges and it suggests effective solutions which permit an exploitation of an increased level of flexibility in future robotics missions.

Zusammenfassung

Rekonfigurierbare Roboter verfügen über die Fähigkeit sich untereinander physikalisch zu koppeln und damit neuartige Roboter zu erzeugen. Von besonderem Interesse sind Koalitionen von Robotern, die mehr Fähigkeiten besitzen als die einzelnen Roboter. Rekonfiguration von Robotern wird oftmals im Kontext von schwarmbasierten Systemen untersucht, denen eine einfache und einheitliche Gestaltung der einzelnen Teilnehmer zugrunde liegt, so dass ein hoher Grad an Systemredundanz ausgenutzt werden kann. Im Gegensatz dazu beschäftigt sich diese Arbeit mit heterogenen Roboterteams, deren einzelne Akteure fähiger sind und sich aufgrund ihres modularen Aufbaus auch rekonfigurieren können. Speziell der autonome Einsatz dieser Systeme für planetare Erkundungsmissionen wird in dieser Arbeit angestrebt und untersucht.

Dazu wird ein Organisationsmodell eingeführt mit dessen Hilfe Roboter beschrieben werden können, deren Verbindungsfähigkeit durch die Verfügbarkeit und Kompatibilität von elektromechanischen Schnittstellen definiert ist. Damit werden die Möglichkeiten zur Formierung von unterschiedlichen Koalitionsstrukturen beschreibbar. Das Organisationsmodell bildet zwischen der Ressourcenstruktur und Funktionalität eines Roboters ab und erlaubt davon abhängig die Charakterisierung von dynamisch gebildeten Koalitionen. Die Identifikation von technisch möglichen und funktional geeigneten Koalitionen stellt ein kombinatorisches Suchproblem dar. Daher werden in dieser Arbeit Strategien zur Einschränkung des Suchraums vorgestellt, die eine effiziente Nutzung einer Vielzahl von rekonfigurierbaren Agenten in der Planung erlauben.

Das Organisationsmodell stellt die Grundlage für einen Missionsplanungsalgorithmus, der die Systemflexibilität von rekonfigurierbaren Multi-Robotersystemen ausnutzen kann. Eine Multi-Roboter Mission wird durch Anforderungen an Systemfunktionen beschrieben, die auf einsatzfähige Agenten bzw. Koalitionen abgebildet werden. Darüber hinaus synchronisiert der Planer Roboteraktivitäten und bewertet Pläne nach drei Kriterien: Effektivität, Effizienz, und Sicherheit. Der Planer wird für eine beispielhafte Weltraummission angewandt.

Diese Forschungsarbeit basiert auf der parallelen Entwicklung von theoretischen Konzepten und praktischen Lösungsansätzen. Die praktische Arbeit mit drei sukzessiv entwickelten rekonfigurierbaren Multi-Robotersystemen erlaubte dabei die Identifizierung von relevanten Einsatzbedingungen. Daraus resultiert die Entwicklung einer operationalen Infrastruktur, die existierende Technologien und Standards zum Aufbau einer verteilten und dynamisch erweiterbaren Kommunikationsinfrastruktur nutzt.

Feldtests erlaubten eine Validierung und Analyse der semi-autonomen und voll-autonomen Systemfähigkeiten. Kritische Voraussetzungen für den praktischen Einsatz konnten so ebenso identifiziert werden, wie Verbesserungspotentiale für das Organisationsmodell.

Insgesamt beschreibt diese Arbeit einen ganzheitlichen Ansatz zum autonomen Betrieb von rekonfigurierbaren Multi-Robotersystemen. Der Ansatz beinhaltet die theoretischen Modelle und Konzepte ebenso wie praktische Lösungen, um die operationale Flexibilität für zukünftige Robotermissionen zu erhöhen.

Acknowledgements

This thesis is the result of a journey through the field of reconfigurable multi-robot systems and has been a continuous learning process: learning about robots and technologies, going beyond limits, and about life in general. This journey would not have been possible without my supervisor Frank Kirchner, who established the Robotics Innovation Center in Bremen. My gratitude goes to him for giving me this unique opportunity to work with reconfigurable multi-robot systems and for supporting me in my research.

Operating multi-robot systems builds upon the continuous efforts over many years of many people developing, constructing, implementing, evaluating and improving hardware and software designs. Thanks to those colleagues and students who took part in the construction process of the robots that seeded the idea of this thesis and who implemented bits and pieces along the way to make the reconfigurable multi-robot systems operational. A particular thanks to the contributors of the Robot Construction Kit aka Rock as basis for the finally infield evaluated system. I am glad to share the experience of the field trip in Utah with Leif Christensen, Florian Cordes, Steffen Planthaber, Roland Sonsalla and Tobias Stark. My thanks also goes to the home crew members Sebastian Kasperski, Michael Maurus, Sankar Natarajan, and Roman Szuka. Thanks to Ajish Babu, Sebastian Bartsch, Leif Christensen, Florian Cordes, Alexander Dettmann, Stefan Haase, Javier Hidalgo Carrió, Peter Kampmann, Daniel Kühn, Malte Langosz, Sankar Natarajan, Stefan Stiene, and Sebastian Stock for sharing their experience and journeys, providing inspiration and moral support throughout the past years. A special thanks goes to Florian Cordes for his relentless reviewing and nagging to bring this thesis to another level.

I also want to thank Joachim Hertzberg for agreeing on acting as second evaluator of this thesis. A 'thank you' to the members and alumni of the institute who sent a smile, shared a laugh and created a work environment which allowed to get through this thesis.

The main work presented here has been performed in the projects RIMRES, TransTerrA and FT-Utah which were funded by the German Federal Ministry of Economics and Technology (BMWi) through the German Space Agency (DLR) Grants 50 RA 0904, 50 RA 1301 and 50 RA 1621.

Finally, I want to express my deep gratitude and love to the most important people, my family.

Contents

1	Introduction	1
1.1	Research objective	3
1.2	Methodology	4
1.3	Contributions	5
1.4	Thesis Structure	6
2	Reconfigurable Multi-Robot Systems	9
2.1	Background	10
2.1.1	Reconfigurable System Properties	11
2.1.2	Implementations of Reconfigurable Multi-Robot Systems	15
2.2	The Reference Systems at Hand	18
2.2.1	LUNARES – Reconfigurable Robots for Extraterrestrial Exploration	18
2.2.2	RIMRES - Reconfigurable Integration Multi-Robot Exploration System	20
2.2.3	TransTerra & FT-Utah - Semi-Autonomous Cooperative Exploration of Planetary Surfaces	23
2.2.4	Categorisation	25
2.3	Future Robotic Space Missions	25
2.3.1	Autonomous Operation	26
2.3.2	Dynamic Team Operations	27
2.4	Defining Reconfigurable Multi-Robot Systems	28
2.4.1	Assumptions	31
2.5	Discussion	32
2.6	Summary	33
3	Organisation Modelling	35
3.1	Background	36
3.1.1	Organisation Models	37
3.1.2	Organisational Metrics	40
3.1.3	Knowledge-based Reasoning	40
3.1.4	Coalition Games	44
3.1.5	Combinatorics	45
3.2	Modelling Approach	47
3.3	Querying Suitable Agents	56
3.3.1	Feasible Agents	56
3.3.2	Suitable Agents	59
3.4	Suitable Coalition Structure	66
3.5	Organisation State	67

3.5.1	Agent Type Properties	68
3.5.2	Organisation Properties	70
3.5.3	Organisation Performance Metrics	74
3.5.4	Negative Effects	75
3.6	Discussion	75
3.7	Summary	77
4	Mission Planning	79
4.1	Background	80
4.1.1	Vehicle Routing Problem	80
4.1.2	Planning	83
4.1.3	Hierarchical Task Networks	87
4.1.4	Temporal Planning and Dealing with Time	88
4.2	Robotic Missions	90
4.2.1	Mission Specification & Representation	93
4.2.2	Mission Solution & Cost Function	97
4.3	Spatio-temporal Planning	98
4.3.1	Stage 1: Synchronisation of Agent Activities	99
4.3.2	Stage 2: Bounding Agent Type Cardinality	101
4.3.3	Stage 3: Assignment of Agent Roles	103
4.3.4	Stage 4: Generation of Agent Role Timelines	105
4.3.5	Stage 5: Local Optimisation	105
4.3.6	Stage 6: Solution Validation	108
4.3.7	Stage 7-9: Candidate Characterisation	109
4.3.8	Optional Extensions	109
4.3.9	Implementation Notes	110
4.4	Evaluation	110
4.4.1	On Demand Transport	111
4.4.2	Constrained Coalition Formation	114
4.4.3	Space Mission	115
4.5	Discussion	117
4.6	Summary	121
5	Operational Infrastructure	125
5.1	Background	126
5.1.1	Architectural Templates	126
5.1.2	Robotic Frameworks	130
5.1.3	Distributed Communication & Ad-Hoc Networks	131
5.2	A Distributed Communication Architecture	132
5.2.1	Message Transport	133
5.2.2	Distributed Sensor Network	136
5.3	Evaluation	137
5.3.1	Message-based Communication	138
5.3.2	Mesh-based Communication	140
5.3.3	Multi-Robot Coordination	142
5.3.4	Standardisation for Interoperability & Maintenance	143
5.4	Control Architecture	144
5.4.1	Hierarchical Model-driven Design	145

5.4.2	Controlling Reconfiguration	146
5.5	Discussion	149
5.6	Summary	150
6	Field Testing	151
6.1	Field tests in Utah	152
6.1.1	Analogue Multi-Robot Mission	152
6.1.2	Validation of Subsystem Functionalities	158
6.2	Autonomous mission in Bremen	161
6.2.1	Baseline Mission	161
6.2.2	Experiments	163
6.3	Discussion	172
6.4	Summary	177
7	Conclusion & Outlook	179
7.1	Thesis Summary	179
7.2	Lessons Learned	181
7.3	Outlook	182
7.3.1	Organisation Modelling	183
7.3.2	Planning	184
7.3.3	Infrastructure	184
7.3.4	Application of Real Reconfigurable Multi-Robot Systems	185
A	Robots: Atomic Agents	187
A.1	Atomic agent properties	187
A.2	Classification of electromechanical interfaces	189
B	Ontologies	191
B.1	Base Ontology	191
B.2	Application-specific Base Ontology	195
B.3	Robot Ontologies	199
B.4	Project Ontology	206
B.5	Interface Test Ontology	206
C	Combinatorics	209
C.1	Generation of composite agent combinations	209
C.2	Generation of limited combinations	209
D	Pictograms	213
D.1	Functions	213
D.2	Robotic systems	214
E	Router configuration	217
E.1	Configuration files	217
F	PDDL: Domain and Problem Examples	219
F.1	Domain definition	219
F.2	Problem definition	221
F.3	Solution: an action plan	222

G	TemPl: Representations and Tools	223
G.1	XML-based representation for the Mission Specification	223
G.2	Multi-commodity min cost flow	226
G.3	A graph-based plan visualisation	228
G.4	Comparing temporally expanded network sizes	229
G.5	Details on the space mission example	229
H	Operational Infrastructure	231
H.1	Interaction protocol examples	231
H.1.1	Distributed locking	232
I	Field testing	235
I.1	Field trial Utah	235
I.2	Autonomous Mission in Bremen (RH1)	236
	Terminology	239
	Acronyms	241
	List of Figures	245
	List of Tables	251
	Bibliography	253

1

Introduction

In the year 1997 the first human-made mobile robot drove autonomously on Mars (Bajracharya, Maimone, and Helmick 2008). On Earth and more than twenty years of research later autonomous mobile robots are slowly entering the public space. In 2018, autonomous household robots cut lawns and vacuum floors and the research and advisory company Gartner (2018) sees the technology 'Autonomous Mobile Robots' close to the so-called 'Peak of Inflated Expectations'. This peak is typically followed by a 'Trough of Disillusionment' as result of open technological challenges. Autonomous cars are still tested on public streets and fully autonomous tractors and harvesters remain under development to support farmers (Quartz 2017). Robotic researchers continue to look for ways to design physical agents with human or even superhuman capabilities. One of the open, yet major challenges lies in achieving robotic systems with a human-like ability to adapt to changing environments and situations.

Adaptivity is an essential capability of our human species 'homo sapiens' and key to our species survival. Five climate and eight vegetation zones exist on Earth and they host a variety of landscapes. Some of these regions, for instance the polar zone, appear to be rather unsuited for human inhabitation. Even more, considering that the human body requires to maintain a constant temperature in a narrow band around 36.7 °C. Still, humans manage to survive in many climate zones including the polar zone.

One of the reasons for the survival in different environments is the human body's ability to maintain a relatively stable physiological state (Frisancho 1993, p. 11). Biological adaptation strategies allow the human body to face what Frisancho refers to as 'environmental stress'. Eskimos, for instance, have an increased blood flow which makes them well suited for cold climate conditions (G. M. Brown et al. 1954). Furthermore, humans which spent time in high altitudes trigger a metabolic change so that their body will produce more red blood cells for oxygen transport (D'Alessandro et al. 2016). This metabolic change can be complemented by an additional adaptation of the ventilation rate to compensate for thin air exposure (Moore 2000). Further examples of adaption can be found for underwater eye sight (Gislén et al. 2003),

breath-holding (Schagatay 2014) or the human immune system (Wikipedia 2018a). Roberts and Stewart (2018) even consider the ability to specially adapt to environmental conditions as key characteristic for the human species turning them into a so-called 'generalist specialist'.

Humans have also established a permanent presence in space - an extreme and hostile environment. This is, however, not due to the previously described biological adaption strategies which are the result of evolution over thousands of years. Instead, humans have invented technological means for adaptation allowing them to survive in hostile environments such as the deep-sea or space. Among these technological means are special suits for diving and space walks, and spacecrafts such as the International Space Station. Accordingly, Ilardo and Nielsen (Ilardo and Nielsen 2018) claim technological developments and methodological advancements, e.g., in hunting and storing food, as primary reason for the broad human presence in various environments.

Technology does, however, not only provide direct location access for humans. When environments are too hazardous or still inaccessible for humans, robots can serve as proxies. Robots are already used in the above mentioned areas: deep-sea (MBARI 2018) and space (NASA Jet Propulsion Laboratory 2018). Any robot that operates in these areas primarily needs to be robust enough and withstand the environmental threats - such as extremes of pressure, temperature or radiation. The deployed robots are constantly threatened with the risk of partial or complete loss due to the unknowns of the environment in which they are operating. Loss might simply be caused by a malfunction of a subsystem or by damage from 'interacting' with the environment.

The development of capable robots for remote activities in deep-sea and space comes with high technical demands, firstly to establish the nominal functionality and secondly to perform risk mitigation. In effect, robots designed to operate in extreme environments are typically so costly, that the predominant control mode of robots is manual operation. Manual control of remotely deployed robots allows human personnel to react to unforeseen events and system failures. In these cases a human operator serves as capable problem solver who initiates a required system adaptation. While a manual control approach is effective, it is likely to be less efficient compared to a fully autonomous control approach. To minimise the cost of a robot operation and maximise efficiency in terms of activities over time, an increase of robot autonomy is an obvious means. Still, an application of autonomous robots has to come with a significant benefit compared to a manual approach, due to the remaining operational risks. Such a benefit might be achieved, when autonomous robots are sufficiently adaptive to cope with changing environmental conditions and perform failure handling at a comparable or superior level to human controlled robots.

Where adaptivity is demanded, so-called reconfigurable multi-robot systems offer a solution approach which is based on hardware modularity. A popular reference to reconfigurable multi-robot systems is found in the toy robots Transformers (Wikipedia 2018b). Reconfigurable multi-robot systems come with a customizable degree of modularity at hardware level. This modularity permits adaptation of the morphology and dependant robot properties and allows multiple robots to merge. Thereby, reconfigurability enables robotic teams to dynamically react to new challenges. Standardised interfaces allow a reconfigurable robot to attach new hardware and exchange or share resources with other reconfigurable robots. Hence, an application of reconfigurable multi-robot systems is attractive due to their superior options for adaptation compared to monolithic robots.

Exploiting these options and especially exploiting them with autonomous robots is an open challenge. Any operation and automation of reconfigurable multi-robot systems has to firstly address the existing difficulties for enabling single autonomous robots. Furthermore, the ability to reconfigure poses a combinatorial challenge - even more when handling heterogeneous teams. This demands scalable solutions. Despite the existing challenges, the autonomous exploitation of reconfigurable multi-robot systems comes with significant potential for increasing the adaptivity of existing robot applications. It requires, however, novel approaches in the areas of system modelling, activity planning and team operation.

1.1 Research objective

Physically reconfigurable robotic systems have been studied in the context of swarm-based systems. Most of these approaches rely on homogeneously designed modules which can interconnect. A highly modular design approach and the use of general-purpose modules introduces a flexibility to design modules and achieve robot coalitions of almost arbitrary shapes. Nevertheless, the actual capabilities of such coalitions of modules have been less sophisticated compared to specifically designed robots. In addition, the focus of previous research has been on shape shifting, i.e., planning and performing a single transition from one to another robot morphology.

In contrast to previous approaches, this thesis deals with reconfigurable multi-robot systems using a highly modular design in combination with individually capable robots. The use of a hybrid approach has not only a higher practical and immediate relevance for real world applications. It also allows for expanding the view onto reconfigurable multi-robot systems by explicitly accounting for a heterogeneous team. Possible designs of reconfigurable systems have been evaluated with the implementation of multiple real robotic teams by Bartsch et al. (2010), Roehr, Cordes, and F. Kirchner (2014), Sonsalla, Cordes, L. Christensen, Roehr, et al. (2017a), and Wilcox et al. (2007) ; the systems are introduced in Chapter 2. The offered flexibility of these robotic teams makes them attractive for many application areas, yet the primary target application considered in this thesis is planetary space exploration.

Traditional space operation favours manual control over a remotely deployed system which acts in a potentially hazardous environment. Any manual approach of operating a reconfigurable robot team becomes a challenge, due to the number of systems involved and the reconfiguration options that have to be considered. Firstly, each individual robot which is involved in the operation needs to be controlled and coordinated. Hence, either the number of operators is proportional to the number of robots or the robots cannot be controlled with a maximum degree of efficiency. Secondly, the ability to combine multiple robots leads to numerous robot coalitions which can be considered for the operation. Therefore, an operator has to identify which coalition is feasible and useful for the operation.

This thesis intends to enable and facilitate the use of reconfigurable multi-robot systems. The main research objective lies in the investigation of an autonomous operation approach for such robotic teams that can account for efficiency and safety of robotic operations. Several subordinated research objectives exist dealing with the engineering challenge to develop an autonomous control approach for multiple autonomous and reconfigurable robots:

- (a) creating a model which can represent the properties of a reconfigurable multi-robot

- team and more particularly agent compositions,
- (b) designing a planning system which can account for dynamically created agent compositions, while respecting connectivity constraints and resource availability,
- (c) supporting a heterogeneous, fully distributed agent team which can dynamically change and where agents appear and disappear,
- (d) proving the feasibility of a fully autonomous reconfiguration sequence, and finally,
- (e) identify and generically solve practical issues which come with the application of real robots.

1.2 Methodology

This thesis sets its focus on reconfigurable multi-robot systems, which are a specialisation of multi-agent systems, but a generalisation of multi-robot systems. To deal with reconfigurable multi-robot systems this thesis uses a theoretical as well as a practical approach. The theoretical approach finds its origins in the hands-on experience with has been gained from the operation of multiple real robotic systems and an evaluation of multiple reconfiguration approaches. Three robotic teams have been worked with along this thesis and they are described in Chapter 2. The developed control and operations approach for reconfigurable multi-robot systems is the result of an iterative process of coevolving theory and applied solution approaches. The teamwork model originally defined by Wooldridge and Nicholas R. Jennings (1999) and further formalised by B. Dunin-Keplicz, Strachocka, and Verbrugge (2011) and B. M. Dunin-Keplicz and Rineke Verbrugge (2010) serves as envelope for using reconfigurable multi-robot teams. Four essential stages are considered for cooperative problem solving:

- (a) **potential recognition:** identification of a potential for cooperation by an agent
- (b) **team formation:** search for agents that will cooperate towards a goal
- (c) **plan formation:** (collectively) plan towards a shared goal
- (d) **team action:** perform the planned activity and control the execution of the plan

This model is the basis for the application of reconfigurable multi-robot systems and this thesis embeds additional considerations. After the recognition of the cooperation potential the process of team formation requires methods to identify and deal with physically combined systems and temporally enabled abilities. An operational infrastructure enables dynamic team formation while considering all active systems that are in communication reach. Plan formation can exploit the special features of a reconfigurable system and minimise the resource usage, so that only a subset of the identified systems will be used for the actual performance. Achieving the shared goal with a reconfigurable multi-robot system takes advantage of morphology changes or rather transitions between so-called coalition structures. Therefore, the plan formation as well as the team action account for intermediate physical coalitions of robots. Although a plan could be negotiated between all members of a reconfigurable system, this thesis focuses on a centralised planning approach: after all available agents have been identified it is assumed that a single agent performs the planning process to outline the team activity and the same or another agent controls the execution of the planned team activity.

1.3 Contributions

This thesis contributes to the state-of-the art by investigating and implementing an automation approach of physically reconfigurable multi-robot systems. By using a holistic approach for the autonomous operation of reconfigurable multi-robot systems this thesis identifies essential requirements and solution components. State-of-the art approaches from organisation theory, knowledge-based systems, multi-agent and operations research are combined into a unique interdisciplinary solution approach. In addition, the contributions of this thesis are as follows:

I Formalisation of Reconfigurable Multi-Robot Systems

Introduction of a formal description for reconfigurable multi-robot systems. The formalisation combines multi-agent game theoretic approaches with the particular aspects of embodied agents. The model is validated through its use in an organisation model and by the development of a planner for reconfigurable multi-robot systems.

II Organisation Model

The organisation model Model for Reconfigurable Multi-Robot Organisations (MoreOrg), developed in this thesis, uses a knowledge-based approach firstly to represent the functionalities of a reconfigurable multi-robot system and secondly to reason with it. It maps between resource structures and functionalities and thereby enables the characterisation of dynamically formed composite systems. MoreOrg implements a generic approach for identifying embodied composite agent systems which can support a particular functionality. The model introduces a functional saturation bound to improve the efficiency of reasoning, so that the model remains applicable to reconfigurable multi-robot systems with many members. In addition, MoreOrg embeds an adaptation of an anytime coalition generation approach to identify functionally constrained coalition structures. The model can be used for system analysis by mission operators and it serves as core element of a multi-robot mission planning approach.

III Mission Planning

Temporal Planning for Reconfigurable Multi-Robot Systems (TemPl) is a result of this thesis. It is a constrained-based mission planner for reconfigurable multi-robot systems. The planner introduces a mission description as generalisation of a Vehicle Routing Problem (VRP) with synchronisation constraints. Planning with reconfigurable multi-robot systems is translated into a multi-commodity min-cost flow problem for local optimisation. By embedding the organisation model MoreOrg TemPl accounts for a generalised way of dealing with physically constrained agent coalitions. TemPl has to deal with a combinatorial search problem and this thesis suggests effective bounding strategies which are based on an agent type based representation.

IV Operational Infrastructure

Identification, development and combined application of methods and technologies to support the operation and maintenance of reconfigurable multi-robot systems. Implementation of a space-related operational infrastructure using a standard specification originally developed in the multi-agent community. Part of the infrastructure is a communication system that supports dynamically changing and fully distributed agent teams. The infrastructure serves as general reference implementation for supporting the flexibility and dynamics of reconfigurable multi-robot systems and it complements the robotic framework Robot Construction Kit (Rock).

V Operation of Reconfigurable Multi-Robot Systems

This thesis illustrates and analyses an automation approach of a field-tested reconfigurable multi-robot system. The author has contributed to the (semi-autonomous) operation of three reconfigurable multi-robot teams, leading to the identification of major design criteria to use reconfigurability of real robotic systems, but also limitations thereof. The performance of a Mars-analogue mission using the semi-autonomous operation of a multi-robot system illustrates the validity of the general operational infrastructure. This thesis presents an empirical analysis with focus on the communication characteristics of robot to robot communication as well as robot to mission control during this Mars-analogue mission. A fully autonomous sample return mission based on two mobile and one immobile agent has been implemented and evaluated. The empirical evaluation of several experiments with real hardware identifies current limitations of using reconfigurable multi-robot systems with the selected automation approach.

VI Open Robotics Community

Several elements of the operational infrastructure are open-source accessible and have been integrated into the robot framework Rock. This includes the implementation of the Foundation for Intelligent Physical Agents (FIPA) message standard (Roehr 2013a,b) as first publicly available implementation which includes the bit-efficient message standard. An algorithm for the efficient generation of combinations by exploiting repetitions has been embedded into the Numeric Library (Schwendner, Joyeux, et al. 2012). A tool for facilitating the generation of binary packages for custom software releases to operate (multi-)robot systems while permitting hierarchical release structures has been developed as part of this thesis (Roehr, Willenbrock, and Joyeux 2018). Further publicly accessible are a C++-library for working with ontologies (Roehr 2019) and a library for handling graphs and graph-based problems, such as solving multi-commodity min-cost flow problems (Roehr, Munteanu, et al. 2019).

1.4 Thesis Structure

The structure of this thesis is depicted in Figure 1.1. Chapter 1 outlines the general approach and contributions of this thesis, while Chapter 2 introduces the general motivation and highlights several design decisions which have been made for the implementation. Furthermore, Chapter 2 presents the robotic teams that are the basis of this work and concludes with the formalised core description of reconfigurable multi-robot systems. Chapter 3 and 4 establish the formal basis and core algorithm for autonomous multi-robot missions. Chapter 3 introduces the model MoreOrg which implements a knowledge-based reasoning system to deal with dynamically created and constrained agent coalitions. The organisation model serves as basis for a constraint-based mission planning approach which is described and evaluated in Chapter 4. This planning approach combines the use of a temporal database and vehicle routing problems, and considers efficacy, efficiency and safety as optimisation criteria. The design considerations and essential components of an operational infrastructure are presented in Chapter 5; the focus lies on the support of dynamically changing, distributed agent teams. The operation infrastructure has been used and validated in field tests. Chapter 6 illustrates the performance of a semi-autonomous multi-robot mission and a fully autonomous sample return mission. The autonomous operation of a sample return mission illustrates the general feasibility of using reconfigurable multi-robot systems in a real mission. Furthermore, these experiments illustrate

the remaining gap between mission planning and execution. Chapter 7 concludes this thesis and provides an outlook to future research activities and newly opened research opportunities.

Note that this thesis does not comprise an overall background chapter. Where required the description of the relevant state of the art is embedded into the chapter. To facilitate the reading a list of selected terms, an acronym list and a symbol list are at the end of this thesis.

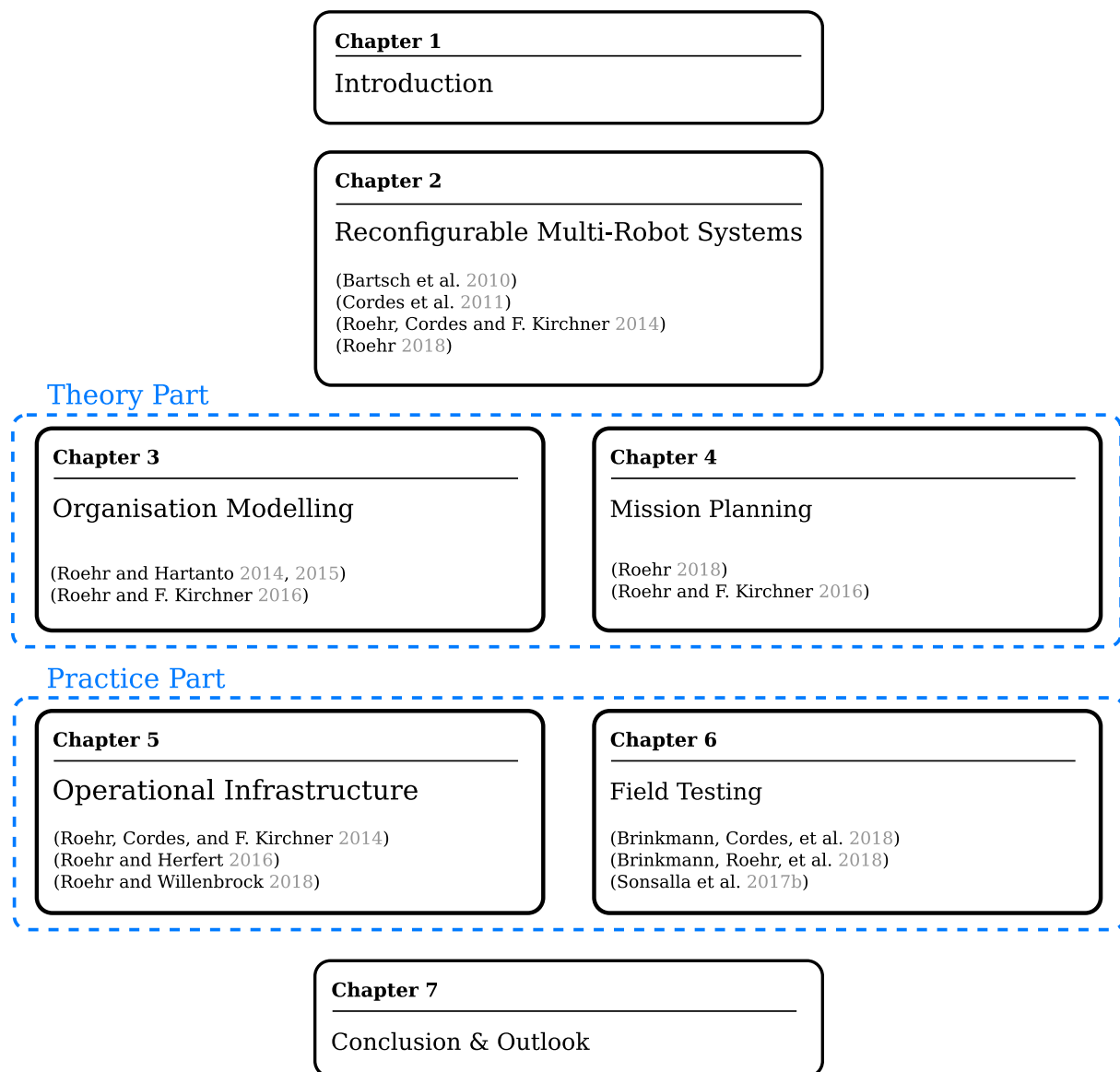


Figure 1.1: Main chapters of this thesis with respective references to authored publications.

2

Reconfigurable Multi-Robot Systems

Disclaimer *This section introduces and expands definitions and ideas which have been published in (Roehr 2018; Roehr, Cordes, and F. Kirchner 2014).*

Multi-robot systems have one major attribute which makes them a special case of multi-agent systems: the agents come with embodiment and thus are physical agents. Reconfigurable multi-robot systems, however, are a more general case of multi-robot systems since robots with the capability to physically interconnect are part of the team. The term *reconfiguration* can generally refer to the change of the structure and parametrisation of a system. With respect to reconfigurable multi-robot systems the term typically refers to the system's ability to change the physical appearance and morphology. This change of morphology can result from adding or removing elements such as structural parts or even complete robots to the system, or just by rearranging existing elements of the system. The ability to change a single robot's morphology and the physical structure of a robotic team is the key characteristic of a reconfigurable multi-robot system. The effects of morphology changes have to be supported and exploited by a software control architecture and thus a physical change often results in a software configuration change. Hence, the aspects of reconfiguration can cover hardware as well as software, where the consideration of available options for software configuration leads to a significantly larger (re)configuration space than for hardware only.

This chapter provides context and background information on reconfigurable multi-robot systems in general in Section 2.1. Section 2.2 introduces the actual hardware used for the development of this thesis. The section highlights core features of the used robotic systems, while complementary details are listed in the Appendix A. Since this thesis finds its motivation in a space related application context Section 2.3 anticipates the benefit for future planetary space missions. Essential definitions and assumptions for the subsequent chapters are introduced in Section 2.4. This chapter is concluded with a short summary in Section 2.6.

2.1 Background

Reconfigurable multi-robot systems introduce flexibility compared to existing robot implementations. The dimensions of change cover structural and/or functional levels as illustrated in Table 2.1. The possibility for reconfiguration leads to a larger, though still restricted configuration space, since dedicated connection interfaces are used. The challenge for autonomous operation lies in managing the resulting configuration space, accounting for physical constraints while planning activities and finally performing reconfiguration.

Table 2.1: *Dimensions of change (adapted with permission from (Roehr, Cordes, and F. Kirchner 2014) ©2013 Wiley Periodicals, Inc.).*

	physical / hardware	virtual / software
structural	change of morphology, tool exchange	change of distribution of software modules across physical devices, reorganizing and relinking data flow, changing dependencies for running components, update kinematic models according to the morphology changes
functional	tool exchange, modes of operation: wheel also being used as foot or sensing device, manipulator also being used as supporting leg	modalities, application of various solution strategies, parametrisation of components, e.g., adaptation of thresholds, configuration parameters in a signal processing chain
mixed	change of morphology changes the set of active capabilities, and for exploitation requires adaptation of the high-level software stack	

Physical reconfigurable systems are a niche product in the area of robotics research, but a significant overlap exists with other areas such as operational research and management studies. V. Dignum (2009) looks at reconfiguration in the context of organisation research. According to Dignum the general need to actively organise robotic teams aims at increasing efficiency, and she sees flexible and adaptive organisation as suitable means to deal with dynamic environments. She suggests that organisations conditionally adapt and should reorganise if this will lead to an increasing success of an organisation; even a suboptimal reorganisation can be better than no response at all. However, the question when to reorganise and when to accept loss is left unanswered. In her work she points to *strategic flexibility*, a concept developed in the scope of managing high-technology industries by Evans (1991).

Evans introduces what he calls "A conceptual framework" for strategic flexibility of enterprises. This framework conceptualises the strategic use of a company's or more generically a market player's flexibility. Flexibility to adapt leads to a significant competitive advantage, since it offers a market player additional means to encounter unforeseen events. Hence, adaptation can directly lead to a greater probability of survival or net monetary benefit for market players. According to Evans the use of a player's manoeuvres can be classified along two dimensions: temporal and intentional. Table 2.2 illustrates the resulting two dimensional matrix and the categorisation.

Evans' conceptual framework is general enough to be applied to the application of reconfigurable multi-robot systems, and the categorisation of manoeuvres can be similarly applied for a characterisation of robotic activities: protective and corrective activities count as defensive

Table 2.2: *Characterisation of manoeuvres according to Evans (1991).*

		<i>temporal</i>	
		proactive	reactive
	<i>intentional</i>	offensive defensive	preemptive protective exploitive corrective

manoeuvres. Protective activities already start with the design of a reconfigurable multi-robot system and aim at risk minimisation, e.g., by increasing inbuilt safety buffers. An online management of redundancies can therefore be considered a protective activity. Corrective activities address the issue of repair and compensation of a (temporary) fault or loss of a device. The restructuring of the complete organisation can be considered a corrective activity, e.g., when it results from compensating the loss of a complete robot. Exploitive operations can be seen in the context of so-called opportunistic science. Estlin et al. (2007) uses such operations in the area of space-exploration. Opportunistic science can be even better supported, when redundancies in the overall reconfigurable multi-robot system exist. While redundancies in general represent a safety buffer, they can then also be exploited for spontaneous scientific harvesting. Hence, redundancies permit to change a mission to react to non-deterministic external triggers. Pre-emptive activities are already part of the design stage of reconfigurable multi-robot systems, since the standardisation of electromechanical interfaces (EMIs) "create[s] a range of options before they are needed" (Evans 1991).

2.1.1 Reconfigurable System Properties

Evans has identified the flexibility to adapt as a key advantage of reconfigurable systems. A strategic exploitation of this flexibility leads to a control change of system properties. Evans uses observations of an artificial, human-made marketplace with a number of players. Complementary, the following paragraphs focus on natural systems. Natural systems not only provide motivating examples for the benefit of reconfigurable systems. They also point to templates and approaches for autonomous operation of reconfigurable systems. The focus is on the system properties efficacy, efficiency, and safety. Chapter 4 shows how these properties are used as optimisation objectives.

Efficacy

Any reconfigurable system inherently offers a greater potential compared to monolithic systems to adapt to tasks and challenges, and hence comes with a broader scope of application. This means a reconfigurable system offers an increased level of efficacy, where the Oxford University Press (2018) defines efficacy as "The ability to produce a desired or intended result". Reconfiguration is still no guarantee for increased efficacy. However, a reconfigurable system - as composition of multiple agents - opens the opportunity for superaddition (Weiss 2009, p. 315). Superaddition is an effect known from the concept of complementarity; a combination of two or more systems is superadditive, when the value of the overall system is greater than the sum of the individual system values. From an external observer's point of view, a composite system that exposes capabilities which none of the compositing individual systems

offers can be called superadditive. Fire ants' bridge building behaviour (Mlot, Tovey, and Hu 2011) serves as an example for superaddition in natural systems: fire ant colonies are often threatened to drown after a flood. Hence, fire ants have developed a rafting behaviour, which allows worker ants to connect to a floating raft to carry and save the colony. So while individual ants can only float, a larger group of ants is capable of providing a complete raft as a service to fellow ants. A similar conditional activation of behaviour has been observed by Bellwood, Hughes, and Hoey (2006) identifying so called "sleeping functional group[s]": while looking at the symbiotic relationship of a coral reef they have observed an increase of what they identify as "unusual feeding behaviour" of batfish. Instead of following their usual diet, which consists of plankton or so-called benthic invertebrates, batfish switch to feeding from algae. This setting can be observed in a dysfunctional coral reef which suffers from extraordinarily microalgae growth. The triggered collective change of batfish behaviour, however, has been identified as main contributor to the reversal of a dysfunctional coral reef. While the rafting behaviour illustrates the natural counterpart for a new functionality arising from physical reconfiguration, the rescue of the coral reef depends only upon a collective and temporarily change. In both cases, the exposed functionality of the complete organism is greater than the functionality shown by each individual, here: exposing rafting and coral reef rescue.

Efficiency

An increase of efficiency depends upon two factors: time and resource usage (typically energy). Shortening execution time for the same task and/or using fewer resources of a system can improve efficiency. Reconfigurable systems especially offer an efficient use of resources with ridesharing as a prominent example in natural system, e.g., marsupials carry their offspring in a pouch, which allows the animal parent to remain mobile, while still protecting their offspring or being able to forage. The degree of specialisation varies, and other animal species show less efficient approaches of transport. Quadruped mammals, for instance, carry their offspring between their teeth and temporarily give up some protective or foraging capabilities. This can be interpreted as a negative effect of reconfiguration (cf. Chapter 3.5.4). The concept of marsupial systems has already entered the field of robotics, where various approaches have been implemented, including combinations of terrestrial and aerial systems to evaluate specifically the marsupial transport concept. In contrast, Roehr, Cordes, and F. Kirchner (2014) encounter ridesharing concepts while developing a more generalised concept of reconfigurable multi-robot systems. The example of shared transport shows how an increase of efficiency in the overall system can be achieved by exploiting the specialisation of a single system. Robots can offer this specialisation, here a more efficient transport to save transport time, as a service to other robots. In addition to exploiting specialisation the use of parallel activities in general offers further potential for increasing efficiency with multi-agent systems. Heterogeneous multi-agent systems can reduce wait times further, given they (a) follow a cooperative collaboration scheme with a global optimisation strategy, and (b) comprise team members which are specialised (and thus more efficient than other team members) for performing particular tasks.

Safety

Improving safety properties is of particular interest in space applications. Reconfigurable multi-robot systems offer means to introduce new approaches for maintaining safety properties. Resource redundancy is one such safety property. Increasing resource redundancy while

lowering costs is a major principle of swarm-based approaches in robotics. In natural systems animal groups such as swarms and large herds offer not only properties to raise the probability of survival of the organism as a whole, but raise the probability of survival of each individual member (Hager and Helfman 1991; Pulliam and Caraco 1984). Foraging is basis for the survival of natural systems, and reducing the time to search for food increases the probability of survival, since time remains for additional defensive activities. On the one hand food sources have to be defended against competitors, and on the other hand defence strategies are required to protect against predators. Forming groups is one strategy, e.g., observed for fish which form large schools to deceive attackers and such behaviour has been not only observed for groups with a single animal species aka homotypic groups, but as well as for so-called heterotypic groups (Ogden and Ehrlich 1977; Pereira, Feitosa, and Ferreira 2011). Depending on the role and importance of an individual for the group, defensive strategies can change, e.g., an ant colony consists of a large number of redundant workers, but only few queens. Since queens play a vital role for an ant colony, ants have developed special protective behaviours: when performing a colony migration (Franks and Sendova-Franks 2000) queens are surrounded by a large number of redundant workers, so that the queens have an increasing protection during colony migration. This protection strategy shows how a dynamically adapting composite system, such as the ant colony, can focus its (redundant) resources for a temporary and local increase of safety properties.

Robustness and resilience are two characteristics which are strongly related to the aspect of safety, and are therefore also used by Evans (1991) to describe defensive system capabilities. While robustness refers to a system which can endure impacts up to a certain degree without breaking, resilience results from the ability to recover from error and return into a functional state. Especially resilience is a key to survival, and not only in natural systems, but for technical and social systems alike shown by examples collected from Zolli and Healy (2012). Resilient systems, however, require adaptation. Therefore, reconfigurability can contribute to an increased resilience of robotic systems.

Interdependence & General Benefit

Safety, efficiency and efficacy are interdependent. An increased efficiency and efficacy contribute to a higher level of safety or rather a higher probability of survival. The need for less resources for the same tasks, and thus a higher efficiency, opens opportunities to consider additional or alternative tasks. Animals like meerkats or cynomys assign individual group members to watch out for predators and warn the rest of the group. This is only possible, when a group shares resources and spares individual members from the foraging task and instead allow them to specialise, e.g., here on guarding (Alexander 1974). Yet, the group might also pay for an increased safety by a decreased efficiency. Pulliam and Caraco (1984) search for an optimal group size in natural systems, but conclude that “an optimal group size may not exist” or will not be found at a natural system’s equilibrium.

Reconfigurable robotic systems encompass the properties of single, multi, and even swarm robotic systems. Table 2.3 illustrates a high-level categorisation of selected attributes of each system type. The focus on designing a single, and typically monolith robotic system leads for example to capable robots which can perform space exploration missions to Mars (NASA Jet Propulsion Laboratory 2018). Although multiple rovers have been deployed on Mars, these systems are not designed to cooperate. They operate independently and rather tightly con-

Table 2.3: *Distinguishing properties of robotic system types: positive attributes are marked in green, negative in red, neutral in yellow, and in purple a mix of positive and negative attributes.*

Properties	Robot system types			
	<i>single</i>	<i>swarm</i>	<i>multi</i>	<i>reconfigurable multi</i>
individual capabilities	advanced	low	low to high	low to high
team capabilities	n/a	medium	advanced	advanced
parallel activities	n/a	yes	yes	yes
individual system cost	low to high	low	low to high	low to high
individual redundancy	fixed	fixed	fixed	adjustable

trolled by human operators. Parallel activities of a single system mainly results from a software based parallel computing approach, although in principle parallel physical interaction, e.g., using multiple manipulators is possible. In contrast to single systems, all variants involving multiple robots naturally can perform parallel activities.

A major distinction between swarm-based and multi-robot systems is the cost factor: swarm-based systems follow a design philosophy using homogeneous, low-cost robots which are controlled by (emergent) behaviours. This bottom-up control approach is a key characteristic of swarm-based systems and it makes swarm-based systems adaptive. However, it turns into a drawback when a validation of such systems is required. The simplicity and low-cost factor of swarm-based systems still makes them attractive for an application in areas, where the loss of individual devices must or can be tolerated. These features motivate an application of swarm-based systems in space, but the verification of emergent algorithms poses a significant challenge according to Truszkowski, Hinchey, and Rash (2004, p. 52). In contrast, multi-robot systems follow no strict design philosophy, but application specific requirements and anticipated environmental settings drive their design. However, implementation costs are proportional to the degree of complexity and capability and will therefore be typically higher than for swarm-based systems.

Regarding safety, Table 2.3 refers to the redundancy of individual agents, where an agent in the reconfigurable multi-robot system can also be a composite system. The redundancy of individual agents is static for any monolithic single, multi, and swarm robotic system. Meanwhile, reconfigurable multi-robot systems have the possibility to adjust redundancy by physically sharing and exchanging resources. With the assumption, that an increased resource redundancy also increases safety, reconfigurability in multi-robot systems is a means to reduce the operational risks. Alternatively, a system can perform more challenging and thus potentially riskier operations when the safety level is temporarily increased in parallel. The possibility to exchange resources between individual team members can also lead to new operational approaches and related cost reductions by using dynamically adaptable safety buffers. In effect, reconfigurable multi-robot systems can perform high-priority robot actions with higher safety buffers than other actions. Hence, compared to a monolithic robot which requires the same maximum redundancy level, the reconfigurable multi-robot system can be designed with a lower average redundancy per robot.

2.1.2 Implementations of Reconfigurable Multi-Robot Systems

Implementations of reconfigurable multi-robot systems come within a spectrum from industrial robots which allow a tool exchange to fully self-reconfigurable multi-robots systems. Research in the area of reconfigurable multi-robot systems has initially been driven by the latter, i.e., the concept of so-called self-reconfigurable systems. This section points to the general characteristics of self-reconfigurable systems with supporting examples to highlight relevant concepts and the differences to the later described references system. For a broader review of self-reconfigurable multi-robot systems the interested reader is referred to the survey papers of Chennareddy, Agrawal, and Karuppiiah (2017) and Liu, X. Zhang, and Hao (2016). This section also looks at a selected set of (electro)mechanical interfaces, which are a necessity of reconfigurable multi-robot systems; they allow modularisation in the first place. Chennareddy, Agrawal, and Karuppiiah as well as Liu, X. Zhang, and Hao leave out the larger, more capable set of reconfigurable multi-robot systems. Therefore, the following sections especially look at distinctive aspects of these larger systems.

Robotic Hardware

The majority of existing self-reconfigurable robots are swarm-based, homogeneously shaped and with few exceptions are composed of a number of compact, lightweight and simple modules. These modules are mainly consisting of interfaces and related hardware to enable the interconnection of modules. To establish a general modular approach and adaptability these interfaces are standardised including the design of either androgynous or gendered connector interfaces, where the latter consists of compatible *male* and *female* connectors.

The categorisation of modular, self-reconfigurable robotic systems according to Liu, X. Zhang, and Hao (2016) and Chennareddy, Agrawal, and Karuppiiah (2017) is based on the following hardware characteristics: (a) structure, (b) form factor, (c) reconfiguration approach, and (d) locomotion schema. The classification for structure encodes the underlying kinematic constraints to form new structures, where the following categories are used: lattice-based, chain-based, hybrid, truss, and free-form objects. Self-reconfigurable systems are formed by atom-like modules that can be composed to replicate almost any system shape. Reconfiguration approaches are categorised as either deterministic or stochastic depending on the control scheme defining how self-reconfigurable systems transitions into a target morphology. Deterministic approach allow a detailed control over the full assembly of the modules and the process to achieve this assembly. Meanwhile, stochastic approaches lacks this level of control in favour of a behaviour-based approach with emergent module structures. A capability looked at in these classification approaches is locomotion. However, the categorisation does not deal with the locomotion capability of an assembly of modules. Instead its focus is on the type of movement of modular components within the system in order to change the morphology of an assembly of modules.

Chennareddy, Agrawal, and Karuppiiah (2017) consider the largest form-factor *macro* by defining a volume threshold of equal or more than 125 cm³. ATRON modules (Brandt, D. J. Christensen, and Lund 2007) serve as example for the typical macro-sized module. ATRON modules are sphere-like, with a diameter of approximately 0.11 m, and have four male and four female interfaces in order to establish mechanical connections with eight other modules at maximum; infrared based sensors enable modules to communicate and sense distances. Multiple modules

can join to form mobile composite systems with different locomotion modes, such as a snake robot, walking system or a car like system. Additionally, Brandt, D. J. Christensen, and Lund illustrate forming a serial manipulator and a conveyor belt.

ATRON modules are only one kind of class of highly granular robots, with a number of similarly relevant implementations such as Telecubes (Suh, Homans, and Marc Yim 2002), Polybot Series (Mark Yim, Duff, and K.D. Roufas 2000; Mark Yim, Kimon Roufas, et al. 2003) Molecubes (Zykov, Chan, and Lipson 2007), M-TRAN (Haruhisa Kurokawa et al. 2007; Murata, Kakomura, and Kurokawa 2008), SMORES (Davey, Kwok, and Mark Yim 2012), and ModRED (Baca et al. 2014) to name only a few. These general modules represent compact units hosting only connectors. For self-reconfiguration the connectors can be rotated, so that the pose of the connection interfaces can be adapted. Existing implementations illustrate a variety of different connection interfaces. Nevertheless, they mainly address the needs to perform reconfiguration into particular shapes, without considering further functional requirements.

The anticipated application area of this thesis, however, is space and space has special environmental conditions that have to be taken into account. The project intelligent Building Block for On-Orbit Satellite Servicing (iBOSS) has developed an androgynous interface called intelligent Space System Interface (iSSI) (Kortman et al. 2015) which has been particularly addressing the requirements for an application in space. Therefore, the iSSI does not only use an electromechanical, but also a thermal coupling mechanism. The iSSI serves as key element in a modular, reconfigurable hardware architecture, which is introduced by Schervan et al. (2017) to compose complete satellite systems. Each $0.4 \times 0.4 \times 0.4 \text{ m}^3$ sized iBOSS module of the component-based hardware system named intelligent Building Block (iBLOCK) can carry subsystem functionality. In contrast to modules like ATRON the iBLOCK has no degree of freedom. In effect, although modular, no self-reconfiguration is possible for a system composed of iBLOCKs only.

Space application is also the target of the six-legged robot ATHLETE (All-Terrain Hex-Limbed Extra Terrestrial Explorer) (Wilcox et al. 2007), which serves as an example for reconfigurable systems with lower granularity. ATHLETE is part of an overall composite system to manage and establish a lunar infrastructure and as such the robot is able to dock specialised end effectors, additional payload modules and other ATHLETE rovers.

S-bots, developed by Mondada et al. (2005) are further class of robots which are capable of autonomous self-assembly. Each s-bot is an autonomous, mobile robot with a diameter of 120 mm, equipped at minimum with a processing unit, camera, proximity sensor and grippers for connection. The gripper allows to establish mechanical connections by attaching to an outer ring of any other robot. Robots can lift other robots using this gripper, but no power or data link connections are established by attaching the gripper. While physically reconfigurable systems require actual mechanical linking, Birnschein et al. (2014) investigate coupling concepts for the domain of autonomous cars and suggest an additional virtual coupling between systems. This virtual coupling can allow virtual road trains to improve the efficiency of transport.

Another type of robotic systems is represented by rovers like Tri-Star IV (Aoki, Murayama, and Hirose 2011) or Scarab (Bartlett, Wettergreen, and Whittaker 2008) and including SherpaTT (see Section 2.2.3). These systems do not require a modular design to be reconfigurable. Instead, they are reconfigurable due to an adaptive locomotion platform. Note that the focus of this thesis is not on exploiting the *embedded* reconfigurability of monolithic robots. The focus is on robots which jointly form a modular and reconfigurable system.

Control Approaches

Highly modular systems can adapt their morphology to activate a target configuration with desired structural or even functional properties. Along with the robotic hardware, various control approaches have been developed for self-reconfigurable multi-robot systems which allow for an either manually predefined or automatically planned transition between two reconfiguration states. Behaviour-based and evolutionary algorithms are popular means to tackle this problem.

Several approaches exist to control the shape shifting process. ATRON modules (Brandt, D. J. Christensen, and Lund 2007) change into a target configuration based on a manually defined sequence of actions. Brandt, D. J. Christensen, and Lund evaluate the concept of meta-modules (an abstract view to a composition of three modules) in simulation to improve scalability; attraction points guide the shape shifting process. In contrast, A. L. Christensen, O’Grady, and Dorigo (2007) use a seed-based mechanism to imitate a stigmergy-controlled build pattern (Camazine et al. 2001): 2D-based shapes such as *rectangle* or *star* serve as target, and a single robot initiates a morphology growing by ‘opening’ connection slots to which other robots can attach. Additional connection slots open iteratively, based on newly connected robots and the target specific (distributed) algorithm.

The ATHLETE rover and its related systems is similar to the systems taken used for evaluation in this thesis in the following ways: a similar level of granularity, heterogeneous system modules, a modular software development approach and space as target application area. As a main difference, however, the ATHLETE rover is not able to self-configure. Wilcox et al. report on the encountered challenges while aiming at an autonomous docking procedure with an alignment tolerance of 1 cm. The implemented approach fails due to a limited ability to compensate for alignment errors and the limited field of view of the cameras’ to track the target alignment markers in close proximity. In effect, Wilcox et al. use only a manually guided docking procedure controlled by a human operator in a ground station.

The number of modules to work with is a limiting factor for both the practical evaluation as well as the scalability of algorithms. Schervan et al. (2017) account for 20 iBLOCKs in a feasibility study. Although they expect more than 50 blocks to be used for larger satellites, they do not suggest or illustrate any approach for (self-)reconfiguration. Mondada et al. (2005) show real-world experiments with up to 20 s-bots for a collaborative transport task and for the collaborative negotiation of an obstacle. Brandt, D. J. Christensen, and Lund (2007) use up to seven modules for most of their real world experiments including a robotic arm, a car and a snake morphology - only the simulation of a conveyor belt uses significantly more. Scalability is tested by Brandt, D. J. Christensen, and Lund in simulation to evaluate shape shifting approaches with up to 500 ATRON modules.

High-level planning approaches with focus on reconfigurable multi-robot systems have not been presented outside the context of this thesis. However, Baca et al. (2014) suggest the search for an optimal coalition in the course of performing dynamic self-reconfiguration. The optimal coalition has a maximum utility value. Their approach is based on coalition games theory and relies on evaluating so-called coalition structures for the reconfigurable multi-robot system. They identify an algorithm by Rahwan, Ramchurn, et al. as suitable candidate, but implement their own approach for finding an optimal coalition. They use, however, a strong assumption - basing utility only on bilateral relationships - to reduce their algorithm’s complexity and at the same time its applicability in this thesis. Dorigo et al. (2005) evaluate a behaviour-

based approach as alternative to form problem specific coalitions. They successfully achieve collaborative transport and bridging behaviour.

The use of reconfiguration can also be found outside of robotics research. Drexel (2013) encounters a similar problem in the context of operations research and more particularly with VRP with Trailers and Transshipments (VRPTT). Drexel considers trucks and attachable trailers to deliver goods to satisfy customer demands. Tackling the planning problem with reconfigurable systems is subject of Chapter 4.

In general, as T. Zhang, W. Zhang, and M. Gupta (2017) point out in their survey paper that a full exploitation of the possibilities of reconfigurable modular robots still requires a higher degree of automation. It therefore remains an open research gap.

2.2 The Reference Systems at Hand

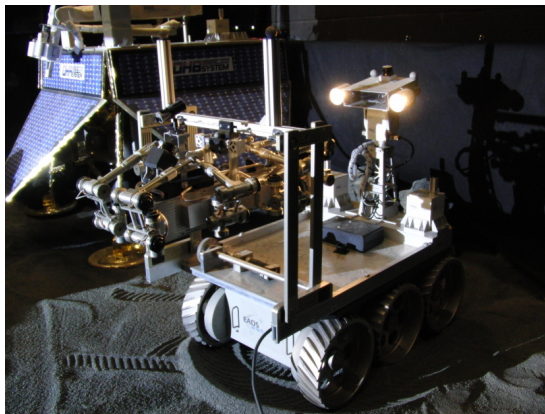
The experience and experiments made in four related robotics projects which deal with reconfigurable multi-robot systems are the basis of this thesis:

- LUNARES (Bartsch et al. 2010; Cordes et al. 2011),
- RIMRES (Roehr, Cordes, and F. Kirchner 2014),
- TransTerra (Sonsalla, Cordes, L. Christensen, Planthaber, et al. 2014), and
- FT-Utah (Sonsalla, Cordes, L. Christensen, Roehr, et al. 2017a).

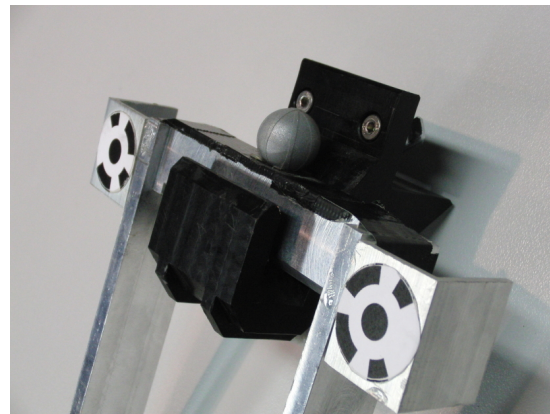
The following sections introduce the hardware and robots this thesis builds upon. All mentioned systems have been used by the author for experimental work. Experiments have either led to quantitative experimental results, explored the feasibility of concepts or spawned new ideas to improve or extend the overall system. The experimental evaluation of the visual servoing approach by Roehr, Cordes, and F. Kirchner (2014) in RIMRES, for example, has led to the adaptation of the design of the EMI in the project TransTerra. The evaluation of three team constellations in different projects has led to a successively advancing design of reconfigurable multi-robot systems. The following sections present the main characteristics of related robotic teams and briefly highlight the shortcoming and incremental improvements in historical order.

2.2.1 LUNARES – Reconfigurable Robots for Extraterrestrial Exploration

The project LUNARES evaluated basic concepts for the operation of multi-robot systems on the lunar surface. It implemented a control approach based on ESA's Functional Reference Model (FRM) (Putz and Elfving 1991). Furthermore, it evaluated cooperative docking manoeuvres to realise a sample return mission. The robotic team consists of a lander mock-up with functional manipulation arm, a rover, and the eight-legged scouting robot Scorpion (Spenneberg and F. Kirchner 2007). LUNARES implemented an exemplary sample-return mission with the following semi-autonomous sequence of activities: (a) lander deploys payload onto rover (b) rover transports the scout to the rim of a lunar crater, (c) rover releases the scout, (d) operator controls descend of the scout into the crater, (e) semi-autonomous pickup of a soil-sample at crater bottom, (f) operator controls ascend of scout, and return to the rover, (g) semi-autonomous docking of scout to rover, (h) rover transports the scout to the lander, and (i) lander extracts payload from rover.



(a) Multi-robot team in LUNARES (Source: DFKI GmbH).



(b) Mechanical docking adapter consisting of hook and handle augmented with visual markers (Source: DFKI GmbH).

Figure 2.1: Components of the robotic team in LUNARES.

The robotic team of LUNARES as illustrated in Figure 2.1a has the following properties: the rover is a marsupial-like system, which can carry the scout, and a semi-autonomous cooperative docking process allows to join both systems as illustrated by Roehr, Cordes, F. Kirchner, and Ahrns (2010). LUNARES reused an existing set of robots and adapted these systems, e.g., by adding a docking adapter depicted in Figure 2.1b. This approach shows that reconfigurability does not necessarily be accounted for at the time of system design. It can be enabled at a later stage. Here, by addition of a hook-based mechanical mechanism.

The cooperative docking approach between rover and scout is based on a master-slave based visual servoing process. It relies on a camera mounted on the rover and a set of position markers attached to the scout. The scout is guided to a target position relative to the rover and into proximity of the released hook mechanism. The lander acted as immobile robotic team member, but was equipped with a manipulator for payload extraction. An additional mechanical extension unit could be optionally attached to the scout to hold a soil sample. The container came with a gripper interface to facilitate extraction using the lander's manipulator. However, this container extraction was not part of the finally implemented mission sequence.

Operations The general operations concept in LUNARES relies on a centralised control architecture. This architecture assumes a mission control centre on ground. On the lunar surface the lander acts as communication endpoint and task controller for the robotic exploration systems. The architecture assumes a continuous connection of the robots to the ground station, so that semi-autonomous control approaches permit to drive the rover, pick-up a payload and dock the scout to the rover. The architecture represents also an implementation of ESA's FRM (see Chapter 5.1.1) - an architecture template for space systems.

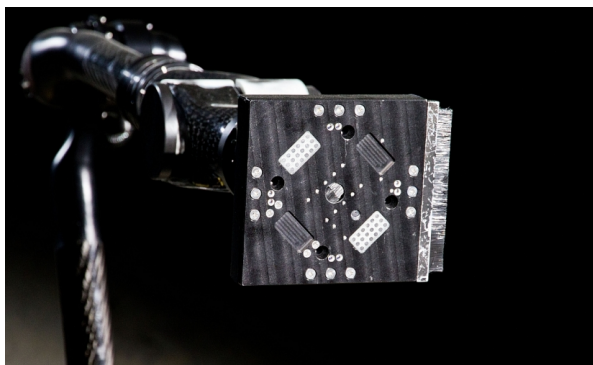
Limitations LUNARES has been a feasibility study. A major limitation of the robot team in LUNARES with respect to reconfiguration lies in the docking adapter which establishes a mechanical connection only. Furthermore, no generalised concepts for exploiting flexibility are part of the development approach: a classical, static robotic team constellation is basis for

the operational approach and a centralised communication and control architecture limits the scope of application of this robotic team.

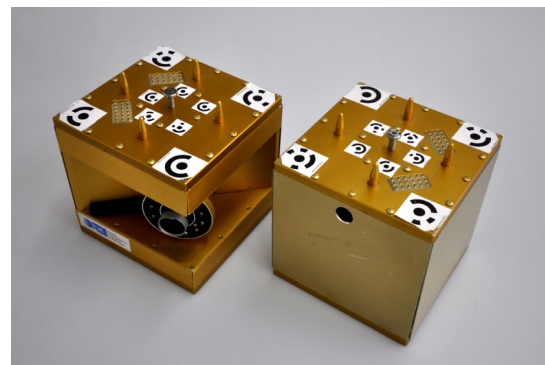
2.2.2 RIMRES - Reconfigurable Integration Multi-Robot Exploration System

Compared to LUNARES, RIMRES developed an incrementally more complex, and fully integrated approach to design and operate reconfigurable multi-robot systems. The project RIMRES uses reconfigurability as key design requirement. In this context, all robotic systems feature a standardised EMI, which has been developed by Wenzel, Cordes, Dettmann, et al. (2011). This EMI allows two agents not only to establish a secure mechanical link, but it also establishes the respective communication networks (for Ethernet and RS422) and a shared power bus. Furthermore, it has been designed to operate in the dusty lunar environment.

Electromechanical Interface The EMI as designed for the project RIMRES is gendered and comes in two variants: a female and a male interface. The female interface variant is also referred to as active EMI while the male is referred to as passive EMI. The female variant contains a motor-driven, gripper-like mechanism to establish a secure connection with a passive pin at the male interface. Figure 2.2 illustrates the two variants. Similar to the iSSI developed by Schervan et al. (2017) the EMI has been developed for space application. However, it is designed for the use in a planetary exploration mission on the lunar surface and not as satellite building block. Therefore, it features special design elements to account for an exposure to lunar regolith. According to the general classification scheme, e.g., as used by Liu, X. Zhang, and Hao (2016), this interface permits chain-based structures.



(a) Female (active) variant as part of the end effector (Source: DFKI GmbH).



(b) Male (passive) variant (Source: DFKI GmbH).

Figure 2.2: Male and female electromechanical interfaces (EMIs) in the project RIMRES.

The EMI embeds a power-bus system, which allows the transfer of energy. All robotic agents feature a power management system which allows to switch between power sources and balance the energy level. All systems connected with an EMI also share an Ethernet connection; a previous setup of the IP configuration is required. Visual markers have been added to the male interface to facilitate the visual servoing process.

The EMI is the key element to create reconfigurable multi-robot systems which are the basis for the research of this thesis. The application of an EMI enables modular robotic systems,

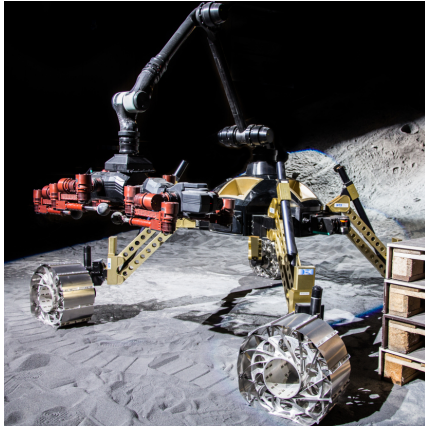


(a) Male interface of the back CREX (Source: DFKI GmbH).



(b) Female interface at the bottom of Sherpa to attach CREX (Source: DFKI GmbH).

Figure 2.3: Interfaces to connect the mobile systems.



(a) Variant 1: CREX attached to Sherpa's manipulator (Source: DFKI GmbH).



(b) Variant 2: CREX attached to the bottom interface of Sherpa (Source: DFKI GmbH).

Figure 2.4: Docking variants of the mobile systems CREX and Sherpa in the project RIMRES.

which are capable of creating larger composite systems, which can still act as if they were a single unit. Especially a shared communication and energy system is essential to compose a new capable robot from multiple reconfigurable robots.

In contrast to swarm-based systems, no fully distributed control approach is applied for the composite agents in RIMRES: each composite agent uses a centralised control approach. Nevertheless, the communication between agent relies on a distributed communication architecture. In contrast to an androgynous interface design, a female interface can only be connected to a male interface and vice versa. In general, interface compatibility will limit the number of feasible reconfiguration options for the robotic hardware; the implications are discussed in Chapter 3. Each female interface contains a camera which can be used for the visual servoing process.

Immobile agents A payload-item, in the sense of the implemented multi-robot system, is a general-purpose module or container with a standardised size of $0.15 \times 0.15 \times 0.15 \text{ m}^3$. It comes with a male interface on top and a female interface at the bottom - no interfaces are present at the sides. In principle, each payload-item is an individual agent with a processing unit (an ARM-based processing board is part of the design). Only power modules, however, have their own energy source.

Payload-items have been additionally equipped corresponding to their role in a space exploration mission: a battery payload item to serve as energy provider, or a camera payload item as sensing device.

Mobile agents The rover Sherpa as illustrated in Figure 2.4 is a wheeled-leg mobile exploration system, which features a manipulator on top and comes with an adaptive locomotion platform. The system has four bays to host payload items. Each bay offers a connection via a male EMI. The robot has one female EMI at the bottom so that systems with a male interface can be docked, e.g., as depicted in Figure 2.4b. This allows for a marsupial-like transport capability. The end effector of the manipulator is equipped with a female EMI as well. Therefore the rover Sherpa is the only system in the overall team of robots which can actively manipulate payload-items and perform reconfiguration of other systems.

The so-called Crater Explorer (CREX) is a six-legged mobile systems with a male EMI on its back (cf. Figure 2.3a). The interface on the back of CREX can be used to dock the system to Sherpa via the rover's bottom interface, and it can also be used to lift the CREX using Sherpa's manipulator as shown in Figure 2.4a. This leads to number of reconfiguration states which among other scenarios enable: (a) marsupial transport, (b) lifting systems to previously unreachable areas, and (c) attaching sensors or tools (here: a payload item or the six-legged robot as gripper device).

Operations The project RIMRES led to a space mission driven operation and allowed exploitation of the overall system capabilities using manual operation and semi-autonomous sequences. The project illustrated the feasibility and benefit of using an EMI and analysed the performance of automated docking of the scout to the exploration rover (see (Roehr, Cordes, and F. Kirchner 2014)). Hence, the project served as primary evaluation step for a reconfigurable multi-robot system which is composed of capable individual robotic systems, and a mixture of mobile and immobile systems. Operating the robots in RIMRES also verified the first iteration of the communication architecture which is described in Chapter 5.

Limitations Operation of the reconfigurable system in RIMRES is limited to the semi-autonomous, time-line based control which is triggered from the ground station. Direct manual control and visual feedback is used by an operator to perform safe operation, e.g., no autonomous simultaneous localization and mapping (SLAM)-based navigation approaches are used. As result of this project Roehr, Cordes, and F. Kirchner (2014) suggested the introduction of an organisation model for reconfigurable multi-robot systems to raise the autonomy level.

2.2.3 TransTerra & FT-Utah - Semi-Autonomous Cooperative Exploration of Planetary Surfaces

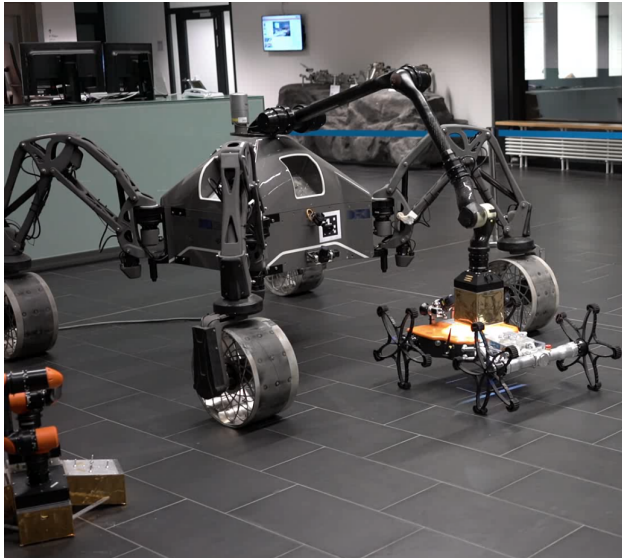
The project TransTerra in combination with a field trip in the scope of the project FT-Utah led to a reconfigurable multi-robot system which is capable of semi-autonomously performing exploration missions (cf. Chapter 6).

The robotic team in TransTerra is designed for the operation of a logistics chain to support the long term operation of terrestrial exploration. The team consists of the mobile systems SherpaTT and Coyote III which are depicted in Figure 2.5a. Further team members are a number of immobile systems: battery payload item, camera payload item, gas sensor array payload item, soil sampling payload item, a Differential Global Positioning System (DGPS) payload item, and an additional manipulator arm called Symmetrical Interface Manipulator (SIMA) (Brinkmann, Cordes, et al. 2018). A base camp serves as logistics hub, and supports the setup of an infrastructure based on placing payload items on base camps. Figure 2.6a shows a base camp. Each base camp has its own computing and communication unit and comes with five male interfaces to couple payload items. Each male interface has pins for guiding a docking process (a) and a central pin for establishing the mechanical connection (b), redundant (spring mounted) pins for the power and communication connection (c), and an optional set of visual markers (d). The rover's end effector also embeds the female variant as illustrated in Figure 2.6b. Furthermore, it comprises LEDS (a) and a camera (d) to support a visual docking process, the central pin of a male interface is received and locked in (c). The redundant set of connection pads (b) establishes the electrical contact with the corresponding pads on the male interface, while guiding holes (e) receive the guiding pins of the male interface. The redundancy of the pads permits 90° stepwise angular rotational difference when connecting a male and a female interface.

Apart of a successively improved electromechanical design, the major difference between Sherpa and SherpaTT is a change of the locomotion system, and the application of a significantly improved automation, parts of which will be presented in Chapter 6.

Operations Both systems SherpaTT and Coyote III can perform exploration fully autonomously, and both cooperate for a distributed SLAM-based mapping approach. A decentralised communication architecture described in Chapter 5 is the foundation for the distributed operation of the overall system. SherpaTT can manipulate payload-items, i.e., pick and place onto the Coyote III. Chapter 6 illustrates the performance of a fully integrated approach of an autonomously executed sample mission sequence. Furthermore, it describes the evaluation of the multi-robot system's abilities and identifies remaining shortcomings.

Limitations The increase of autonomous capabilities of the individual systems in the project TransTerra allows for significantly more advanced operations in comparison to the projects LUNARES and RIMRES. The overall team has continuously been improved, but remains an evaluation system of high complexity. For this reason, several limitations exist on low-level and on high-level posing a risk of failure. Failures can affect any automated operation so that autonomous robots require local failure handling routines, but might still fall back to direct operator interaction. Chapters 6 and Chapter 7 continue the discussion with detail.

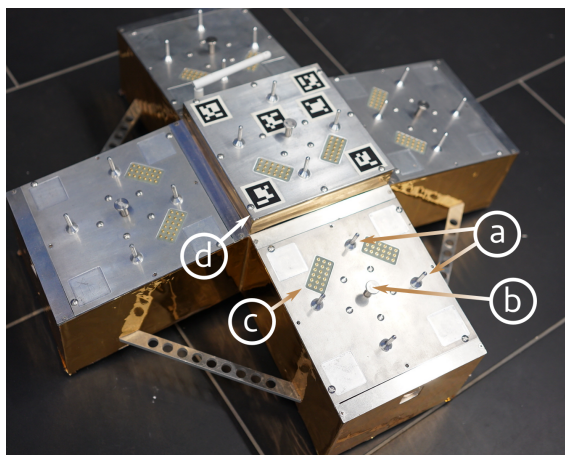


(a) The mobile agents of the reconfigurable multi-robot system in TransTerra while exchanging a payload between SherpaTT (left) and Coyote III (right).

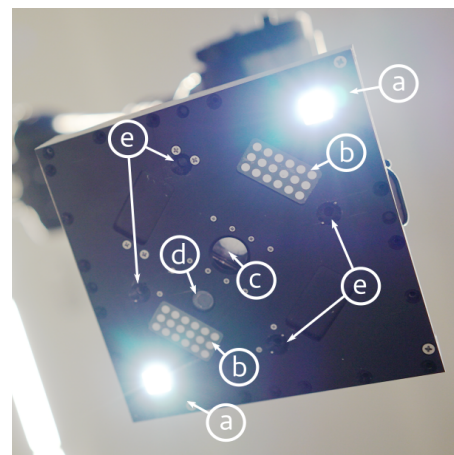


(b) The manipulator SIMA comprises two female interfaces for docking, here attached to a base camp. With connection to a power source, it can serve as agent with limited mobility, e.g., wander by attaching to male interfaces.

Figure 2.5: Mobile and immobile agents in the project TransTerra.



(a) Base Camp with five male interfaces.



(b) End effector with female interface.

Figure 2.6: Variants of electromechanical interfaces in the project TransTerra.

2.2.4 Categorisation

The reference systems developed in RIMRES and TransTerra show properties of multi-agent systems with high capabilities. Furthermore, they embed properties of self-reconfigurable robotic systems. To relate the reference systems to self-reconfigurable multi-robot systems, the categories structure, form factor, reconfiguration type as mentioned by Chennareddy, Agrawal, and Karuppiyah (2017) can be used. In principle, the EMI allows for lattice-based and chain-based structures. However, implementation and evaluation for systems in the projects RIMRES and TransTerra is limited to chain-based structures (see Chapter 3). Payload items have a default volume of 3375 cm^3 . They are the smallest units in the reference systems. Therefore, the reference systems are classified as macro-sized systems.

Furthermore, only deterministic and no stochastic reconfiguration approaches are applied. The basic payload-item design does not allow for locomotion, although specialised modules could be designed for that purpose. Modules are not self-propelled and require the use of an external manipulation device for relocation. In contrast, mobile agents such as rover and scout are based on traditional locomotion systems. According to Moubarak and Ben-Tzvi (2012) they fall into the category of mobile configuration change (MMC).

The suggested categorisation by Chennareddy, Agrawal, and Karuppiyah (2017) and Moubarak and Ben-Tzvi (2012) focuses on homogeneous building blocks for self-reconfigurable multi-robot systems, where modules and interfaces form a monolithic unit. In contrast, the reference systems used in this thesis are composed of heterogeneous robots. Each robot can be considered a module which is connectible via at least one EMI. As a result a module or rather a robot can have significantly different properties, so that the categorisations for self-reconfigurable multi-robot systems can be only partially applied. Furthermore, iBOSS and EMI are interfaces designed to be embedded as subsystems. Their application results in a broader applicability of reconfiguration concepts since they permit a heterogeneous system design in the first place. Hence, appendix A.2 lists further interface categorisation options for future consideration. In general, a new 'hybrid' category is required for mixing aspects of classical robotic systems with highly modular systems.

2.3 Future Robotic Space Missions

The state of the art in planetary space missions are single robotic systems. Despite the fact that international space agencies operate with multiple rovers on the same planet, cooperation between these system has not been targeted. With the consideration of building up infrastructure and creating habitats to prepare human presence, this paradigm has to shift in order to maximise the usage of resources. Figure 2.7 depicts the basic idea for incremental missions, where the actual design of hardware can evolve with the experience made in previous missions. All missions come with the challenge to maintain the available system. Current planetary space operations have to rely on ground operators for repair, which leads to a very slow and costly process. The dependency on earth-based maintenance, or even hardware deliveries should be minimised for future long-term space mission. Instead, ad-hoc in-situ maintenance and replacement of units which suffer from functional degradation should be aspired. Hence, the exploitation of redundancy and sharing of resources between robotic systems is one step towards long-term operations. An autonomous exploitation of the features of a reconfigurable

multi-robot system requires particular support, and the above mentioned examples show, that autonomy is not only desirable for nominal operations, but also for maintenance: the possibility to extend or refurbish existing hardware is an advantage, even more when a robot team can perform this process autonomously.

The modular design of reconfigurable multi-robot systems will initially increase the cost of development, but serves as key to new mission concepts and cost reductions in the long run by enabling reuse of components. Attributes of swarm-based systems can be exploited in the sense, that when robots share hardware resources, the risk of failure can easier be mitigated. In effect, a higher risk of failure per device can be tolerated, when the redundancy increases. Therefore, device development might be cheaper leading to a reduction to overall mission cost. This benefit increases when the technology for in-situ creation of repair structures and devices advances (ESA 2018a) is available for space missions and allows to save launch costs.

2.3.1 Autonomous Operation

The aim of increasing autonomy in space missions is the result of “pressure to reduce manpower during routine mission operations” (European Cooperation for Space Standardization 2008a). Truszkowski, Hinchey, and Rash (2004) favour swarm-based approaches to use as a solution and argue that an application of “large number of spacecraft provide[s] greater flexibility, reliability and autonomy than the more familiar spacecraft”. They judge, however, the verifiability of these approaches as major limitation, which still persists according to Vashev et al. (2012). While Truszkowski, Hinchey, and Rash (2004) refer to spacecrafts, their argument applies as well to planetary space robots. Bajracharya, Maimone, and Helmick (2008) see that even minimal mission success requires an increased level of autonomy: sample return missions will require ascent vehicles, which come with a limited lifetime, increasing the pressure to operate sampling collecting rovers with higher speeds. In this context, however, the communication channel limits direct control schemes. The Mars Science Laboratory has a data rate in the range of 500 to 32,000 bps (NASA 2018). Interaction between ground stations and robotic systems involves satellite communication and comes with long latency: to communicate a message between Earth and Mars takes an approximate time between 4 and 24 minutes (ESA 2018b). Additionally, using the Deep Space Network (DSN) (Washington et al. 1999) limits the communication to time windows and hence fragments a direct operation.

To maximise the use of deployed space robots the European Collaboration for Space Standardization (ECSS) already considers the availability of the ground station for space robots and

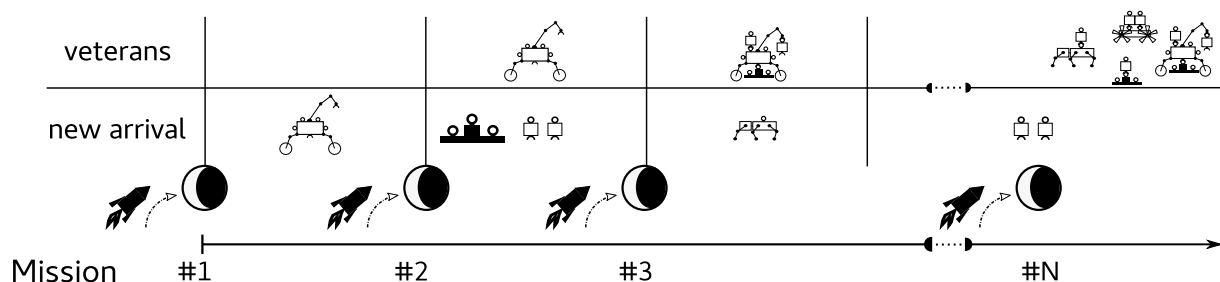


Figure 2.7: Schematic description of an incremental design of planetary space missions using reconfigurable multi-robot systems.

Table 2.4: *Autonomy levels for nominal mission operation according to the ECSS (European Cooperation for Space Standardization 2005).*

Level	Description	Functions
E1	Mission execution under ground control; limited onboard capability for safety issues	Real-time control from ground for nominal operations, Execution of time-tagged commands for safety issues
E2	Execution of pre-planned, ground defined, mission operations on-board	Capability to store time-based commands in an on-board scheduler
E3	Execution of adaptive mission operations on-board	Event-based autonomous operations, Execution of on-board operations control procedures
E4	Execution of goal-oriented mission operations on-board	Goal-oriented mission re-planning

requires such robots to tolerate a so-called “non-availability period”. The ECSS classifies autonomous operation into levels of autonomy according to Table 2.4 (European Cooperation for Space Standardization 2005, Section 5.7). Currently, no space rover is deployed on a remote planet which supports an autonomy level E4. Only the rover ExoMars is expected to show full autonomy (E4) with an average driving speed of 14 m/h (Gao et al. 2016; Winter et al. 2015).

The ECSS also accounts for the autonomy level for failure handling (see Table 2.5) which defines the need for fault tolerant systems with limited self-repair capabilities. The autonomous fault handling levels categorise system resilience (cf. Section 2.1), and clearly refer to reconfiguration as a major means to perform failure handling. Particularly modular and redundantly designed systems can support the demands of F1 to reconfigure systems and isolate faulty components. Furthermore, they can enable advanced (re)planning approaches for F2.

Increasing autonomy for robotic systems is part of the Global Exploration Roadmap defined by the International Space Exploration Coordination Group (ISEG) (International Space Exploration Coordination Group 2013). The ISEG also looks at affordability, capability evolution, and robustness among other aspects as mission driving principles and thereby implicitly stresses the relevance of reconfigurable multi-robot systems for future space missions. However, while commercialisation in space industry is already starting in this area, e.g., with the iBOSS system as basis for supporting reconfigurable satellites, an exploitation of the flexibility of this reconfigurable multi-robot system hardware still demands modelling and planning approaches, as well as a supporting software architecture that allows autonomous operation of such systems.

2.3.2 Dynamic Team Operations

Robotic exploration missions might require the participating robots to react to unforeseen events and changing priorities. Reconfigurable multi-robot systems are able to adapt to the demands of robotic operations by changing their morphologies and their overall coalition structure. Depending on the time and cost of reconfiguration of an overall team, dynamically changing teams enable new operation schemes. Their flexibility maintains alternative ways to achieve mission goals under adversarial circumstances. In effect, they offer potential for a safer autonomous operation compared to monolithic systems.

To support long-term missions, as well as dynamically changing requirements, new (or dynam-

Table 2.5: *Autonomy levels for failure handling according to the ECSS (European Cooperation for Space Standardization 2005).*

Level	Description	Functions
F1	Establish safe space segment configuration following an onboard failure	Identify anomalies and report to ground segment, reconfigure on-board systems to isolate failed equipment or functions, place space segment in a safe state
F2	Re-establish nominal mission operations following an on-board failure	As F1, plus reconfigure to a nominal operational configuration, resume execution of nominal operations, resume generation of mission products

ically created) agents have to be included and accounted for in the overall multi-robot system. A new agent can add capabilities to the overall system. This can only be exploited, when the given software infrastructure and high-level planning mechanisms can account for these new capabilities. Hence, interoperation and extensibility of a reconfigurable multi-robot system depend upon a significant level of standardisation. Additionally, model-based development approaches and model-based reasoning can support a generic infrastructure and automation approaches as described in Chapter 3: the EMI developed in RIMRES and TransTerra is only one interface, and a reconfigurable multi-robot system might also use multiple variants.

Dynamically changing coalitions are observable in two situations. Firstly, as result of a coalition structure change, e.g., triggered to fulfil changing functional needs or to address safety issues. Secondly, when the overall number of available agents changes; either through loss or addition of individual agents. The requirements arising from both variants demand a transparent mechanism of adding and removing robotic systems. The coalition structure might change disruptively, i.e., leaving some robots unpowered. These requirements and an intended application for space exploration in unknown or partially known environments suggest applying a distributed communication approach. This approach comes with the benefit of enabling local and self-sustained operative coalitions: a subteam of agents can remain operational independent of the communication to other agents.

2.4 Defining Reconfigurable Multi-Robot Systems

Based on the experience from working with the reference systems, this thesis defines a formal framework to deal with reconfigurable multi-robot systems and to support the autonomous operation of robotic (space) missions. This section provides the basic notation, definitions and the underlying assumptions regarding reconfigurable multi-robot systems. The notation builds on the formalisms found in coalition games (Weiss 2009). In particular, the agent-type representation is based on the representations developed by Shrot, Aumann, and Kraus (2010) and Ueda et al. (2011).

As already mentioned reconfiguration can take place on different levels in hardware and software. Since the focus of this work is on physical agents, the level of granularity is chosen correspondingly. Therefore, the lowest level of granularity is a physical agent which cannot be separated further into two or more physical agents. This agent is denoted by **atomic agent**.

Definition 2.1 (Atomic agent). An **atomic agent** a represents a monolithic phys-

ical robotic system, where $A = \{a_1, \dots, a_{|A|}\}$, is the set of all atomic agents, and $a \in A$ or equivalently $\{a\} \subseteq A$.

Note that a physical agent representing an atomic agent still contains subsystems. They are, however, inseparable parts of the physical agent.

Reconfigurability and a standardisation of connection interfaces opens the opportunity for combining two or more atomic agents. A composition from two or more atomic agents is referred to as **composite agent**. Note that the use of the join operator \cup in the following Definition 2.2 aligns well with the actual physical join operation of atomic agents. This allows for an intuitive representation.

Definition 2.2 (Composite agent). A mechanically coupled system of two or more atomic agents is denoted by **composite agent** $CA = a_i \cup \dots \cup a_j = \{a_i, \dots, a_j\}$, where $a_i, \dots, a_j \in A, |A| \geq |CA| > 1$.

Additional ways of coupling two agents can be considered, e.g., electromechanical or thermoelectromechanical. However, Definition 2.2 requires that a composite agent has at least a mechanical connection between its atomic agents.

Figure 2.8 illustrates the approach to agent composition. To facilitate the understanding of the following definitions, here an application example: A mobile robot (atomic agent m) can share its power source with other robots, but it has no camera. After attaching an unpowered atomic agent c which has one camera as a subsystem, the composite agent $\{m, c\}$ is not only equipped to take images. It can now move to any location and take images - a functionality neither of the atomic agents m or c provides.

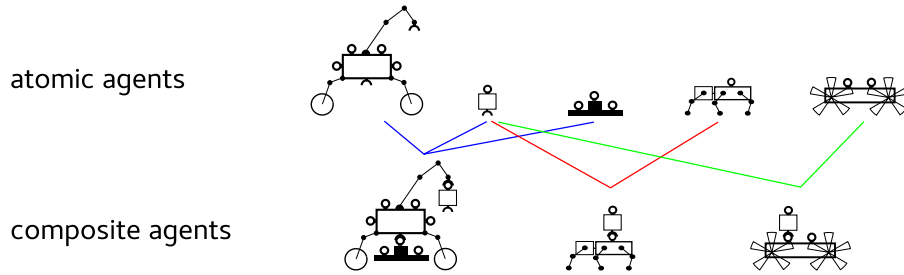


Figure 2.8: An available set of atomic agents and a subset of composite agents that can be formed by combining different atomic agents.

Combinatorial explosion is one of the main challenges to deal with when considering a reconfigurable system with a large number of atomic agents. One means to reduce the effects of combinatorial explosion is typing. Agent typing allows dealing with same typed agents using homogeneously formed partitions of an overall set of agents.

Definition 2.3 (Atomic and composite agent type). The type of an atomic agent a is denoted by \hat{a} and equivalently for a composite agent CA the type is denoted by \widehat{CA} . The set of all atomic agent types is denoted by $\theta(A) = \{1, \dots, |\theta(A)|\}$, with the corresponding type-partitioned sets of agent instances $A^1, \dots, A^{|\theta(A)|}$, where $A = A^1 \cup \dots \cup A^{|\theta(A)|}$.

The concept of a (general) agent wraps the concepts of atomic and composite agents. Henceforth, in this thesis, the term agent is equivalently used to the term general agent.

Definition 2.4 (General agent). Any subset $GA \subseteq A$, where $GA \neq \emptyset$ forms a physical coalition is denoted by **general agent**. A (general) agent has a corresponding atomic agent type partitioned set of agent instances $GA^1, \dots, GA^{|\theta(A)|}$, where $GA = GA^1 \cup \dots \cup GA^{|\theta(A)|}$.

Definition 2.5 (General agent type). The type of a (general) agent GA is denoted by \widehat{GA} . A general agent type \widehat{GA} is represented as a function $\gamma_{\widehat{GA}} : \theta(A) \rightarrow \mathbb{N}_0$. The function $\gamma_{\widehat{GA}}$ maps an atomic agent type \hat{a} to the cardinality $c_{\hat{a}}$ of the type partition of \widehat{GA} , such that $c_{\hat{a}} = |GA^{\hat{a}}|$. Equivalently to $\gamma_{\widehat{GA}}(\hat{a}) \geq 1$ the following notation will be used: $\hat{a} \in \widehat{GA}$, and $\hat{a} \notin \widehat{GA}$ for $\gamma_{\widehat{GA}}(\hat{a}) = 0$.

A general agent type is also represented as a collection of tuples relating agent type and cardinality: $\{(\hat{a}_0, c_{\hat{a}_0}), (\hat{a}_1, c_{\hat{a}_1}), \dots, (\hat{a}_n, c_{\hat{a}_n})\}$.

Definition 2.6 (Constructible agent types). The set of all constructible general agent types from a set of atomic agents A is denoted by $\Theta(A)$; it represents the collection of all general agent types that are found in the powerset of all agents \mathcal{P}^A .

Two representation options for a collection of atomic agents or rather an agent pool exist. Firstly, the set of atomic agents as plain description. Secondly, the representation as a general agent type. The general agent type representation is more compact since it defines only the number of atomic agent instances per agent type.

Definition 2.7 (Agent pool). An **agent pool** A denotes a set of atomic agent instances. It can equivalently be represented by a general agent type \widehat{A} , such that $\forall a \in A : \gamma_{\widehat{A}}(\hat{a}) = |A^{\hat{a}}|$.

To execute robotic missions, atomic agents from an available agent pool will be assigned to particular tasks. However, if multiple atomic agents of the same type exist and equal start conditions hold for these atomic agents, multiple equivalent assignments of atomic agents to a task are possible. For that purpose, requirements for atomic agents will be defined by so-called roles, which act as correctly typed placeholders for instances of an agent type.

Definition 2.8 (Atomic agent role). An **atomic agent role** $r^{\hat{a}}$ represents an anonymous agent instance of an atomic agent type \hat{a} . A set of agent roles with a one-to-one mapping to an agent pool A is denoted by $r(A)$.

Given an overall set of atomic agents, various reconfiguration states of the overall systems are possible. These reconfiguration states result from forming different sets of composite agents, but always with the restriction of the overall available set of atomic agents. In the field of multi-agent systems and particularly characteristic function games (Weiss 2009, p. 332) this leads to so-called coalition structures. A coalition structure represents the set of active atomic and composite agents that form a reconfigurable multi-robot system.

Definition 2.9 (Coalition structure). A coalition structure of an agent set A is denoted by CS^A and is represented by a set of disjunct general agents $CS^A = \{GA_0, \dots, GA_n\}$, where $GA_0 \cup \dots \cup GA_n = A$, and $i, j = 0, \dots, n, \forall i, j, i \neq j : GA_i \cap GA_j = \emptyset$.

Composite agents result from the combination of atomic agents. Definition 2.10 separates the current (realised and physically assembled) set of general agents in a coalition structure from the (virtual) set of agents, which can be formed from the set of atomic agents.

Definition 2.10 (Operative and dormant agents). Let the current state of a reconfigurable multi-robot system be described by a coalition structure CS^A . Then all general agents $GA \in CS^A$ are referred to as **operative agents**, and complementary, all general agents $GA \in \mathcal{P}^A \wedge GA \notin CS$ are referred to as **dormant agents**.

The previous definitions look at a reconfigurable multi-robot system as a collection of agents, and consider pairing and coalitions only at this level of modularity. Chapter 3.3.1 accounts for the physical interfaces as subsystems of an agent to analyse the feasibility of a composite agent. A reconfigurable multi-robot system can form composite agents in different ways depending upon the compatibility of these interfaces. The scope of the formal description based on a set-theory description covers what is denoted by agent space, which is a restricted view onto link space.

Definition 2.11 (Link space). **Link space** denotes a graph-theory view to analyse the structure of a reconfigurable multi-robot system. In link space a reconfigurable multi-robot system is represented by an undirected graph $G = (V, E)$, where each vertex $v \in V$ maps to an atomic agent's interface and an edge $e = (u, v)$, $u, v \in V$ represents the existing connection between two interfaces.

Definition 2.12 (Agent space). **Agent space** denotes the set-theory based view to a reconfigurable multi-robot system. The preceding definitions establish atomic, composite and general agents, as well as coalition structures. These definitions do not detail the connections between any two agents. A feasible composite agent implies, however, the existence of a connected graph in link space for its composing set of atomic agents.

2.4.1 Assumptions

As illustrated in the previous sections, a large spectrum of reconfigurable multi-robot systems exists. Most often, fully distributed control approaches apply, due to the use of swarm-based systems. The definition of the general agent already reflects one important design consideration and design philosophy for thesis, which relaxes this apparent requirement for distribution. Instead of enforcing distributed control approaches at all system levels, centralised control approaches for locally autonomous and self-sustained operation of agents are permitted and feasible. This implicitly allows an atomic agent to act as a temporary 'master' in a master-slave architecture. When forming a composite agent, for instance, a single atomic agent in this formation acts as master, which is able to control all other attached atomic agents. In

effect, each general agent represents as a single-minded (collaborative) agent. The distribution of the overall agent system is still maintained by an appropriate design of the operational infrastructure.

Assumption 2.1 (*Individual agent*). *Each atomic and composite agent comprises a central controller and thus represents an individual, single-minded agent.*

Generally, two atomic agents can connect via multiple interfaces. This thesis, however, assumes limited connectivity and does not consider geometrical constraints. The application of this restriction intends to set this thesis' focus onto the identification of essential needs for modelling and automation of reconfigurable multi-robot systems.

Assumption 2.2 (*Single link*). *A mechanical coupling between two atomic agents can only be established through two and only two compatible coupling interfaces.*

In principle Definition 2.3 allows a single agent to have multiple types. However, this thesis assumes a single characterising atomic agent type. Meanwhile, one agent type can still inherit the properties of a parent type.

Assumption 2.3 (*Single agent type*). *An agent can be mapped to a single agent type only.*

Assumption 2.4 (*Agent type inheritance*). *An agent type can inherit the properties of another agent type.*

When two or more atomic agents form a composite agent, they join their set of resources. In principle, geometrical restrictions might apply to reuse the set of resources effectively. However, this thesis initially assumes that resources of each atomic agent are shared without restriction within a composite agent.

Assumption 2.5 (*Resource usage*). *A composite agent can reuse the subsystems of its composing atomic agents.*

2.5 Discussion

Modular systems introduce the change of morphology as a significant problem especially for a large number of modules. The focus of many researchers in the area of modular reconfigurable systems is therefore on shape-shifting. Nevertheless, shape-shifting is an intermediate step for the automated operation of a reconfigurable multi-robot system. An automated operation requires changing the morphology of a reconfigurable system to achieve the best suited coalition structure for the current task. Baca et al. (2014) motivate the use of coalition structure generation in the context of reconfigurable modular systems. They did not, however, investigate further towards a fully automated planning approach. In contrast, this thesis formalises the composability of a reconfigurable system and introduces corresponding definitions. The coalition structure generation as suggested by Rahwan, Ramchurn, et al. (2009) is also part of a corresponding organisation model (see Section 3.4) which finally permits planning with reconfigurable multi-robot systems.

The definitions and assumptions introduced in the previous section come with limitations. These limitations are the result of trading a reduction of complexity against the precision and generality of the representation. An agent pool or general agent type defines the composite type using agent space. Agent space accounts for the combination of agents without considering the details on how the connection between agents is established. This abstraction level corresponds to the previously mentioned choice of granularity for dealing with modularity. Hence, this basic set of definitions does not account for any type of interface and compatibility. Therefore, it can represent feasible and infeasible composite agents without further distinction.

In contrast to agent space stands a representation of an agent coalition in link space. The distinguishing factors between agent types include the physical connections between combined atomic agents. Establishing these links follows constraints. Coupling interfaces can be generated and cannot be arbitrarily paired. Hence, the model requires extension to tackle the issue of connectivity in a reconfigurable multi-robot system. Chapter 3 describes an organisation model which deals with this limitation. Note here already that the model does not solve the general issue of creating a morphology which induces negative side-effects. For example, if a module is placed in front of a camera, the camera's field of view can be severely constrained. Effectively, the functionality of the camera might be disabled.

2.6 Summary

This chapter motivates the application of reconfigurable multi-robot systems for space missions. It highlights the theoretical advantages of reconfiguration in the context of a framework of so-called strategic flexibility. This framework considers support for defensive actions as well as offensive actions. For space robotics especially the opportunity for an increase of defensive actions is attractive. A higher resilience of robotic missions can already be seen in swarm-based systems, which expose self-repair and self-healing capabilities. Further potential lies in the performance of offensive actions. In general, reconfigurability introduces new flexibilities for future robotic missions. It opens the opportunity for incremental mission designs and the operation of robotic systems with a controllable degree of efficacy, efficiency and safety.

This chapter presents the set of reconfigurable multi-robot systems which have been used throughout the development of this thesis. Three different teams of robots served as platform for the empirical evaluation of automation approaches for reconfigurable systems. Their successive development allowed developing and validating the theoretical foundations. The basic notation and core assumptions for dealing with reconfigurable multi-robot systems are introduced in this chapter. Hence, this chapter formalises essential concepts of this thesis. This formal framework is implemented in an organisation model, which is presented in Chapter 3. The following chapters successively augment the formal framework, which represents the basis for planning with reconfigurable multi-robot systems.

3

Organisation Modelling

Disclaimer *This section introduces and expands on works which have been published in (Roehr and Hartanto 2014, 2015; Roehr and F. Kirchner 2016).*

An organisation model in the context of this thesis represents the formalised state description for a reconfigurable multi-robot system. This description is fundamental to quantify the main characteristics of an organisation composed of multiple agents and to reason about the organisation in general. This chapter therefore introduces a dedicated model: Model for Reconfigurable Multi-Robot Organisations (MoreOrg).

Multiple robots that operate towards a common goal and which adhere to rules of interaction can be perceived as a single, possibly intelligent organisation. In this thesis, a robotic mission represents the common goal and it defines functional requirements in combination with spatial, temporal and resource constraints; details on the mission specification are provided in Chapter 4. An organisation's ability to perform a mission directly depends on its structure, since the activation of an organisation's functions depends on the current resource structure. The structure of a multi-agent organisation can follow a broad set of paradigms including but not limited to hierarchical, holarchical, coalition-based, team-based, and coalition-based structures (Horling and Lesser 2004). The choice of the applied paradigm for the organisation structure depends on the target domain, and practical considerations. Each paradigm demands a control infrastructure or more generally, the support of the participating agents. This support can be provided in the form of communication and coordination protocols which are adapted for a particular control structure. Out of this set of paradigms, coalitions have been of particular interest for game theorists. MoreOrg builds upon the existing theoretical foundation to model the composition of physical agents, or rather composite agents. The composite agents which are dealt with in this thesis are a special type of coalitions since members of a coalition are physically linked. An organisation of reconfigurable robots can adapt the

linking and effectively its coalition structure. This enables the exploitation of synergies and continuous optimisation of organisation properties. Game theorists do not make a distinction between a tight coupling and a loose coupling. Hence, MoreOrg advances the existing theoretical approaches for the domain of reconfigurable multi-robot systems. The ability to form composite agents is the key attribute of MoreOrg's modelling approach to manage reconfigurable multi-robot systems and the distinctive element compared to other approaches.

A single robot can be described by its hardware equipment including sensors and actuators and its software components. The combination of hardware and software defines the capabilities and overall functionality of the robot. Based on its capabilities a robot can perform a set of assigned tasks, but the general applicability of a robot to a particular task does not only depend upon its available set of resources. It also depends upon the robot's physical availability. This observation becomes even more relevant for reconfigurable multi-robot systems where multiple robots can merge to form a composite agent. When two or more agents form a composite agent, the participating and previously available agents will become unavailable and dormant, while a new and possibly more capable operative agent appears.

MoreOrg formalises the construction of composite agents using a model-based approach. It accounts for the availability and the merging of resources to identify possible synergies and resulting superadditive effects. In effect, the organisation model is the basis for reasoning with a reconfigurable multi-robot system. According to the formal description in Chapter 2.4 the organisation model MoreOrg provides a static description of atomic agents, while allowing for a dynamic description of composite agents. The approach covers the characterisation of the organisation structure based on the availability of functionality and existing safety properties. This model is exploited for a corresponding planning approach as described in Chapter 4.

The implementation of MoreOrg touches multiple fields of research, and the presented approach mixes elements of organisational, multi-agent, multi-robot, planning research and system reliability theory. The organisation model MoreOrg thereby closes a research gap by introducing a knowledge-based modelling approach for physically reconfigurable multi-robot system.

Section 3.1 introduces related approaches and the author's preparatory works which contributed to the development of MoreOrg. The background section is followed by a description of the ontology-based modelling approach of MoreOrg in Section 3.2. An essential goal of the model is to identify agents which provide the required functionality to contribute towards an organisation's goals. Section 3.3.1 therefore details the algorithm to identify feasible composite agents. Section 3.3.2 outlines the concepts of functionality support and an essential mapping between an agent's resource structure and its functionality. A description of the organisational state requires the development and application of suitable metrics, which are presented and motivated in Section 3.5. This section also includes the description of policies and heuristics to infer properties of composite agents. The chapter concludes with a discussion in Section 3.6 and a summary in Section 3.7.

3.1 Background

Robotics research requires an interdisciplinary approach and the interaction of multiple fields of research. This likewise holds for the development of an organisation model of reconfigurable

multi-robot systems. The organisation model MoreOrg brings together approaches which are found in the areas of organisational research, knowledge-based reasoning, multi-agent and robotics research. Additionally, the development of the model led to the identification of suitable approaches and existing algorithms in order to deal with the combinatorial challenge encountered. The following sections highlight the related research which is the basis for the design of the organisation model.

3.1.1 Organisation Models

An organisation is defined as “an organised group of people with a particular purpose, such as a business or government department” (Oxford University Press 2018). This definition can easily be mapped to multi-agent organisations representing an organised group of agents with a particular purpose. Most relevant in this context is the purpose of the organisation which motivates the existence of the organisation in the first place.

Organisation models can be found in the areas of organisation management, multi-agent research and robotics research. The main intention of existing research approaches lies in the formalisation of goal driven organisations, which are formed by a number of virtual or physical agents, e.g., software agents, robots, or humans. Existing organisation models do not account for changing agents at a microscopic level, i.e., they do not change or adapt agent’s internals. However, agent behaviour can be adapted or rather enforced through the implementation of organisational structures and norms, such as interaction rules.

OMNI The organisation modelling approach Organisational Model for Normative Institutions (OMNI) (V. Dignum 2009) is a formal approach from organisational research, and it allows to check the conformance of the behaviour of agents with a set of organisational rules. The research around OMNI generally focuses on “a human-centred perspective, where norms may be violated” (Putten et al. 2009). Correspondingly, an agent rather represents a real person in this context. The formalisation of a model allows to monitor the actual work practice in a (human) organisation and compare it with the intended agent behaviour, so that behaviours outside the norm can be penalised or sanctioned. The need for such a strong external organisation control, arises from a strong autonomy assumption of agents, i.e., each agent can operate autonomously and with its own agenda, yet, has to follow the organisation’s rules.

Therefore OMNI uses information about non-conforming behaviour to sanction agents and thereby control or rather enforce the agents’ contribution to an organisation’s goal. OMNI is the combination of two separately developed models: OperA (V. Dignum, F. Dignum, and Meyer 2004; Putten et al. 2009) and HarmonIA (Vázquez-Salceda and F. Dignum 2003). OperA represents a top-down modelling approach to describe structure and goals of an open agent society, while HarmonIA is a formal framework to implement norms in organisations which are participating in online marketplaces and using electronic transactions (F. Dignum 2001).

Through the combination of these two existing models, OMNI inherits an organisational, an ontological and a normative dimension (also referred to as deontic dimension by Hübner, Sichman, and Boissier (2004)). OperA provides the organisational dimension which can be further split into three models: (1) an organisation model which captures the organisational structure using roles and interaction templates called scene scripts, (2) a social model which defines the responsibilities that come with a role and the required capabilities for a role, and (3) an

interaction model to represent bilateral agreements between agents to define their pairwise interaction. The normative dimension is originally found in HarmonIA, which defines how abstract norms can be activated in an agent society. HarmonIA's approach follows an iterative approach to concretise abstract norms, first into rules and policies, and finally into concrete procedure implementations.

The usage of ontologies is an integral part of OperA and HarmonIA and V. Dignum (2009) see the integral use of ontologies as advantage over other approaches, where ontologies are only seen as external components. Hence, OMNI comprises an additional ontological dimension to establish the common understanding in an open agent architecture, i.e., the ontological dimension defines how and what can be communicated between agents so that a knowledge exchange can be achieved.

MOISE+ Hübner, Sichman, and Boissier (2004) developed Model of Organisation for multi-agent SystEms (MOISE+) as an organisation model with a focus on the reorganisation capability of multi-agent systems. Their design philosophy is based on three dimensions of an organisation: (1) structural, (2) function, and (3) deontic (normative). Hübner, Sichman, and Boissier assume an organisation which imposes constraints on its member agents, and each organisational dimension brings its own set of constraints or restrictions: the structural dimension, e.g., defines how the organisation is divided into groups, or what kind of roles agent fulfil. The functional dimension involves behavioural templates or rather plans, which can be followed by an agent to perform a task. The deontic dimension defines a set of social interaction rules, which the agents have to follow during operation. The organisation thrives towards a goal, and the combination of restrictions in the three dimensions controls the observable organisational behaviour.

With this setup OMNI and MOISE+ have a very similar decomposition, but Hübner, Sichman, and Boissier focus on reconfiguration of the organisation: an optimal team structure depends on the environmental context and the goal. Hence, reconfiguration can allow to adapt and thus optimise the agent team structure.

A transition from one team structure to another can be planned or unplanned: planned transitions can be triggered in a top-down fashion by an external operator, or they can be scheduled for a specific time. Hübner, Sichman, and Boissier require planned transitions to follow a previously defined and therefore static reorganisation pattern, while unplanned transitions have to be dynamically controlled by agents.

Generally however, a transition follows a phase pattern for organisation change, originally defined by So and Durfee (1993) for distributed networks. The pattern consists of: (1) a *monitor* phase, (2) a *design* phase, (3) an *evaluate and select* phase, (4) and an *implement and execute*. This pattern is similar to the four stage model of teamwork by B. M. Dunin-Keplicz and Rineke Verbrugge (2010) with the corresponding stages: potential recognition, team formation, plan formation, and team action. In other cases reconfiguration can be also viewed as preparation of the team for a new task. The reorganisation process in MOISE+ itself requires forming a predefined group structure: one agent has to adopt the role of the so-called *OrgManager* in order to organise the overall reconfiguration. The reconfiguration group also requires at least one agent to take over the *Designer* role, in order to analyse the current status of the organisational structure, and suggest a potentially better structure. By encoding the required tasks for reconfiguration as agent roles, Hübner, Sichman, and Boissier identify the core elements

of a general reconfiguration recipe for distributed teams. They illustrate their approach using a robot soccer simulation, i.e., involving 11 agents, and apply Q-learning to identify the best reconfiguration policy for a game, where the opponent maintains a static team organisation. They show, however, no experiments with real robotic systems.

OMACS In the area of robotics Organisation Model for Adaptive Computational Systems (OMACS) is another approach for designing an organisation model presented by DeLoach (2009) and Deloach, Oyen, and Matson (2008). The main concepts in OMACS are goals, roles, agents and capabilities. OMACS uses a capability-based representation for a role, i.e., a role is defined by a set of capabilities, and the quality of an agent's capability can be quantified using values normalised to $[0, 1]$. In the same way DeLoach quantify an agent's ability to fulfil a role based on its capabilities. Independent of the agent definition, OMACS accounts for a degree of suitability of a role to achieve a goal, and by combining the information about roles, agents and goals OMACS allows to quantify the quality of an overall agent assignment with respect to goals.

The model assumes atomic capabilities without composition, and the value normalisation to $[0, 1]$ restricts the quantification, e.g., for a qualification of capability, to a single dimension. The quality of an agent's capability has therefore (initially) unclear semantics, which limits the applicability of the approach in practical applications.

Similar to the deontic dimension in MOISE+, DeLoach suggest the use of behaviour policies to control the cooperative behaviour of agents. In OMACS an organisation designer can explicitly define reorganisation rules. For instance to specify if and how one agent can replace another agent once the latter becomes unable to fulfil a role. An application of runtime reorganisation has been shown with three real robots, and a single laptop agent by Zhong and DeLoach (2011). The robots perform dynamic reorganisation to maintain a general patrolling task either after an agent fails to communicate or after the degradation of a capability which is required for the patrolling task. The scenario has been verified in simulation using eleven robots.

Summary OMNI and its comprising modelling approaches are frameworks for the specification and design of open multi-agent organisations. They implement a rigid formal frame for autonomously acting agents, which are mostly human, in order to achieve organisation goals. This is a valid approach for organisations with low control on the internal design of agents, e.g., as it is true for human agents.

The main missing element however in OMNI is an explicit accounting for the dynamics of change in an organisation. This is done by MOISE+. It implements a pattern to control the reconfiguration process. Therefore it can continuously optimise the organisation to achieve the organisation's objectives. Similarly, and applied to robotics, OMACS sets the main focus on the quantification of the potential of abstract roles and agents to contribute towards an organisation's success. The usage of this information allows to improve the team structure to increase the likelihood of an organisation's success. Generally, however, organisation modelling approaches have been limited to reconfiguration as reassignment of tasks to systems. They do not account for any type of superadditive effects.

3.1.2 Organisational Metrics

Risk management for space related projects (European Cooperation for Space Standardization 2008b) comes with a need to define reliability and probability of survival of systems. The general workflow defined in the corresponding ECSS standard consists of the following steps: (1) defining a risk management policy, (2) prepare a risk management plan, (3) identify risk scenarios, (4) assess the risk, (5) decide if the risks may be accepted, (6) reduce the risks, and finally (7) recommend acceptance. Risks may, or may not be accepted without any changes, but an initial estimation of risk is required. Hence, risk management comprises identifying the likelihood of critical conditions and estimating the severity of this critical condition. A resulting classification scheme for individual missions or projects can be developed to allow a categorisation of issues into fuzzy risk classes: very low, low, medium, high, very high.

The development of organisational metrics can help to proactively support the identification and management of risks and supports the automated and systematic estimation based on an organisation model. Hence, the development of an organisation model can contribute to the common goal, to minimise risks “in a systematic, proactive, comprehensive and cost effective manner” (European Cooperation for Space Standardization 2008b, p.6).

Organisation properties have to be quantified to describe the state of an organisation’s structure. In the context of OMACS, DeLoach and Kolesnikov (2006) apply a model checking framework to compute design metrics and quantify an organisation’s flexibility at design time with respect to a given goal. To quantify flexibility they assume a full exploration of the state space of an organisation. An organisation state can be reached through goal directed activities, reconfiguration and change of executing agents after failure. Flexibility is therefore a global metric associated with an organisation. In contrast, this thesis suggests dynamic organisation properties as state metrics such as safety. The characterisation of the system’s safety, in the sense of probability of survival, is already formalised in the area of reliability theory (Meyna and Pauli 2010; Rausand and Høyland 2009). Functional decomposition into serial and parallel subsystems serves as one approach in reliability theory to compute a system’s probability of survival. Hence, MoreOrg combines the knowledge about the resource composition of agent types with reliability theory to compute a safety metric for reconfigurable multi-robot systems.

3.1.3 Knowledge-based Reasoning

Semantic technologies are applied across multiple domains, e.g., bioinformatics, organisation research and robotics research. All require solutions to manage knowledge in a generic way. The application of semantic technologies involves the construction and use of so called knowledge-based systems, which can store knowledge fragments, relate them and derive new information from existing knowledge by applying automated reasoning techniques. The application of semantic technologies allows agents to gather and store knowledge in generic ways. Furthermore, they establish a standard to share and communicate knowledge between agents. As already mentioned, in the context of organisation modelling, this standardisation is an important element of open agent architectures, which can be in parts established by the use of ontologies.

Ontologies represent knowledge, e.g., by defining concepts, concept relations and properties, and thus establish a common understanding even about previously unknown concepts. To

Table 3.1: Language features of the DL *SHOIQ* (Krötzsch, Simancik, and Ian Horrocks 2012).

Feature Label	Description
\mathcal{S}	Stands for the base language which among other things allows for concept intersection, complex concept negation and transitive roles
\mathcal{H}	role hierarchies
\mathcal{O}	nominals
\mathcal{I}	inverse properties
\mathcal{Q}	qualified cardinality restrictions

represent ontologies the World Wide Web Consortium (W3C) recommends Web Ontology Language 2 (OWL 2) (W3C OWL Working Group 2012) as an international standard. The intention of this representation standard is to enable to computational processing of knowledge. OWL 2 is strongly related to Description Logic (see next paragraph) and comes with different profiles, i.e., sublanguages with different expressiveness. Depending on the need of a domain, e.g., for domains with a large number of instances or a large number of classes, different profiles might be suited. Generally however, only the OWL 2 Direct Semantics (Motik, Patel-Schneider, et al. 2012) is a known decidable logic (Motik, Grau, et al. 2012, Table. 10).

Tool support Multiple tools exists to support the development and design of ontologies, enable reasoning or facilitate the application of reasoners. The OWL API (Horridge and Bechhofer 2011) and the graphical user interface Protégé (Stanford Center for Biomedical Informatics Research 2015) represent two popular tools. OWL API not only defines a general interface to manage ontologies in conformance to the W3C standards. It also provides an implementation for an in-memory representation of ontologies. Hence, OWL API is an essential tool for computational processing of ontologies. The graphical user interface Protégé facilitates the manual inspection and design of ontologies. Tools for managing ontologies are predominantly developed and available for Java. In contrast, the robotics domain often requires or prefers libraries written in C or C++ due to its need to interface with hardware. Although the graphical user interface Protégé has been used to develop and maintain the ontologies in this thesis, the final implementation of the organisation model relies on the thesis author’s C++-based implementation of the OWL API in order to maintain a homogeneous infrastructure. The implementation does not support the full feature set of the Java-based API, but it provides special support enable the model-based reasoning based on qualified cardinality constraints.

Description Logic The OWL 2-based ontology representation is strongly related to Description Logic (DL), and both representations use an overlapping terminology, e.g., a *class* in OWL 2 refers to a *concept* in DL, *instances* of concepts or classes are uniformly called *individuals*, and a *property* in OWL 2 corresponds to a *role* in DL. DL comes with a corresponding separation in a so-called terminological box (TBox) to store concepts and associated properties, and an assertional box (ABox) (Bader et al. 2007; Krötzsch, Simancik, and Ian Horrocks 2012) to describe individuals. Although not formally required this separation intends to facilitate the automated processing. A DL represents a trade-off between expressivity and reasoning complexity (Krötzsch, Simancik, and Ian Horrocks 2012), and reasoners have different capabilities to support inference for a so-called knowledge base which is encoded with a DL. The DL reasoner FaCT++ (Tsarkov and Horrocks 2006), for example, supports the DL *SHOIQ*, where Table 3.1 explains the language features. OWL 2 supports the use of *SROIQ^(D)* (Ian Horrocks,

Kutz, and Sattler 2006; W3C OWL Working Group 2012), where *SROIQ* is a further extension of *SHOIN*, which was used for the previous Web Ontology Language (OWL) 1 profile DL. *SROIQ* adds expressivity and improves the practical usability, e.g., Ian Horrocks, Kutz, and Sattler (2006) introduce a role box (RBox) for maintaining the hierarchy of roles and role assertions.

Later sections describe ontologies with a DL formalism. Table 3.2 shows the essentials to follow the notation including concept subsumption, equivalence and qualified cardinality constraints. The illustrated elements are sufficient for following the ontology design in this thesis. For readability the notation is limited to a minimum. For a full introduction into the basic notation the user is referred to (Bader et al. 2007).

Table 3.2: *Notation for ontology description based on (Krötzsch, Simancik, and Ian Horrocks 2012).*

Syntax	Description
$A \sqsubseteq B$	concept A subsumes B , i.e., A is a subconcept of B
$A \equiv B$	concept A equals B
$\leq n.R.C$	relation R to at most n instances of concept C
$\geq n.R.C$	relation R to at least n instances of concept C
$C(a)$	assertion of the instance a to a concept C
$R(a,b)$	assertion of the relation between instances a and b

Knowledge-based systems in robotics research Robotics research approaches use semantic descriptions and in particular ontologies to represent knowledge to establish a common language between robotic agents. The project ROSETTA (Patel, Hedelind, and Lozan-Villegas 2012) for example uses semantic technologies to improve the application of robots for industrial automation. An ontology-based infrastructure named Knowledge Integration Framework (KIF) serves to maintain a knowledge base with distributed sources for a manufacturing system which intends to operate robots in close partnership and interaction with human workers. KIF, developed by Björkelund et al. (2011) and Persson et al. (2010), translates automation descriptions, which are given in the standardised format AutomationML (Schmidt and Lüder 2015), into OWL 2 knowledge fragments known as Resource Description Framework (RDF) triples (Manola and Miller 2004). These triples can be stored in special databases, so-called triplestores. KIF accounts for multiple available and distributed triplestores, and thereby forms a distributed data store - a common query interface based on the standardised language format SPARQL (Prud'hommeaus and Seaborne 2008) acts as a single point of access. KIF reuses essential and typical elements of an OWL 2 based infrastructure to store knowledge fragments and query data, but it leaves out an explicit reasoning layer.

Rockel et al. (2013) use ontologies in a robotic learning context. They suggest an architecture to semantically relate actuator and sensory experience for a single robot. Similarly to Persson et al. they use an RDF triplestore as knowledge base. Reasoning services, such as reasoning over time and space, are implemented based on a constraint processing approach. To prepare the use of quantitative data for the reasoning process, Rockel et al. apply Semantic Web Rule Language (SWRL) (Ian Horrocks, Patel-Schneider, et al. 2004) which allows to augment an existing knowledge database with Horn-like rules. This way, forward inference can be easily applied.

Another related approach is KnowRob: Tenorth and Beetz (2013) define a knowledge-based processing infrastructure mainly for single robots which operate autonomously. KnowRob provides an ontology-based modelling for robots and define dependencies between function and robot hardware. Thereby, the system is capable of identifying capabilities of robots by verifying the existence of the required hardware components in a robot. Similar to other ontology-based approaches KnowRob provides an open extensible knowledge database so that existing ontologies can be reused, e.g., ontologies about common sense knowledge (R. Gupta and Kochenderfer 2004). KnowRob's model-based reasoning is based on OWL 2's property restriction *someValuesFrom* which represents an existential resource constraint. The general reasoning in KnowRob, e.g., upon available hardware and capabilities, relies on Prolog, which Tenorth and Beetz (2013) see as an efficient reasoning approach especially when its use is applied to rather simple SQL-like queries. The use of KnowRob is a popular choice in context of the Robot Operating System (ROS) (Quigley et al. 2009), because it can be used to adapt and augment generic plan definitions according to a robot's current context. Such plans can be specified using the CREAM Plan Language (CPL) (Beetz, Mösenlechner, and Tenorth 2010; Tenorth and Beetz 2013), which permits a late symbol grounding. Thereby, plan templates are augmented or instantiated at runtime with the latest information, including collected experience and current observations. Thereby the overall robot performance can be improved.

In the context of reconfigurable robotic systems Burroughes (2017) relies on semantic technologies. He designs a robot control architecture which can reconfigure a single robot as a result of an observed system failure. Similar to KnowRob, the architecture only refers to a single robot only and Burroughes introduces a reconfiguration layer which stretches over all layers of the classic 3-tier architecture, i.e., the deliberative, executive, and functional layer. The architecture reuses the existing template for an *autonomic manager* and therefore implements a MAPE-K (IBM 2005) based reconfiguration process, where MAPE-K refers to a control cycle consisting of four steps: (M) monitor, (A) analyse, (P) plan, and (E) execute. The (K) refers to a shared knowledge base, for which Burroughes relies on ontologies. Burroughes motivates the use of ontologies with a comparison of First Order Language (FOL), DL, ontologies, and pure model-based approaches. He concludes that ontologies represent the best comprise in term of expressiveness and computational effort. Furthermore, for single robotic systems Hernández, Bermejo-Alonso, and Sanz (2018) suggest a framework for self-adaptation based on so-called functional ontologies. They introduce the concept of a meta-controller which uses knowledge about the functional composition of a system to adapt. The main objective of using a meta-controller is to find and use an alternative functional setup to cope with failures of individual components. A model for function decomposition and mapping from structure to function will similarly be used in MoreOrg to identify functionally equivalent agent compositions.

Summary The use of semantic technologies is a proven approach in practice to support open extensible knowledge-based systems. In addition, any ontology-based modelling approach can benefit from international standardisation and related support tools. While ontologies provide a limited description of a domain they should be designed by domain experts. For instance for the robotics domain the development of an common ontology is fostered by Ontologies for Robotics and Automation (ORA) (Prestes et al. 2013). However, there is no standard approach to construct an ontology, to improve interoperation and knowledge exchange increasing standardisation in terms of ontologies is needed.

3.1.4 Coalition Games

A *characteristic function game* considers agent coalitions and allows to quantify the general benefit of the overall coalition as well as the resulting benefit for member agents. Hence, a coalition or groups of agents maps to single real values (see (Weiss 2009)). Agents can only participate in a single coalition which leads to forming distinct so-called coalition structures. An evaluation of a coalition structure allows the computation of a payoff vector which quantifies the benefit of this coalition structure for each agent.

Superadditive games are a subclass of characteristic function games, and especially relevant for reconfigurable multi-robot systems since they can represent synergy: each formed coalition can have an equal or greater value than the sum of its individual forming agents. Variants of superadditive games try to add further elements to the approach, e.g., by considering resources or skills, and weighted voting games and coalitional skill games are two distinct approaches which embed a representation for resources. Weighted voting games use a single weight to encode the amount of resources which a single agent holds to support a voting. Coalitions are tested upon a quota of the resource in order to identify if they earn a fix payoff; no synergy is considered in the weighted voting games.

Coalitional skill games rely on a skill-based representation of coalitions. In a coalitional skill game a task is defined as sets of skill requirements. Each agents has a set of skills, so that capable agents can be mapped to tasks. In a superadditive game the payoff for agents can be increased through synergies. Hence, agents can be motivated to form coalitions, or to leave a coalition when this change improves their payoff. Conitzer and Sandholm (2006) suggest a special representation for coalition structure with synergies, and introduce the concept *core* to define stable coalition structures. The stability of a coalition structure arises in any coalition structure, where no agent gained an incentive when it would depart from its current coalition.

Coalition structures might be differently suited to support an application or task requirement. Of interest is a coalition structure which optimally solves a problem. Finding this optimal coalition structure is an NP-complete problem (Sandholm et al. 1999). An agent type representation can reduce computational complexity by exploiting symmetries in the agent structure. Elkind, Rahwan, and Nicholas R Jennings (2013) describe the use of an agent-type representation as “significantly more succinct, than the standard one [representation]”. Still, when all agents have different types this leads to a worst case complexity which is equal to ignoring the agent type. Chapter 2.4 lists the basic definitions for a reconfigurable multi-robot system. These definitions are based upon this agent type representation in coalition games as a first means to reduce combinatorial explosion.

Characteristic function games map coalitions to some kind of real or integer value in order to quantify the benefit of a single coalition or the overall coalition structure. This is a necessary prerequisite to search for an optimal coalition structure. Anytime approaches to find an optimal coalition formation have been developed by Rahwan, Ramchurn, et al. (2009) based on Integer Partitioning. These approaches need to search the full search space in worst case to find the optimal coalition, so that $\mathcal{O}(n^n)$, where n is the number of agents involved. In the context of reconfigurable multi-robot systems Baca et al. (2014) have reduced the computational complexity to $\mathcal{O}(\log n)$, but only by introducing stronger assumptions: two linked agents maintain a constant utility independent of the coalition they are in.

Table 3.3: Setup of the multi-robot scenario.

Atomic agent types	Interfaces		Instances
	# male	# female	
Sherpa	4	2	1
Base Camp	5	0	3
CREX	1	0	2
Payload	1	1	10

Table 3.4: Requirements for the given multi-robot scenario.

	count
atomic agents	16
interfaces	43
feasible links	354

3.1.5 Combinatorics

A reconfigurable multi-robot system can form composite agents in different ways depending upon the connectivity of the agents. To consider connectivity of agents, one can search for feasible composition in *agent space*, or in *link space*. Agent space only considers a compatibility level of atomic agents, i.e., defining if two atomic agents are pairwise compatible or not. The existing constraint-coalition formation approaches by Rahwan, Michalak, et al. (2011) use a reduced form of agent space compatibility checking by defining that two agent should not be part of the same coalition. This defines a strong incompatibility of two agents, since the two agents should never, not even indirectly, cooperate. Therefore, the approach is unsuited for an application with reconfigurable multi-robot systems. A relaxed form of connectivity checking prevents the direct cooperation of agent, but allows cooperation through a proxy agent. In a composite agent, for instance, it might be impossible to connect two agents directly since no compatible interface are available. However, a third atomic agent can still act as proxy or rather adapter. Relaxed connectivity checking is therefore required for reconfigurable multi-robot systems. Connectivity checking is detailed in Section 3.3.1.

In agent space the maximum number of agent combinations is based on the powerset \mathcal{P}^n and $|\mathcal{P}^n| = 2^n$ including the empty set, where n represents the number of agents. For example for the agent set $A = \{a, b\}$: $\mathcal{P}^A = \{\emptyset, \{a\}, \{b\}, \{a, b\}\}$. In link space the number combination of agents can be significantly larger, e.g., when two atomic agents can be connected in multiple ways since they comprise multiple interfaces. Given a set of interfaces I a maximum of $l = \binom{|I|}{2}$ links and thus $|\mathcal{P}^I| = 2^l$ combinations have to be considered. The count l is only an upper bound, since the creation of composite agents has to follow additional restrictions, e.g., a physical interface can only be used for a single link, compatibility between interfaces is limited and self-referencing links are not be permitted. To estimate the gap between number of feasible combinations and the upper bound, and to take an initial view at restricted connectivity in link space a generative approach has been applied as part of this thesis.

Table 3.3 illustrates an example setup for a reconfigurable multi-robot team based on the robotic team in RIMRES (see Chapter 2.2.2). The number of pairwise feasible links is listed in Table 3.4, and it shows that from $\binom{43}{2} = 903$ links only 354 are feasible. Figure 3.1 shows the number of composite agent instances up to a composition size of 4 atomic agents. The respective algorithm for the generation is listed in Appendix C. The bound for feasible composite agents remains more than a magnitude smaller than the upper bound, but still multiple magnitudes larger than the number of composite agent types. Compared to agent space the number of composite agent types is a magnitude larger in link space, i.e., as a result of multiple ways to connect two agents. This evaluation explores empirically the computational bounds for constrained coalition formation in link space for a very restricted coalition size.

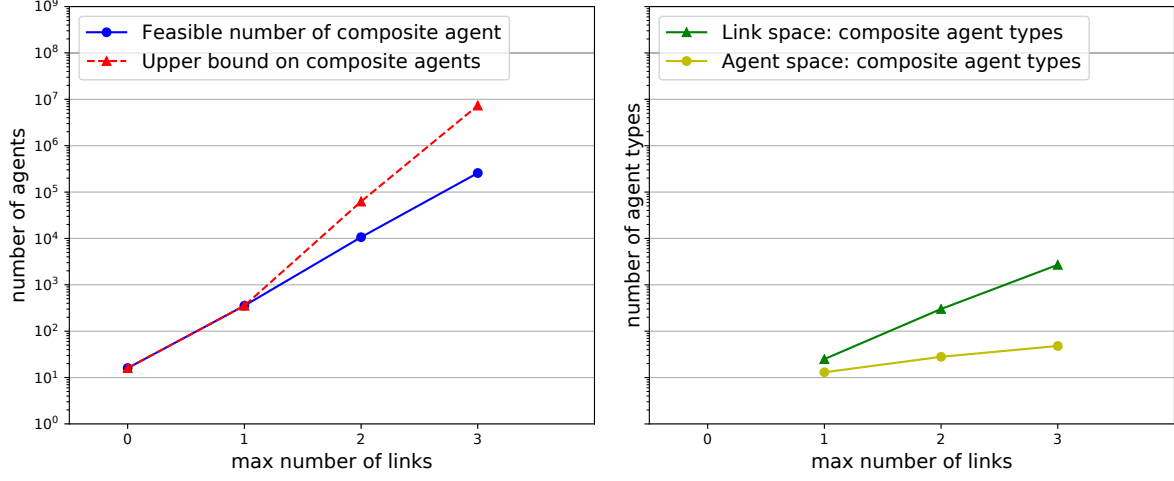


Figure 3.1: Combinatorial explosion for an example scenario with an upper link bound of 3 (left), and the corresponding composite agent types for link space and agent space (right) (both y-axis have the same logarithmic scale).

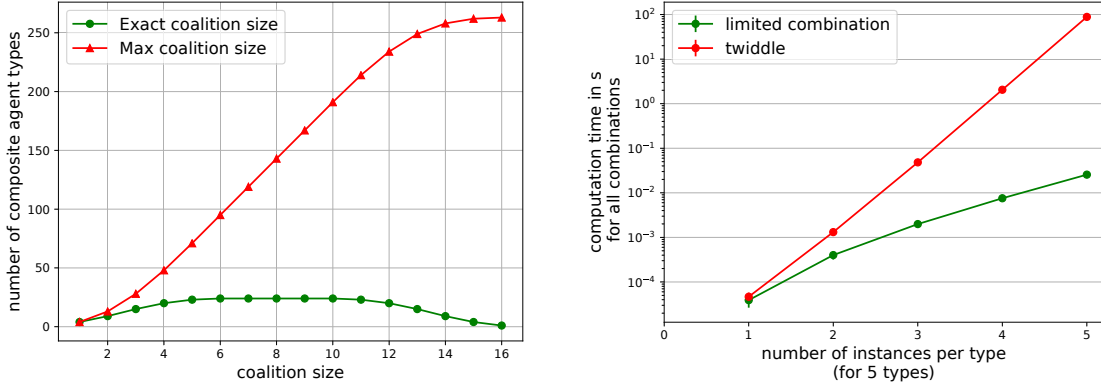
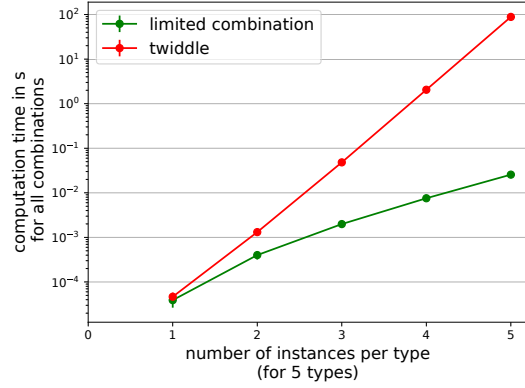


Figure 3.2: Upper bound in agent space: exact and accumulated composite agent types.

Figure 3.3: Comparison of performance between Twiddle and the generative limited combination approach as developed in this thesis. The number of element types is fixed to five, while the number of instances per element type homogeneously varies. Y-axis with logarithmic scale.



The evaluation suggests to work with agent types in agent space to lower the computational complexity. Figure 3.2, however, illustrates the development of the number of composite agent types in a worst case scenario, i.e., when all combinations of agents are feasible. The amount of composite agent types to be considered is still significantly larger than the number of agents. Hence, dealing with a large number of reconfigurable agents requires additional measures to reduce computational complexity apart from accounting for agent types in agent space only.

The general agent definition in Chapter 2.4 is representing agent types in agent space, and Section 3.3.1 further introduces a connectivity test to check for feasible agent coalitions in link space.

Combinatorial Algorithms The generation of permutations and combinations is an often recurring operation when dealing with composite entities. The generation of combinations is frequently occurring when instantiating an agent pool. The generation of permutations is an operation embedded in the standard libraries of C++. For an efficient generation of combinations, however, additional libraries or implementations are required. The algorithm *Twiddle* (Belmonte 1996; Chase 1970) is a popular choice. The runtime performance for generation combinations can be improved when using a generative approach which exploits repetitions of elements. Consequently, the *limited combination* approach has been developed in this thesis. The approach generates a combination from a set of unique elements S , and cardinality s_n for each $s \in S$ which defines the number of repetitions of s (see (Schwendner, Joyeux, et al. 2012)).

The implemented algorithm as listed in Appendix C.2 outperforms the Twiddle-based generation of all combinations when repetitions of elements are required. Figure 3.3 shows the comparison of Twiddle against the generative limited Combination approach. In the test setup all combinations for five element types are generated. For each element type the number of its instances in a generated combination are defined. The generation is tested for cardinalities from 1 to 5, i.e., leading to a maximum combination length of 25. Results have been averaged over ten runs (performed on an Intel i7 4600U CPU @ 2.10 GHz, 12 GB RAM). Figure 3.3 illustrates the computational advantage of the limited combination algorithm for an increasing number of element repetitions.

3.2 Modelling Approach

Reconfigurable multi-robot systems are flexible to form composite agents which can be composed according to task requirements. This flexibility arises from a high degree of modularity and thus from the standardisation of interfaces which permit the extension of the hardware. While the hardware interface enables the physical linkage, the robot control software (and architecture) has to support the extension of the reconfigurable multi-robot system. Section 2.3 outlines an incremental mission design approach as a practical example to show an obvious benefit of an open extensible reconfigurable multi-robot systems. However, the real benefit is foreseen in the flexibility to adapt during robot operations. The configuration options and thus the options for adaptation increase with the number of atomic agents that participate in a robotic organisation.

Handling a large number of atomic agents, however, suffers from a combinatorial challenge. Feasible coalition structures have to be identified and one, ideally the optimal with respect to

the current requirements, has to be selected. The application of a reconfigurable multi-robot system hence easily turns into an optimisation problem, which is defined by the available set of agents and the goal of the organisation. Chapter 4 deals with this optimisation problem as part of a planning approach, which requires a model of the organisation as prerequisite to work with organisation states. This model not only serves as basis for planning, but also allows system designers to analyse a robotic organisation with respect to its properties and functionalities.

The organisation model also embeds practical considerations. To achieve a scalable management approach of a reconfigurable multi-robot system a high degree of standardisation at multiple levels of hardware and software is required. The definition of the physical compatibility and enabling a connection via EMIs is only one enabling requirement. This section therefore outlines the central design decision for the organisation model MoreOrg, where the main objective of the model lies in reflecting the merging of multiple atomic agents to reason with and maintain a scalable, open extensible agent architecture.

The modelling approach splits into three levels as depicted in Figure 3.4: (1) organisation level. (2) (general) agent level, and (3) atomic agent level. Each level corresponds to a distinct decomposition view of a robotic organisation. Inference has to be used to characterise the agents at all layers. Atomic agents are primarily defined by static properties and static resource assignments. However, they also comprise properties which depend upon other properties and functionalities that are inferred from the existence of hardware and software resources. The overall set of properties for both composite as well as atomic agents is, however, unified as result of their shared definition as general agent. For instance, whether an atomic agent is mobile depends upon the existence of a power source in combination with the functionalities locomotion, mapping, and localisation (see also Figure 3.8). Either an atomic agent already encompasses all required functionalities, or a composite agent gathers these functionalities by combination of multiple atomic agents.

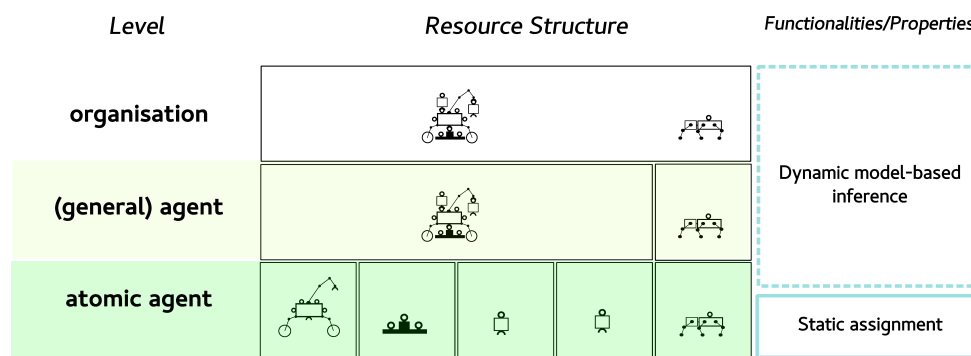


Figure 3.4: Characterisation of the robotic organisation at its different decomposition levels.

Ontology-based Modelling

Semantic technologies have a wide range of applications and have already been adopted by robotic practitioners to design robot architectures. Available standards and tools make an application of semantic technologies attractive, since they allow for a scalable approach and support knowledge exchange through the use of standardised representation formats. Drawbacks

can be observed in the general overhead for rather small or performance critical applications, so that the use of domain-specific model-based approaches still remains a viable alternative.

Extensibility and scalability result from the ability to process new facts coming from new atomic agent types and the combination of new with the existing knowledge. The use of a domain specific implementation based on a standard programming language is one available option to achieve knowledge-based reasoning. As Russell and Norvig (2003, p.241) state: “What programming languages lack is any general mechanism for deriving facts from other facts” and they “lack the expressiveness required to handle partial information”. Even without the need to derive new facts, the use of programming languages for knowledge representation is restricting, e.g., there is neither a standard mechanism to extract information about the class hierarchy, nor a standard way to manipulate this class hierarchy. Hence, even for limited scenarios where a knowledge exchange or update between multiple agents is required, the use of a commonly agreed representation is beneficial.

To guarantee an openly extensible system, MoreOrg is based on an ontological database in combination with using DL as foundation for reasoning. An additional custom reasoning approach tackles particular needs to model reconfigurable multi-robot systems. The usage of the ontological database is an integral part of MoreOrg and exploits available standardisation. Figure 3.5 illustrates the general architecture of the model implementation. The ontological description of atomic agents is a static part of the database and uses OWL 2 (W3C OWL Working Group 2012) as standard technology. Inference is required to compute agent and organisation properties and MoreOrg relies on a generic and a domain specific reasoning part. The generic reasoner is mainly used to identify the modelling of class and property hierarchies, and to identify resource relations. Meanwhile, the domain specific reasoning identifies suitable agent for a particularly requested functionality. The following paragraphs discuss some of the design decisions and features of developed organisation modelling approach.

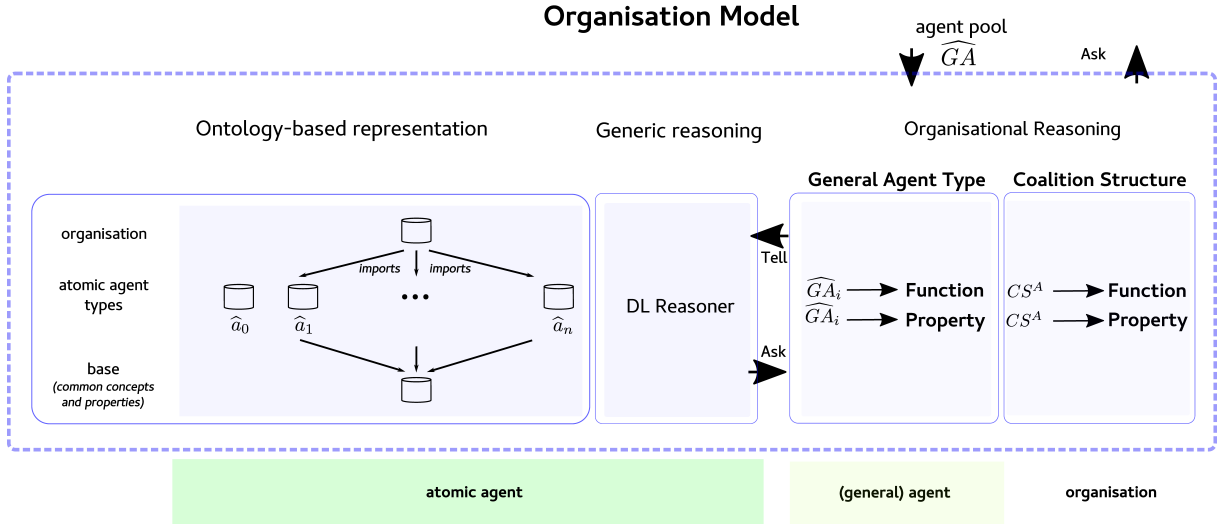


Figure 3.5: General architecture of the organisation model.

Modular Ontology Structure Modularisation of ontologies can be seen as a best practice to manage common knowledge and domain specific knowledge. The common knowledge is defined in a so-called *upper* or *top* ontology which is detailed by adding domain specific ontolo-

gies. The ontological description for the organisation model is likewise modularised, so that the knowledge database can be decomposed according to the modularity of a reconfigurable multi-robot system. Each atomic agent provides its own ontological description, so that an extension of the knowledge base can be easily performed when new agents enter the organisation. Figure 3.5 depicts the base level in the ontological description, which represents all commonly shared concepts and properties to define reconfigurable multi-robot systems. An example for an ontological description is provided in Figure 3.8 and in Appendix B.

For any presented evaluation in this thesis the base ontology contains only the minimum knowledge required to perform the evaluations. Any extension and inclusion of general, e.g., common sense ontologies, has to be performed at this base level.

The organisation level ontology is a single ontology which imports all descriptions of atomic agents. It comprises organisational specific concepts and restrictions, e.g., property constraints can be used to detail the relation between atomic agent types. MoreOrg does not apply social norms, i.e., it lacks a normative level which is found in organisation models like MOISE+ or OMNI (as discussed in Section 3.1). It considers restrictions only for the structural and functional levels.

The organisation modelling approach can support centralised and distributed approaches by maintaining a modular structure. Any chosen agent control architecture still needs to take care that an agent acquires information from other agents and syncs available information. This might be needed to account for a dynamically changing system where agents can enter and leave the organisation. Hence, depending upon the final application context a frequent update of the organisation level ontology might be necessary. This includes a discovery and data collection process for available atomic agents.

Meta-modelling The organisation model uses a meta-modelling approach and focuses on the description of atomic agent types. Meta-modelling is one of the improvements of OWL 2 over its predecessor OWL 1. OWL 2 permits the use of the meta-modelling feature *punning* (Golbreich, E. K. Wallace, and Patel-Schneider 2012), which relaxes a previously strict separation between class names and names of individuals. As a result, a class name might also be used as name for an individual, which can be used to describe class characteristics by making assertion to the correspondingly named individual. In addition to this meta-modelling capability OWL 2 allows the property qualification of cardinality restrictions, i.e., the relation between instances of particular classes can be constrained.

The ontology encodes class inheritance, properties of classes, and qualified cardinality constraints, which is sufficient to define the resource structure of atomic agents. The use of a cardinality-based representation in combination with using the meta-modelling feature *punning* is a flexible and scalable approach to model agent types. The ontology specifies the maximum number of associated resources using qualified maximum cardinality constraints for each atomic agent type. The qualification refers to hardware and software resources. Since (general) agents' capabilities depend only upon the availability of these resources, this description is the basis for the mapping between available resources and functionality of an agent type. Each atomic agent type's ontology can encode the decomposition of an agent type down to a custom level of detail. Figure 3.6 depicts the class hierarchy of the base ontology, with the core classes described in Table 3.5.

The hierarchy of properties used in MoreOrg is shown in Figure 3.7. Data properties are as-

Table 3.5: *Class descriptions.*

Name	Description
Agent	The parent class for all agent types
PhysicalEntity	The parent concept for hardware components
Capability	Represents an ability of an agent (instance) to perform some function. Software components or drivers might be required to establish a certain capability for the actual robot. They are, however, not directly available to other robots.
Service	A function (offer) to benefit other clients (including the agent itself)
Functionality	The parent and wrapping concept for Service and Capability. Checking the availability of a Functionality is essential for the planning approach outlined in Chapter 4. It permits the identification of agent suitable to perform a task.
Interface	An interface might be a hardware or software interface, permitting to establish inter- and intra-agent connections.

sociating numeric types with class instances. The selected set of properties enables a basic characterisation of atomic agent, e.g., mass and typical (nominal) speed for operation, where immobile agents have a speed of 0 m/s.

Although only computed dynamically, a composite agent is likewise characterised by properties. The quantification of properties, however, is performed on the basis of heuristics and policies to allow for a systematic inference and attribution. Section 3.5.1 provides the details of this inference. The object property `has` is used to relate resources to subcomponents, e.g., to define how many physical entities a atomic agent has. It is therefore the most often used object property. The object property `has` is transitive, so that any child class inherits the components of a parent class. The object property `compatibleWith` allows to declare the compatibility of interface classes among each other. The object property `compatibleWith` is symmetric, i.e., for interface classes A and B the following holds: $A \text{ compatibleWith } B \iff B \text{ compatibleWith } A$. The object property `hasTransportCapacity` allows to restrict the number of resources of a particular class which can be transported by using property qualification.

The ontology-based representation allows to characterise atomic agent types using a set of data and object properties, and the organisation model allows to subsequently infer properties for composite agent types (see Section 3.5.1) and the overall organisation. Figure 3.8 shows a related presentation using DL, where the `has` property is used to define requirements on other resources.

Qualified Cardinality Constraints The general ontology design in MoreOrg is similar to the KnowRob approach developed by Tenorth and Beetz (2013). A significant distinction, however, is the use of qualified cardinality constraints in MoreOrg to encode resource dependencies. KnowRob uses the property constraint `owl:someValuesFrom` to encode a dependency, which is equivalent to a minimum cardinality of one for a particular resource. Furthermore, in KnowRob an agent's available resources are explicitly listed, which corresponds to specifying a minimum and maximum cardinality of one for an available resource.

Using (qualitative) cardinality constraints in MoreOrg has the advantage of making the number of resources that an atomic agent type provides directly quantifiable. The alternative representation, which is equivalent to KnowRob's approach, has to relate a set of resource instances with

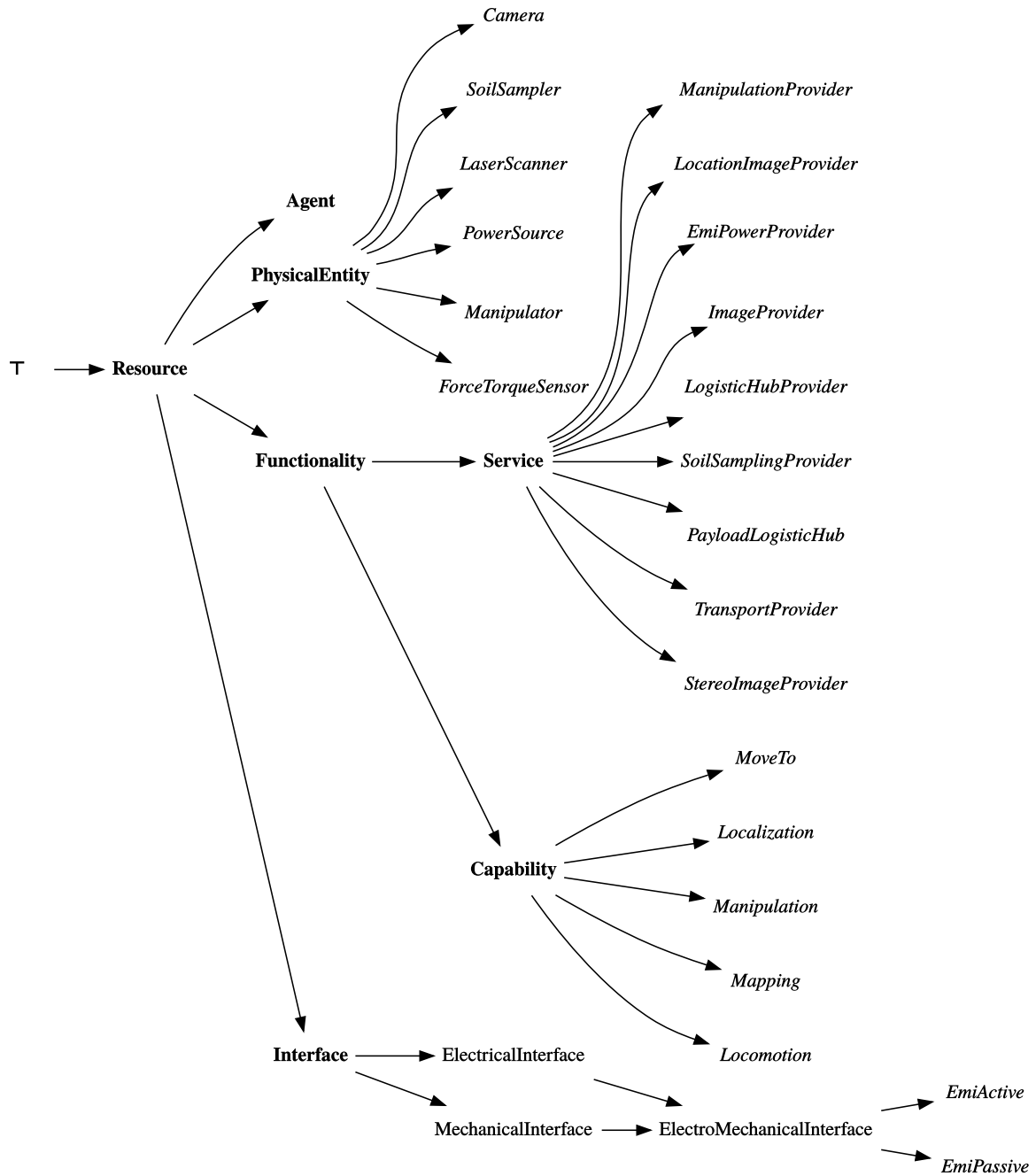


Figure 3.6: Class hierarchy excerpt of the base ontology.

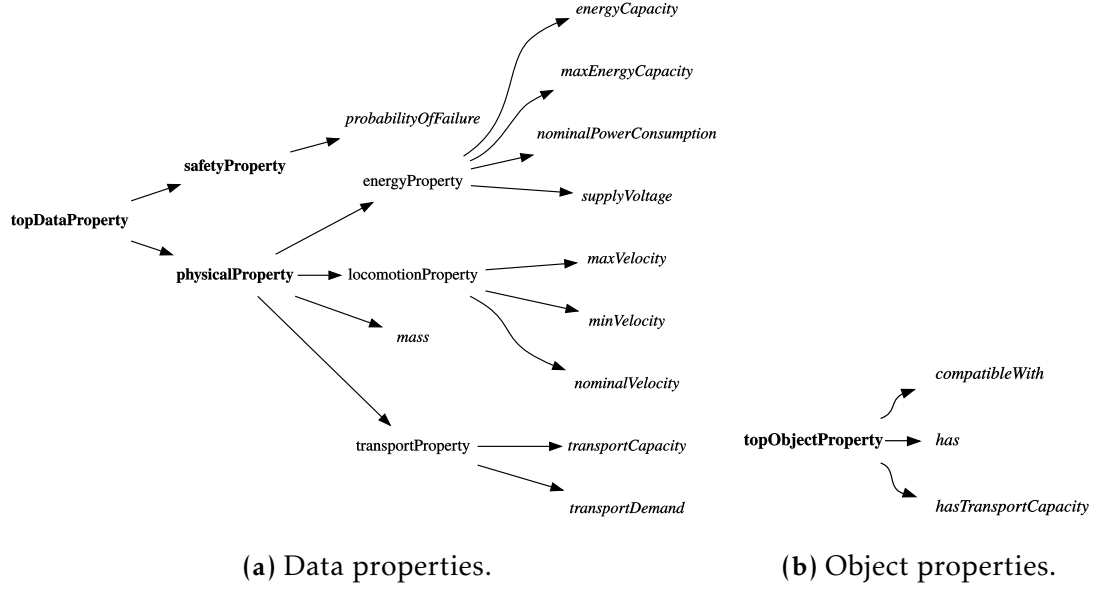


Figure 3.7: Property hierarchies of the base ontology.

<i>Capability</i>	$\sqsubseteq \text{Functionality} \sqsubseteq \text{Resource} \sqsubseteq \top$
<i>Service</i>	$\sqsubseteq \text{Functionality} \sqsubseteq \text{Resource} \sqsubseteq \top$
<i>MoveTo</i>	$\sqsubseteq \text{Capability}$
<i>ImageProvider</i>	$\sqsubseteq \text{Service}$
<i>MoveTo</i>	$\equiv \geq 1.\text{has.Locomotion}$
	$\sqcap \geq 1.\text{has.Localization}$
	$\sqcap \geq 1.\text{has.Mapping}$
	$\sqcap \geq 1.\text{has.PowerSource}$
<i>ImageProvider</i>	$\equiv \geq 1.\text{has.Camera}$
	$\sqcap \geq 1.\text{has.PowerSource}$
<i>LocationImageProvider</i>	$\equiv \geq 1.\text{has.ImageProvider}$
	$\sqcap \geq 1.\text{has.MoveTo}$
<i>ARobot</i>	$\equiv \text{Agent}$
	$\sqcap \leq 1.\text{has.Locomotion}$
	$\sqcap \leq 1.\text{has.Localization}$
	$\sqcap \leq 1.\text{has.Mapping}$
	$\sqcap \leq 4.\text{has.Camera}$
	$\sqcap \leq 2.\text{has.EmiActive}$
	$\sqcap \leq 4.\text{has.EmiPassive}$
	$\sqcap \leq 1.\text{has.PowerSource}$

Figure 3.8: Organisation model excerpt of a DL-based description of an atomic agent concept *ARobot*. The example associates a functionality named *LocationImageProvider* with the agent concept, reflecting the ability to take images at different locations.

a given atomic agent type. Figure 3.8 illustrates the essential use of minimum and maximum cardinality constraints in MoreOrg, i.e., $\geq n.R.C$ and $\leq n.R.C$, with cardinality n , relation/property R , and concept/class C . Minimum cardinality constraints define resource requirements, whereas maximum cardinality constraints encode resource availability. An agent comes with a maximum number of available resources and is viewed as a resource provider. In contrast, functionality is defined by its minimum required number of resources.

While cardinality constraints make subcomponents of resources countable, minimum and maximum cardinality constraints are also a means to deal with model changes. Real robots suffer for example from an outage of a hardware or software component. This loss of a component can be modelled by adding a (now lower) maximum cardinality constraint. This measure, however, prevents a usage of this new robot type as substitute where the parent type is actually required. As a consequence, as soon as an agent loses a component, it effectively changes its type.

Reasoning MoreOrg does account for two levels of reasoning: (a) a generic reasoning which is based on DL, and (b) the domain specific reasoning built on top of the generic reasoning and additionally applying model-based reasoning. The generic reasoning can be achieved by an existing OWL 2 related reasoner. The domain specific reasoning is a contribution of this thesis. It focuses on reconfigurable multi-robot systems and adds organisational reasoning. It enables a model-based approach for reasoning with composite agents which is based on qualified cardinality constraints.

Joining two agents results in a merge of the related resources, so that the resulting composite agent can be described with a set of cardinality restrictions. The respective cardinalities of the atomic agents are summed. MoreOrg introduces an algebra to deal with functionality and property inference of composite agent types.

For each resource concept r the following summation rule holds:

$$card_{max}(r, \widehat{A}) = \sum_{\hat{a} \in \widehat{A}} \gamma_{\widehat{A}}(\hat{a}) \cdot card_{max}(r, \hat{a}) \quad , \quad (3.1)$$

where $card_{max}(r, \hat{a})$ is the maximum cardinality of resource concept r for a given agent type \hat{a} . The equivalent operation applies for the summation of minimum cardinality restrictions $card_{min}(r, \hat{a})$. This model-based approach uses information about an agent type and interprets the maximum resource cardinalities as indication of its nominal resource composition. An equivalent approach which does rely on the explicit association with resource instances requires to count all available instances:

$$card(r, A) = \sum_{a \in A} card(r, a) \quad , \quad (3.2)$$

where $card(r, a)$ is the exact cardinality of a resource concept r for a given agent. The former min, max cardinality approach has the advantage of a compact representation, which can directly use the specified cardinality. The latter can directly link resource instances to system resources, which thereby leads to an exact description of an agent. It requires, however, to count the associated resource to identify (exact) resource cardinalities. The model-based approach in combination with min and max cardinalities has been selected for use in MoreOrg, since it provides a concise and flexible representation which permits the direct reasoning with agent

<i>compatibleWith</i>	$\sqsubseteq \text{ObjectProperty}$
<i>ElectronicalInterface</i>	$\sqsubseteq \text{Interface} \sqsubseteq \top$
<i>MechanicalInterface</i>	$\sqsubseteq \text{Interface} \sqsubseteq \top$
<i>ElectroMechanicalInterface</i>	$\equiv \text{MechanicalInterface} \sqcup \text{ElectronicalInterface}$
<i>EmiPassive</i>	$\sqsubseteq \text{ElectroMechanicalInterface}$
<i>EmiActive</i>	$\sqsubseteq \text{ElectroMechanicalInterface}$
<i>EmiActive(EmiActive)</i>	
<i>EmiPassive(EmiPassive)</i>	
<i>compatibleWith(EmiActive, EmiPassive)</i>	

Figure 3.9: Knowledge base example to account for interface compatibility.

types. Furthermore, it is open for a complementary application of the mentioned approach of using resource instances.

Interface Compatibility As described in Chapter 2.2 the use of an EMI is the distinctive feature of a reconfigurable multi-robot system compared to a standard multi-robot system. The design of these interfaces varies. For instance, in the case of this thesis' reference system three different coupling mechanisms exists. These involve gendered interfaces, so that connections of atomic agents are limited to compatible interfaces.

The initial description and definitions which are provided in Chapter 2.4 do not embed interface compatibility (and thus link space), and only describe combinations of atomic agents in agent space. In order to define link compatibility MoreOrg models interface classes, in combination with the object property *compatibleWith*, which permits the definition of the compatibility between two interface classes.

In general, the organisation model assumes incompatibility. Two interface types are only compatible if explicitly specified. Figure 3.9 lists an example which focuses on the representation of the EMI of the reference system. Note that the feature of previously mentioned *punning* is used to define a class *EmiActive* with a corresponding instance *EmiActive*, as well as a class *EmiPassive* with a corresponding instance *EmiPassive*. This enables the definition of compatibility on the interface instances. Yet, the class based definition of interfaces in MoreOrg is not restricted to the interfaces mentioned here. It can account for an interface concept as soon as it is described in the ontology along with the list of compatible interfaces (see Section 3.3.1).

As shown in Section 3.1 the consideration of interface compatibility reduces the number of feasible composite agents. While the search space in link space is orders of magnitudes larger compared to considering agent space, MoreOrg provides a heuristic search approach to support to the identification of feasible agents. To verify the feasibility of a composite agent MoreOrg searches for a suitable link assignments using a constraint-based programming approach. Section 3.3.1 details the approach.

Implementation Notes FaCT++ (Tsarkov and Horrocks 2006) has been used for the actual implementation to provide the general reasoning capabilities for DL. The use of FaCT++ permits to check for subsumption, enumerate subclasses and subproperties, and it can account for symmetric and transitive properties. The domain specific reasoning uses an in-memory representation of the ontology, and for that purpose a C++-library (*owl_api*) has been implemented based on the work of Horridge and Bechhofer (2011). This library provides additional

functionality to support the reasoning with qualified cardinality constraints, and hence serves as backend in the implementation of MoreOrg.

3.3 Querying Suitable Agents

An organisation has goals. These goals imply tasks and functionality requirements for the agents that form the organisation. Due to the flexibility of reconfigurable multi-robot systems a large variety of agents can be considered. Hence, to achieve its goals an organisation can transition through various coalition structures.

To support the identification of suitable coalition structure, MoreOrg introduces query support to identify relevant and suitable agents with respect to required functionality. Figure 3.10 provides a schematic overview over available filtering stages. It illustrates the essential steps to reduce the list of candidate agents, until a set of suitable agents has been identified. Note that in the actual implementation these steps do not apply in the illustrated order. For Figure 3.10 they are arranged to ease the understanding about the relationship of the stages. The implementation aims at computational efficiency and therefore applies filtering stages with tighter bounds first. The following sections detail each step in this search process.

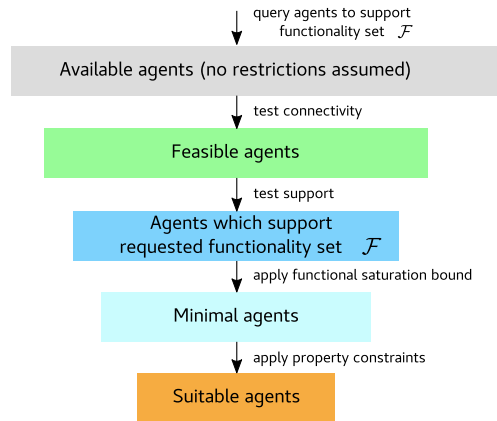


Figure 3.10: Filtering suitable agents out of the set of available agents requires the application of bounds and filtering stages.

3.3.1 Feasible Agents

The main feature of a reconfigurable multi-robot system is the possibility for physical interconnection. The compatibility and availability of physical interfaces restricts connectivity of agents. This restricted connectivity reduces the number of feasible compositions of atomic agents. Hence, a combination of atomic agents in agent space is only feasible, when at least one composite agent can be created from this set of atomic agents in link space.

Definition 3.1 (Feasible agent). Any agent GA for which a feasible link structure exists is denoted by *feasible agent*. Trivially, all atomic agents are feasible agents.

As mentioned in Chapter 2, this thesis' reference systems use an EMI to couple two agents via a single link. A male and female (also referred to as *EmiPassive* and *EmiActive*) variant of the interface exist, whereas only a male and a female interface can be coupled. Atomic agents can comprise any number of interfaces, however, following Assumption 2.2 exactly one interface can be used for the connection to another agent's interface. For a successful connection, both interfaces need to be compatible.

The compatibility of interfaces can limit the connectivity, and therefore *MoreOrg* supports checking the feasibility of forming a composite agent from a collection of agents. To be able to perform this check, the model requires information about the interfaces which are part of an atomic agent. The class hierarchy as illustrated in Figure 3.6 comprises the definition of the *EmiActive* and *EmiPassive* class, which are used to exemplify the compatibility checking problem for the reference system.

Checking the feasibility of a reconfigurable multi-robot system can be understood as a matching problem for a graph $G = (V, E)$, where each vertex $v \in V$ represents a single interface. The matching problem is restricted through a set of constraints X limiting the existence of edges.

I^A denotes the set of all interfaces of a set of agents A , so that $|V| = |I^A|$ with the corresponding partitioning $I^A = I^1 \cup I^2 \cup \dots \cup I^{|A|}$.

The set of interfaces of an agent a_k is represented as $I^k = \{i_{k,1}, i_{k,2}, \dots, i_{k,|I^k|}\}$. To reflect that each interface belongs to an atomic agent, the interface symbol $i_{a,l}$ has two subindexes: the first indicates the atomic agent to which the interface belongs to, and the second defines the local (per atomic agent) index of this interface.

The adjacency matrix of the graph G is an $n \times n$ matrix \mathbf{C} , where $n = |I^A|$, and $\forall i, j \in I^A : c_{i,j} \in \{0, 1\}$. The rows and columns in the following equation are annotated with the corresponding interface symbol to improve readability.

$$\mathbf{C} = \begin{pmatrix} \overset{i_{1,1}}{c_{i_{1,1},i_{1,1}}} & \overset{i_{1,2}}{c_{i_{1,1},i_{1,2}}} & \cdots & \overset{i_{n,n}}{c_{i_{0,0},i_{n,n}}} \\ c_{i_{1,2},i_{1,1}} & c_{i_{1,2},i_{1,2}} & \cdots & c_{i_{0,1},i_{n,n}} \\ \vdots & \vdots & \vdots & \ddots \\ c_{i_{n,n},i_{1,1}} & c_{i_{n,n},i_{1,2}} & \cdots & c_{i_{n,n},i_{n,n}} \end{pmatrix} \begin{matrix} \overset{i_{1,1}}{i_{1,1}} \\ \overset{i_{1,2}}{i_{1,2}} \\ \vdots \\ \overset{i_{n,n}}{i_{n,n}} \end{matrix} \quad (3.3)$$

This matrix \mathbf{C} is symmetric, i.e., $c_{p,q} = c_{q,p}$, where $p, q \in I^A$. Checking connectivity involves searching for a valid assignment for the adjacency matrix \mathbf{C} , subject to the following constraints:

Constraint 3.1 (No self-connection). *An atomic agent cannot create a composite agent, by connecting to itself. Therefore, no self-links are allowed for an atomic agent:*

$$\forall a_k \in A, p, q \in I^k : c_{p,q} = 0 \quad (3.4)$$

Constraint 3.2 (One link per interface). *The design of existing EMIs restricts each interface to be part of maximum one link:*

$$\forall a_k \in A, p \in I^k : \sum_{q \in I^A} c_{p,q} \leq 1 \quad (3.5)$$

Constraint 3.3 (One link between any two agents). *According to Assumption 2.2 two atomic agents can be connected by one and only one link:*

$$\forall a_k, a_l \in A : \sum_{p \in I^k} \sum_{q \in I^l} c_{p,q} \leq 1 \quad (3.6)$$

To define a single composite agent, any resulting graph has to be connected, i.e., at least one path has to exist between any two vertices. Depending upon the features of a reconfigurable multi-robot system, loops in the connection graph might be possible. For example, a robot that attaches its manipulator to pick an already mounted payload is creating such a loop. MoreOrg assumes a tree-based composition of atomic agents as the default, since loops are not mandatory for a feasible composite agent. The tree shape is verified by additional constraints on the connection graph: the connection graph $G = (V, E)$ has to be connected, and $|E| = |V| - 1$ (see (Newman 2010, p. 129)).

The assignment problem is solved using constraint-based programming and implemented using Generic constraint programming framework (Gecode) (Schulte and Tack 2012), where the matrix entries represent the constraint-satisfaction problem (CSP) variables, each with the domain $D_c = \{0, 1\}$ (The domain directly corresponds to the binary element assignments of the adjacency matrix). The constraint-based programming framework allows to generate candidate solutions for feasible agents, and each resulting graph is finally tested if it is connected. If a candidate assignment does not represent a connected graph the search is restarted.

Figure 3.11 shows the result of a successful search for a feasible composition for a set of 20 atomic agents, 10 for each of type Sherpa and CREX. Each vertex represents a single atomic agent - the vertex label shows the atomic agent type. Edges are coloured according to the interface which belongs to the source vertex. For a compact representation only the used interfaces are part of the visualisation: the atomic agent type Sherpa comprises in total four *EmiPassive* and two *EmiActive* interfaces, the atomic agent type CREX has one *EmiActive*. The existence check proofs that the composite agent consisting of 10 Sherpa and 10 CREX is feasible.

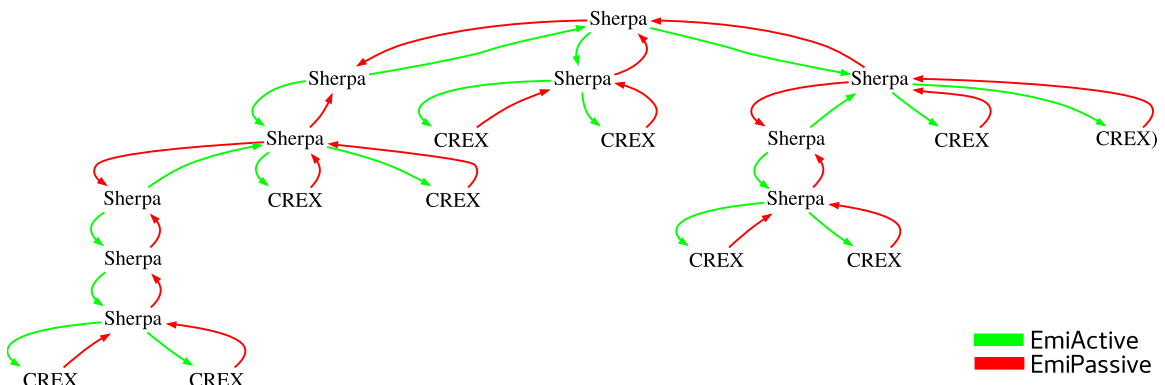


Figure 3.11: One feasible link structure (out of many) for a composite agent after solving the assignment problem. Edges are annotated with the interface corresponding to the source vertex. Agent models and interfaces are related to the reference system described in Section 2.2.

Atomic agents of the same agent type are interchangeable, and interfaces of the same type with the same agent are likewise interchangeable. Therefore, corresponding column assignments

in the adjacency matrix are interchangeable. This information about interchangeable columns can be exploited for symmetry breaking (Mears, De La Banda, and M. Wallace 2014) which is applied in MoreOrg to reduce the number of redundant solutions. A further speed up of the search for a feasible solution has been achieved with the introduction of a heuristic variable assignment. The heuristic assigns variables by preferring interfaces of the currently least constrained atomic agent denoted by a^* :

$$a^* = b + \operatorname{argmin}_{a^k \in A} \frac{1}{|I^k|} \sum_{q \in I^A} \sum_{p \in I^k} c_{p,q}^* \quad , \quad (3.7)$$

where

$$c_{p,q}^* = \begin{cases} 1 & \text{if } c_{p,q} \text{ is already assigned} \\ 0 & \text{otherwise} \end{cases} \quad \text{and}$$

$$b = \text{uniformly distributed random number on } [0, 1 \times 10^{-6}]$$

The small fractional random bias b serves as tie breaker when multiple variables with minimal heuristic value exist. Figure 3.12 illustrates a comparison between a random variable selection strategy, and using the least constrained atomic agent (referred to as *merit min* strategy). The comparison checks the feasibility of a composite agent, which is formed from multiple instance of the same atomic agent type with male and female interfaces. For compositions of atomic agents with hardly any options for connecting to another atomic agents, the *merit min* strategy shows a worse performance compared to a random selection strategy. The computation of the least constrained atomic agent is an overhead, which cannot be compensated for in this case.

Figure 3.12 illustrates the evaluation of the feasibility checking for a set of standard blocks. A standard block can have multiple male and female interfaces. The plots are labelled accordingly: Block <num of female interfaces>-<num of male interfaces>. The feasibility checking for the model of the robot Sherpa is depicted in the bottom right figure. Particularly the evaluation with Block 2-2 and Sherpa show the advantage of the custom selection heuristic. While the number of connection options rises, the random selection strategy is clearly outperformed. The evaluation also indicates how the feasibility checking performs for larger coalitions, here for up to 50 agents.

3.3.2 Suitable Agents

An organisation tries to reach a given goal with the available agents. Since the agents can be created dynamically, the organisation can choose out of all feasible agents. However, depending upon the functional requirements to achieve the goal out of this set of feasible agents, only a subset is capable of contributing towards this goal. This section therefore details the algorithm to identify suitable agents with respect to an organisation's goal and current functional requirements.

Definition 3.2 (Suitable agent). Any feasible agent GA which can support a requested functionality set \mathcal{F} is denoted by *suitable* agent with respect to \mathcal{F} .

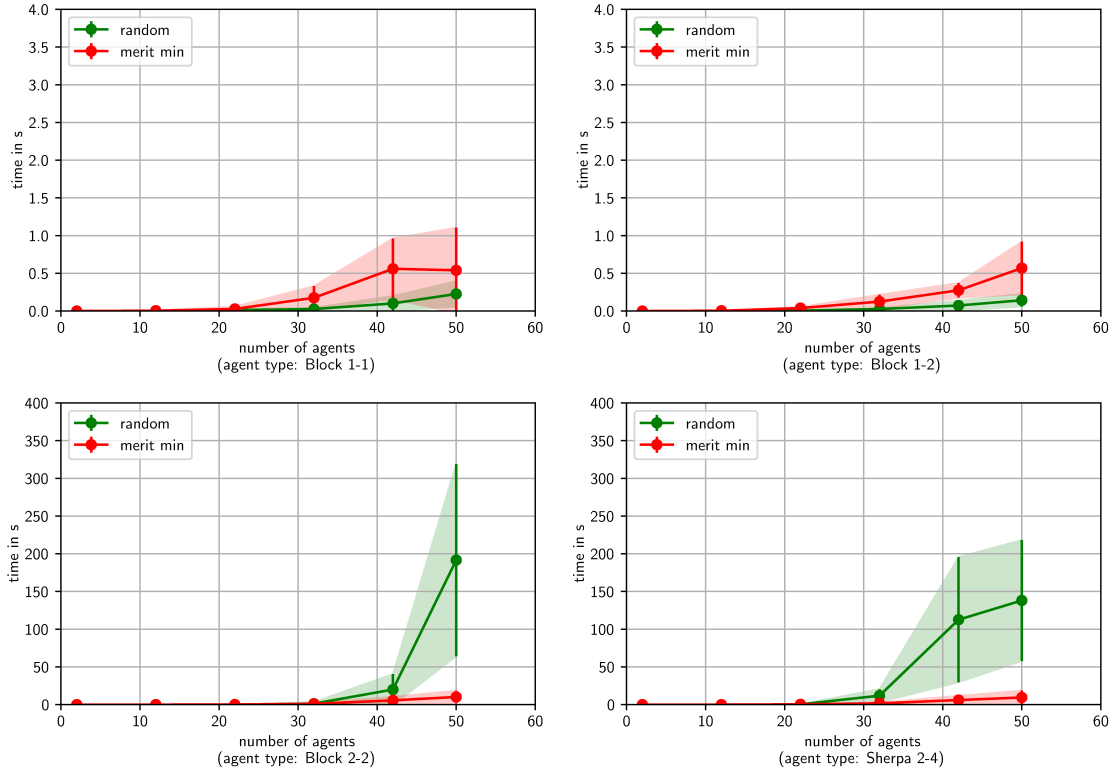


Figure 3.12: The performance of feasibility checking for a composite agent depends upon the selection strategy for variable assignment and the available set of interfaces. Note that different y-axis ranges are used between the upper and lower plots to achieve a more detailed visualisation.

MoreOrg models agents as a collection of resources with a hierarchical dependency structure. The resources that compose an agent are either physical components or virtual components including functionalities. Joining multiple agents into a single composite agent brings together the atomic agents' resources. A composite agent is assumed to represent a single agent which has access to all comprising resources. This resource access forms the basis for superaddition. The agent-based concept of composition and classification into atomic and composite structure is therefore also applied to functionality. The inference of available functionality is based on the same mechanisms for all agents.

Definition 3.3 (Atomic functionality). A functionality f of an agent is defined as *atomic* when no modelled resource dependencies exist and the functionality cannot be further decomposed.

Definition 3.4 (Composite functionality). A functionality f of an agent is defined as *composite* when it can be decomposed into further resources which are part of the agent including, but not limited to other functionalities.

Based on the selected abstraction level, functionality can be modelled as atomic functionality without any further dependency and can trivially be related to an agent type. The definition of composite functionalities can be interpreted as inference rules, which can be applied to check

resource collections for so called functionality support.

Functionality Support

To check whether a composite functionality is supported by a composite agent type the following assumption holds:

Assumption 3.1 (Availability of composite functionality). *The availability of a composite functionality f in an agent a only depends on the availability of the required resources of f .*

Hence, any composite functionality becomes available, when a particular set of resources is brought together by joining two or more atomic agents. The availability or rather *support* quantifies the availability of functionality. Functionality support is defined for an atomic agent type and a single resource concept c as follows (see also (Roehr and F. Kirchner 2016)):

$$support(\hat{a}, c, f) = \begin{cases} 0 & card_{min}(c, f) = 0 \\ \frac{card_{max}(c, \hat{a})}{card_{min}(c, f)} & \text{otherwise} \end{cases}, \quad (3.8)$$

where $card_{min}$ and $card_{max}$ return the minimum and maximum required cardinality of resource instances (including instances of derived resource concepts), respectively.

Accordingly, support of a functionality f with respect to a resource class c can be categorised as follows:

$$support(\hat{a}, c, f) = \begin{cases} 0 & \text{no support} \\ \geq 1 & \text{full support} \\ > 0 \text{ and } < 1 & \text{partial support} \end{cases} \quad (3.9)$$

The categorisation permits to identify useful atomic agents for constructing a composite system that offers a particular functionality: An atomic agent that provide no support do not need to be considered. A single atomic agent that provides full support is sufficient to fulfil the resource requirements. In the case of partial support further analysis is required. The actual support value permits an estimation of the maximum required number of atomic agents to satisfy the resource requirements for the given functionality. While support here is computed for a single resource concept, the categorisation of no, partial and full support has to be likewise applied to a general agent and with respect to multiple functionalities. This is done in the following.

The relationship between the existence of the functionality f and the agent type \hat{a} (here the notation of the agent type \hat{a} represents the corresponding class in the ontology) is described as follows:

$$\hat{a} \sqsubseteq 1.has.f \iff \forall c : card_{min}(c, f) = 0 \vee support(\hat{a}, c, f) \geq 1 \quad (3.10)$$

The inference of functionality support for a general agent \widehat{GA} relies on a similar definition of support:

$$support(\widehat{GA}, c, f) = \frac{card_{max}(c, \widehat{GA})}{card_{min}(c, f)}, \quad (3.11)$$

where the maximum resource cardinalities are computed according to Equation 3.1. The support value can again be categorised in no, partial or full support - equivalently to Equation 3.9.

Support needs to be computed in most cases for a set of functionalities, so that the support for a single functionality can be defined as:

$$support(\widehat{GA}, f) = \min_{c \in \mathcal{C}} \frac{card_{max}(c, \widehat{GA})}{card_{min}(c, f)} , \quad (3.12)$$

where \mathcal{C} is a set of resource concepts and $\forall c \in \mathcal{C} : card_{min}(c, f) \geq 1$ to account only for relevant resource concepts. If there exists a required concept c , such that $support(\hat{a}, c, f) = 0$ then $FSB(\hat{a}, f) = \infty$.

The computation of support by a general agent type for a set of functionalities \mathcal{F} follows:

$$support(\widehat{GA}, \mathcal{F}) = \min_{f \in \mathcal{F}} support(\widehat{GA}, f) \quad (3.13)$$

Example An atomic agent a has at maximum one camera, and one power source. A functionality `StereoCameraProvider` requires a minimum of two available resources `Camera` and at least one `PowerSource`. The atomic agent provides full support with respect to `PowerSource`:

$$support(\hat{a}, \text{PowerSource}, \text{StereoCameraProvider}) = \frac{card_{max}(\text{PowerSource}, \hat{a})}{card_{min}(\text{PowerSource}, \text{StereoCameraProvider})} = \frac{1}{1}$$

The overall support for a stereo camera provider by the atomic agent a is only partial, since only a composite agent type $\widehat{CA} = (\hat{a}, 2)$ has a sufficient number of cameras to fulfil the requirement of a `StereoCameraProvider`:

$$support(\hat{a}, \text{Camera}, \text{StereoCameraProvider}) = \frac{card_{max}(\text{Camera}, \hat{a})}{card_{min}(\text{Camera}, \text{StereoCameraProvider})} = \frac{1}{2}$$

Mapping between Structure and Function

To use the organisation model for planning a mapping between agents and their respective functionalities is required. The mapping function of an organisation always depends upon the available set of atomic agents, and the known functionalities. The available set of atomic agents is represented as agent pool which defines the cardinality for each atomic agent type in the organisation. The corresponding mapping function μ relates functionalities and constructible (general) agent types $\Theta(A)$:

$$\mu : \mathcal{P}^{\mathcal{F}} \rightarrow \mathcal{P}^{\Theta(A)} , \quad (3.14)$$

where $\mathcal{P}^{\mathcal{F}}$ denotes the powerset of all functionalities and $\mathcal{P}^{\Theta(A)}$ denotes the powerset of all constructible agent types.

The general reasoning mechanism to identify support for functionality is presented in Section 3.2. It is the basis to compute the mapping between functionalities and agents. The computation of the mapping function requires the set of functionalities, the known resource concepts and the available set of agents. The organisation model also allows to infer the functionality set from a given set of agent types (based on its associated resource structure):

$$\mu^{-1} : \mathcal{P}^{\Theta(A)} \rightarrow \mathcal{P}^{\mathcal{F}} . \quad (3.15)$$

Since the mapping function uses only information about agent types, the organisation model can precompute the mapping between structure and function. However, the approach suffers from the combinatorial challenge, since an exhaustive computation has to account for all agent types $\Theta(A)$. The powerset of \mathcal{P}^A describes all combinations of atomic agents in A including the empty set. Since $|\mathcal{P}^A| = 2^{|A|}$ a worst case complexity of $\mathcal{O}(2^{|A|})$ exists. Therefore, an exhaustive computation is only reasonable for small agent organisations.

According to the summation rules the cardinality of a resource type in a composite agent is monotonic increasing with each addition of an atomic agent. Yet, an agent has *full support* for a functionality as soon as the minimum resource requirements for this functionality are fulfilled. Composite agents might therefore comprise a high level of resource redundancy with respect to a functionality. So while the total number of agent types supporting some functionality can be large, a subset of $\Theta(A)$ exists which has minimal redundancy.

Definition 3.5 (Minimal agent type). A general agent type which supports a set of functionalities \mathcal{F} and whose agent type cardinalities cannot be further reduced without losing support for \mathcal{F} is denoted by **minimal** with respect to \mathcal{F} . The set of all minimal agent types for an available pool of agents A and a functionality set \mathcal{F} is denoted by $\theta^*(\mathcal{F}, A)$ or equivalently $\theta^*(\mathcal{F}, \widehat{A})$ for the general agent type which correspondingly represents the agent pool A .

A minimal agent type $\widehat{A}' \in \theta^*(\mathcal{F}, A)$ represents a lower bound to satisfy functionality requirements \mathcal{F} with a given combination of atomic agent types. Thereby, a minimal agent type represents a so-called *functional saturation bound*.

Definition 3.6 (Functional Saturation Bound). The minimal number of atomic agents to support a particular functionality (set) is denoted by **functional saturation bound**.

The functional saturation bound for an atomic agent type \hat{a} with respect to functionality f can be computed using the inverse of support (see Definition 3.8):

$$FSB(\hat{a}, f) = \max_{c \in \mathcal{C}} \frac{1}{\text{support}(\hat{a}, c, f)} \quad , \quad (3.16)$$

where \mathcal{C} is a set of resource concepts and $\forall c \in \mathcal{C} : \text{card}_{\min}(c, f) \geq 1$ to account only for relevant resource concepts. If there exists a required concept c , such that $\text{support}(\hat{a}, c, f) = 0$ then $FSB(\hat{a}, f) = \infty$. Similarly, the bound for a set of functions \mathcal{F} is defined as:

$$FSB(\hat{a}, \mathcal{F}) = \max_{f \in \mathcal{F}} FSB(\hat{a}, f) \quad (3.17)$$

The effect of the functional saturation bound can be illustrated with a simple example setup: an atomic agent type \hat{a} has one resource Camera and one PowerSource. Merging two agents of type \hat{a} provides two resources Camera and one PowerSource. The functionality StereoCameraProvider depends upon the availability of two resources Camera and one PowerSource. When the only needed functionality is StereoCameraProvider, any composition of more than two agents of type \hat{a} adds redundancy.

The functional saturation bound for a general agent type \widehat{A} is also a general agent type, here denoted by $\widehat{A}_{\mathcal{F}}$:

$$FSB(\widehat{A}, \mathcal{F}) = \widehat{A}_{\mathcal{F}}, \text{ where } \forall \hat{a} \in \widehat{A} : \gamma_{\widehat{A}_{\mathcal{F}}}(\hat{a}) = FSB(\hat{a}, \mathcal{F}) \quad (3.18)$$

Table 3.6: *A heterogeneous reconfigurable multi-robot team.*

Atomic agent types	Interfaces		Instances in scenario
	# male	# female	
SherpaTT	4	2	5
Base Camp	5	0	3
CoyoteIII	2	0	3
PayloadBattery	1	1	4
PayloadCamera	1	1	4
PayloadSoilSampler	1	1	4

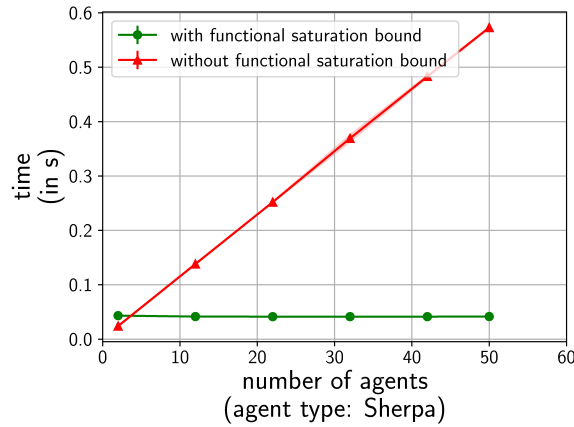
When the number of instances of a general agent type \widehat{A} exceeds the cardinality $\gamma_{\widehat{A}_{\mathcal{F}}}(\hat{a})$, the general agent type is guaranteed to contain redundancies with respect to the functionality set \mathcal{F} . The bound is not exact in the sense, that redundancies can be completely avoided. Even if all atomic agent type cardinalities are below a functional saturation bound, redundancies might exist.

A set of 14 different functionalities as illustrated in Figure 3.6 is part of the default organisation model used in this thesis. Figure 3.13a shows the computation of the functionality mapping with and without application of the functional saturation bound for a homogeneous collection of atomic agents of agent type Sherpa. The composition of two or more Sherpa does not provide superadditive effects, so that the functional saturation easily improves the computation of the functionality mapping. Comparing the computing time for a single instance shows, however, that the computation and application of the functional saturation bound come with additional cost. Figure 3.13b shows the performance for computing the functionality mapping for a heterogeneous team as listed in Table 3.6. It can be seen directly, that the computation time corresponds to the number of agent types to be considered, which remains significantly lower when the functional saturation bound is applied.

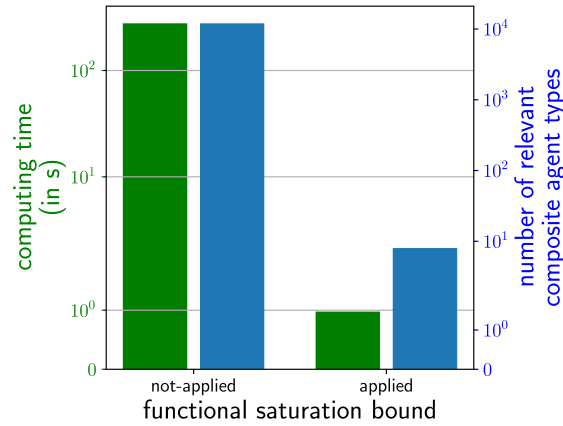
The suggested computation of the functional saturation bound is an effective means to reduce redundant computations. However, some atomic agents that are part of a composite agent do not necessarily contribute directly to functionality. These atomic agents might still be required as structural elements, when otherwise the overall composite agent would not be feasible. Therefore, an exploration of a neighbourhood $\mathcal{N}_{\widehat{A}_{\mathcal{F}}}^{\epsilon}$ of the initial functional saturation bound is necessary, when the bound $\widehat{A}_{\mathcal{F}}$ represents an infeasible agent. The neighbourhood size is bound by the number of additionally considered atomic agents ϵ to enable (structural) feasibility. The selection of the neighbourhood size has a direct influence on the computational efficiency since exponentially more composite agent types have to be considered. Figure 3.13c illustrates the influence of the neighbourhood size ϵ on the computation time for the given example team listed in Table 3.6.

Property Constraints

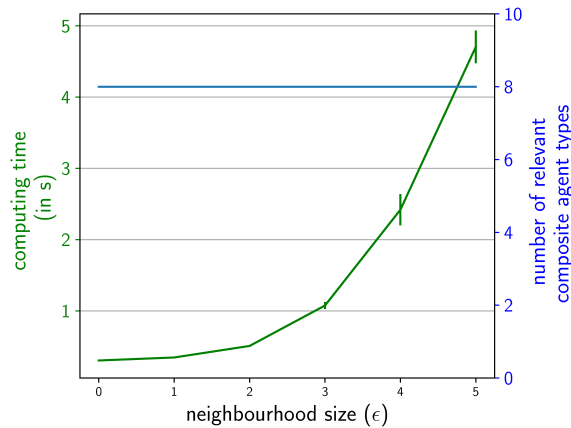
The mapping function μ alone enables queries to identify agent types which support a functionality. However, some applications might need to constrain the query result. Narrowing the resulting set of agent types further can be based on agent type properties. These properties can



(a) Computation of the functionality mapping for a homogeneous collection of atomic agents of type Sherpa.



(b) Time to compute the functionality mapping and number of agent types that have to be considered in the functionality mapping (with and without functional saturation bound) for the heterogeneous team listed in Table 3.6 with neighbourhood size $\epsilon = 0$.



(c) Computation time and relevant composite agent types with respect to the structural neighbourhood size.

Figure 3.13: Example application of the functional saturation bound.

detail the characterisation of the functionality. Ridesharing serves here as an example, since it is also used in Chapter 4.

An application of ridesharing requires at least one system which is capable to transport other agents. An atomic agent which is mobile and which has interfaces to attach other atomic agents can be used as transport system. The related service is named `TransportProvider` (see also Figure 3.6). The transport service might be limited, e.g., to a maximum number of other agents which can be carried.

To express this type of query an additional property-based query level is also part of `MoreOrg`. The goal of this query support is the identification of agent types suited for a set of tasks, whereas each task is attributed with required functionality and agent properties which need to be fulfilled for this task. The combination of functionality and agent property in a query can be understood as a partial template for an agent type. This is similar to agents that fulfil roles in other works (DeLoach 2009; Hübner, Sichman, and Boissier 2004).

A property constraint is defined as $n' \text{ op } n$ for $n.R$, where $\text{op} \in OP$ and OP is a relation operator $OP = \{<, >, <=, >=\}$, and R is the relation/property for an agent $\hat{A} \sqsubseteq n.R$. All functionality requirements can be augmented with property constraints, such that all suitable agents have not only to support the functionality, but in addition also have to fulfil the property constraints. Property restrictions are applied on the set of agents which have been identified to support a requested functionality. Similarly to the summation of resources to identify support for functionality, application of property constraints assumes monotonic increasing property values, e.g., such as mass of a composite system. The property restrictions are verified against the inferred agent properties, e.g., using a requirement for $< 100.\text{mass}$ (SI units apply) allows to restrict the result to all agents below 100 kg of mass.

3.4 Suitable Coalition Structure

The identification of feasible and eventually suitable agents is the basis for the identification of feasible coalition structures, which can support a set of agents with particular functionalities.

Definition 3.7 (Suitable coalition structure). A coalition structure CS^A of an agent set A , where all formed agents provide full support for a functionality set \mathcal{F} , such that:

$$\forall A_i \in CS^A : \text{support}(\hat{A}_i, \mathcal{F}) \geq 1 \quad (3.19)$$

is referred to as suitable coalition structure with respect to \mathcal{F} .

To find a suitable coalition structure `MoreOrg` uses an integer partitioning based algorithm developed by Rahwan, Ramchurn, et al. (2009). While the original algorithm permits the identification of an optimal coalition structure, the algorithm has been adapted to support finding a suitable or rather satisficing coalition structure.

The algorithm represents a characteristic function game approach which relies on value functions for coalitions and coalition structures. A coalition structure in Rahwan, Ramchurn, et al.'s terminology maps to a general agent in `MoreOrg`. The following adaptations lead to a

satisficing search. The value $V(A)$ for a general agent A is computed based on support:

$$V(A) = \begin{cases} 1.0 & , \text{ if } \text{support}(\widehat{A}, \mathcal{F}) \geq 1 \\ 0.0 & \text{ otherwise} \end{cases} \quad (3.20)$$

$V(A)$ thereby encodes the availability or absence of full functionality support. The value $V(CS)$ for a coalition structure CS is computed equally based on the definition of support, and a maximum value requires all agents to fully support the functionality:

$$V(CS) = \begin{cases} 1.0 & , \text{ if } \forall A_i \in CS : \text{support}(\widehat{A}_i, \mathcal{F}) \geq 1 \\ 0.0 & \text{ otherwise} \end{cases} \quad (3.21)$$

Rahwan, Ramchurn, et al.'s algorithm uses integer partitioning to organise the search for possible candidate partitions. For example, for a set of 4 agents, the integer partitions: $\{4\}$, $\{3, 1\}$, $\{2, 2\}$, $\{2, 1, 1\}$, $\{1, 1, 1, 1\}$ have to be checked. Each integer is interpreted as coalition size. Hence, each integer partition can be mapped to a set of coalition structures, e.g., $\{3, 1\}$ defines a team of two agents, when one agent consists of three atomic agents, and the other agent is atomic. The algorithm first identifies the best or most promising integer partition to explore by computing minimum and maximum value bounds for each partition. After identification of the current best integer partition, this partition is exhaustively explored. This exploration computes the characteristic values for all coalition structures which are represented by this integer partition. In worst case all partitions are explored exhaustively.

The search for a suitable coalition structure is used in the planning approach described in Chapter 4 to validate feasible agent transitions between two locations. The algorithm permits the identification of a coalition structure, where all coalitions or rather agents are mobile - if such coalition structure exists.

3.5 Organisation State

The organisation model not only serves to infer a composite agent's functionalities, but in general allows to provide a state description for a reconfigurable multi-robot system. The functionality map is part of this state description as well as the property based characterisation at the organisation, general agent and atomic agent level. Figure 3.14 illustrates the decomposition with correspondence to the required property inference. The atomic agent level defines numeric attributes of the atomic agents. These attributes are known at design time of the atomic agents. In contrast, the attributes at the generic agent level have to be computed dynamically. An exhaustive computation of all feasible composite agent types is inefficient if not infeasible for many applications. The characterisation of a composite agent requires a set of inference and composition rules, which take the properties of atomic agent members into account. The structural status of a reconfigurable multi-robot system can be described as organisation state.

Definition 3.8 (Organisation state). The **organisation state** $OS(A, t)$ denotes the coalition structure $CS^A \in \mathcal{P}^A$ at timepoint t .

MoreOrg provides a description at organisation level and hence characterisation of a reconfigurable multi-robot system based on the organisation state. The following subsections outline

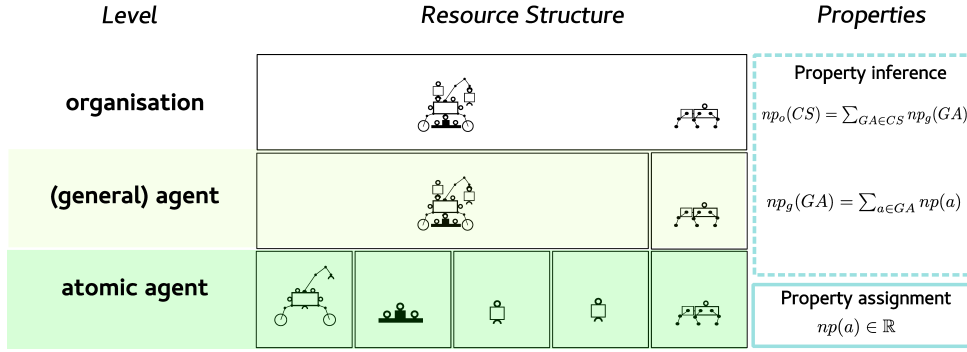


Figure 3.14: The formulas depicted on the right hand side serve as examples for property inference which is based a property value summation.

how MoreOrg quantifies efficacy, efficiency and safety for agent types and subsequently for the overall organisation.

3.5.1 Agent Type Properties

Atomic and composite agents come with a set of attributes and predicates. MoreOrg focuses on a basic set of numeric properties which allow to describe mobile agents that are capable to transport other agents.

Atomic Agent Type Properties

Atomic agent type properties as listed in Table 3.7 are mainly statically defined. However, inference is used to identify, for example, mobility. These properties have been chosen with consideration of the reconfigurable multi-robot planning problem which is described in Chapter 4.

Composite Agent Type Properties

Properties of atomic agent types are either directly set, or can be inferred from the available set of resources in an atomic agent. In contrast, properties of composite agents can only be inferred. Therefore, policies define rules for selection and attribution to guide the inference of properties of a composite agent type.

The default policy corresponds to the summation rule for resources, and a value for a numeric property np for an agent type \widehat{GA} can be derived from its atomic agents:

$$np(\widehat{GA}) = \sum_{\hat{a} \in \widehat{GA}} \gamma_{\widehat{GA}}(\hat{a}) \cdot np(\hat{a}) \quad (3.22)$$

In addition, MoreOrg implements two custom policies: a transport provider policy and a energy consumption policy,

Table 3.7: Atomic agent type properties.

Property	Syntax	Description
mobility	$mobile(\hat{a})$	predicate that defines whether an agent of type \hat{a} is mobile ($mobile(\hat{a})$) or not ($\neg mobile(\hat{a})$). This predicate is inferred by MoreOrg based on the availability of functionalities <i>Locomotion</i> , <i>Mapping</i> , <i>Localisation</i> and a <i>PowerSource</i> .
velocity	$v_{nom}(\hat{a}_i)$	nominal velocity of an agent type \hat{a}_i , $ v_{nom} \geq 0$ for mobile atomic agent types and $v_{nom} = 0$ for immobile
transport capacity	$tcap(\hat{a})$	maximum total capacity of an agent of type \hat{a} to transport others; $acap(\hat{a}_i, \hat{a}_j)$ defines the maximum capacity of an agent type \hat{a}_i to transport an agent type \hat{a}_j , it defaults to the maximum total capacity
transport consumption	$tcon(\hat{a})$	number of storage units an agent of type \hat{a} consumes, when being transported by another agent ($tcon$ is set to 1 for all agent types if not mentioned otherwise)
transport load	$tload(a)$	current load transported by an atomic agent a , i.e., represents the consumed transport capacity of an agent
energy capacity	$ecap(\hat{a})$	maximum total (electrical) energy capacity of an atomic agent type \hat{a}
power consumption	$pw(\hat{a})$	(electrical) power consumption of an agent of type \hat{a}

Transport Related Properties Multiple mobile atomic agents can be part of a composite agent, and mobility might result from superaddition. The evaluated scenario of the reference applications, however, considers only one mobile atomic agent as main transport platform for a composite agent (see Section 2.2). Thus, an atomic agent must be selected which acts as main transport provider. Atomic agents are limited in their transport capacity and therefore a transport policy allows to choose one atomic agent as part of a general agent GA . The selection is based on the subset of mobile atomic agents $MA \subseteq GA \wedge \forall a \in MA : mobile(\hat{a}) = 1$. The atomic agent with the largest transport capacity a^* is chosen:

$$a^* = \underset{a \in MA}{\operatorname{argmax}} tcap(\hat{a}) \quad (3.23)$$

The selection of an atomic agent for transport is the basis for the computation of the properties *velocity* and *transport capacity* of an agent. The velocity of a composite agent type is equal to the selected atomic agent type's velocity:

$$v(\widehat{GA}) = v(\widehat{a}^*) \quad (3.24)$$

The new or remaining transport capacity of an agent type is based on the actual demand of all contained atomic agent types in comparison to the available transport capacity:

$$tcap(\widehat{GA}) = tcap(\widehat{a}^*) + tload(\widehat{a}^*) - \sum_{\hat{a} \in \widehat{GA}} \gamma_{\widehat{GA}}(\hat{a}) \cdot tload(\hat{a}) \quad (3.25)$$

The current approach demands a single transport provider in a composite agent. This can be changed by adding a transport policy that can compute the properties, e.g., nominal velocity, for more complex transport platforms.

Energy Related Properties The energy capacity for an agent type \widehat{GA} is computed based on the default summation rule:

$$ecap(\widehat{GA}) = \sum_{\hat{a} \in \widehat{GA}} \gamma_{\widehat{GA}}(\hat{a}) \cdot ecap(\hat{a}) \quad (3.26)$$

The general power consumption of agents can be computed with the default summation rule:

$$pw(\widehat{GA}) = \sum_{\hat{a} \in \widehat{GA}} \gamma_{\widehat{GA}}(\hat{a}) \cdot pw(\hat{a}) \quad (3.27)$$

Multiple power sources might exist in a composite agent, since atomic agents can share their energy through a power management system as described in Section 2.2. Thus, all powered atomic agents can share operation cost, while an energy consumption policy defines the contribution of each atomic agent. An atomic agent a which provides a power source, i.e., $ecap(\hat{a}) > 0$, takes a share $pw_{quota}(a)$ of the overall power consumption $pw(GA)$ of the agent GA . The share is based on the atomic agent's relative contribution to the overall energy capacity, so that:

$$pw_{quota}(a) = pw(\widehat{GA}) \frac{ecap(\hat{a})}{ecap(\widehat{GA})} \quad (3.28)$$

3.5.2 Organisation Properties

The active set of agents, i.e., the coalition structure of the organisation, has to be analysed dynamically in order to evaluate the selected properties efficacy, efficiency and safety. Chapter 2.1 has introduced these properties, which serve as optimisation objectives for the planning approach outlined in Chapter 4.

Efficacy Efficacy in the context of MoreOrg describes the ability of a reconfigurable multi-robot system to provide a particular functionality. To measure an organisation's efficacy an objective has to be given as a set of required functionalities. The identification of efficacy leads only to a binary result: either the organisation supports the given functionality or not. The mapping between structure and functionality as outlined in Section 3.3.2 is the basis to quantify the efficacy of an organisation, since it allows to verify whether an active set of agents supports a set of functionality. The efficacy of an agent with respect to a required set of functionalities is defined based upon a general agent's *support* for a set of functionalities \mathcal{F} :

$$efficacy(GA, \mathcal{F}) = \begin{cases} 1 & support(\widehat{GA}, \mathcal{F}) \geq 1 \\ 0 & \text{otherwise} \end{cases} \quad (3.29)$$

Likewise, the efficacy of a coalition structure can be accordingly defined as:

$$efficacy(CS, \mathcal{F}) = \min_{GA \in CS} support(\widehat{GA}, \mathcal{F}). \quad (3.30)$$

Efficiency Efficiency describes the cost of performing a task, here measured through the resources 'energy'. To analyse efficiency MoreOrg estimates the operation cost for all agents, which is accounted for as consumed energy. The consumed energy depends upon the time of

operation and the power consumption of each agent. The time of operation depends upon estimated travel cost, time for the requested operation and time for reconfiguration. This approach extends an approach used by Wurm et al. (2013) to estimate the cost based on the travel time between two locations. MoreOrg expects the nominal speed v_{nom} as default property for atomic agents, so that based on this information the duration estimate can be computed. As long as no better estimation is available, the line-of-sight distance between two locations is the basis for the cost computation. Typically, robotic systems consume electrical energy and MoreOrg additionally expects a definition of all agents nominal power consumption. Robotic agents can have a different power consumption. Operation time is therefore not an accurate cost measure. Hence, MoreOrg uses the total energy consumption of a reconfigurable multi-robot system as cost measure. Power consumption can still vary over time with the type of activity. MoreOrg assumes a constant power consumption of all operative atomic agents and leaves a more sophisticated estimation as future enhancement. The total required energy of an organisation represented by the atomic agent set A to perform a mission \mathcal{M} (see Chapter 4 for the complete definition) represents the efficiency of an organisation. It is defined as:

$$E(A, \mathcal{M}) = \sum_{a \in A} op(a, \mathcal{M}) \cdot pw(\hat{a}) \quad , \quad (3.31)$$

where $op(a, \mathcal{M})$ defines the operation time of an agent $a \in A$ in the mission \mathcal{M} .

Reconfiguration of an organisation comes at a cost, and the operation time is influenced by transitions between coalition structures. The time to transition from one coalition structure CS_i^A to another CS_j^A is therefore estimated with a heuristic function. The heuristic firstly assumes basic cost for the number of atomic agents which are involved in the reconfiguration. Secondly, additional and significantly higher cost arise from the need to coordinate multiple agents to exchange atomic agents or to merge. The reconfiguration cost function to form a single agent from an existing coalition structure is:

$$\rho(GA, CS) = t_a \cdot |GA| + \sum_{GA' \in CS} t_b \cdot |GA' \cap GA| \quad , \quad (3.32)$$

where t_a and t_b are heuristic time constants. Robotic experiments are required to establish a realistic estimate for the magnitude of these parameters. The evaluation in Chapter 6 gives an indication for these parameters for small teams. To model the cost of reconfiguration for the reference system in this thesis, the default setting of $t_a = 100s$ and $t_b = 600s$ applies. The values are estimates which consider time for additional error handling. The overall reconfiguration cost to transition from a coalition structure CS_i^A to another CS_j^A is defined as:

$$\rho(CS_i^A, CS_j^A) = \sum_{GA \in CS_j^A} \rho(GA, CS_i^A) \quad (3.33)$$

The reconfiguration cost heuristic does not account for relocation cost. Instead, the following assumption holds.

Assumption 3.2 (Precondition for Reconfiguration). *All agents which take part in a reconfiguration process to form a single agent operate in direct proximity.*

Furthermore, real reconfigurable multi-robot systems come with limitations regarding their reconfigurability. For instance, payload items of the reference system cannot self-reconfigure.

They can neither relocate nor have a degree of freedom. Thus, they require external support to attach to other agents.

Assumption 3.3 (Agent self-reconfigurability). *Mobile agents can self-reconfigure, i.e. attach and detach other atomic agents, and permit the reconfiguration of other agents.*

In effect, a transition from CS_i^A to CS_j^A is considered feasible, when all agents operate in direct proximity and at least one mobile agent is present.

Safety In MoreOrg the computation of safety of an agent is based on resource redundancy. Fostering highly redundant agents in an organisation ultimately leads to a single monolithic agent (if that agent is feasible). For the search for an optimal organisation it has to be considered, that due to a high degree of redundancy a safer organisation might be less efficient. Therefore, any optimisation has to trade safety and efficiency against each other. A measure for redundancy is the central part of the safety heuristic and it is based on the standard modelling approach for parallel and serial component-based systems (Rausand and Høyland 2009, pp. 118-125). Each resource can be associated with a probability of survival, so that an overall probability of survival can be computed using a function decomposition tree approach. Information about the probability of survival of components has to be part of an initial system identification and has to be augmented with performance information from real systems. Using redundancy as safety measure follows an assumption regarding failing components.

Assumption 3.4 (Component substitution). *To maintain the functionality of an agent, one component can replace another if it is an instance of the other's class, which also includes instances of subclasses.*

This seems like a strong assumption, since even if components are instances of the same concept (e.g., a camera) it might not be possible to substitute one with the other without losing functionality. However, this is a matter of modelling equivalence classes in the ontology. Hence, according to Assumption 2.5, MoreOrg considers a shared use of resources in a composite agent.

The reliability R_f (also referred to as probability of survival) of a single functionality f can be computed by accounting for parallel components, i.e., resources that are not strictly required but which can serve as replacement:

$$R_f(t) = \begin{cases} 1 - \prod_{i=1}^n (1 - p_i(t)) & \text{parallel system} \\ \prod_{i=1}^n p_i(t) & \text{serial system} \end{cases}, \quad (3.34)$$

where $p_i(t)$ is the time-dependant probability of survival with $0 \leq p_i(t) \leq 1$. Component degrading can be one reason for a change of the probability of survival. MoreOrg leaves the use of time-dependence as future improvement and instead uses a static probability of survival with $t = 0$.

Definition 3.9 (Functional reliability). $R(\mathcal{F}, \widehat{GA})$ denotes the **reliability** of a set of required functionalities \mathcal{F} which is provided by an agent \widehat{GA} .

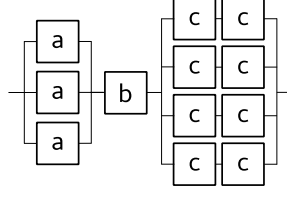


Figure 3.15: Schematic of a system composition consisting of three resource types: a, b, c , where the ratio from required to available is for a 1:3, for b 1:1, and for c 2:8.

The computation of $R(\mathcal{F}, \widehat{GA})$ is based on the functional decomposition of the agent type \widehat{GA} into atomic resources. For each resource a redundancy at component level (cf. (Rausand and Høyland 2009, p. 129)) is assumed. As a heuristic the redundancy is computed based on a type partitioning considering all resources which have no further dependencies. All resources of the same type are modelled as subsystems, which again form a serial system. Figure 3.15 illustrates this modelling approach.

For each subsystem which is composed of a single resource type the redundancy is computed for r required instances, n available instances and the probability of survival p for the resource type:

$$rsub(r, n, p) = \begin{cases} 1 - [(1 - p)^r]^{\frac{n}{r}} & , \text{ where } n \geq r \\ 0 & \text{ otherwise} \end{cases} \quad (3.35)$$

The function REQ maps a set of functionalities \mathcal{F} to the required number of instances for each resource type:

$$REQ(\mathcal{F}) = \{req_1, \dots, req_{|REQ(\mathcal{F})|}\} \quad , \quad (3.36)$$

where req_i represents the minimum cardinality of a resource type i to fulfil all functionalities $f \in \mathcal{F}$.

The function AVL maps an agent type to the number of maximum available resources with respect to a functionality set \mathcal{F} . Only resources that can contribute to the provision of \mathcal{F} need to be considered:

$$AVL(\mathcal{F}, \widehat{GA}) = \{avl_1, \dots, avl_{|REQ(\mathcal{F})|}\} \quad , \quad (3.37)$$

where avl_i represents the maximum cardinality of a resource type i available in the general agent type \widehat{GA} .

Resources lead to a heuristic system structure as shown in Figure 3.15 using serial and parallel systems. Based on this structure, an agent's reliability is defined as:

$$R(\mathcal{F}, \widehat{GA}) = \prod_{i=1}^{|REQ(\mathcal{F})|} rsub(req_i, avl_i, p_i) \quad , \quad (3.38)$$

where p_i represents the probability of survival for a resource type i .

Table 3.8: Resources of the agent type *SherpaTT* which are relevant to provision the functionality *LocationImageProvider*.

Resource	req_i	avl_i	p_i
Localization	1	1	0.95
Locomotion	1	1	0.95
Mapping	1	1	0.95
PowerSource	1	1	0.95
Camera	1	2	0.95

Example A functionality *LocationImageProvider* depends on a functionality *ImageProvider* and a functionality *MoveTo*. The requirements for the *ImageProvider* are one of each resource: Camera and PowerSource. *MoveTo* requires one of each resource: Localization, Locomotion, Mapping, PowerSource.

Table 3.8 lists the cardinalities and probabilities of survival for an atomic agent type *SherpaTT* with relevant resources. This agent type provides the functionality *LocationImageProvider* with a redundant camera system, and otherwise a series system. According to Equation 3.38 the probability of survival for the functionality *LocationImageProvider* is $P = (1 - (1 - 0.95)^2) \cdot 0.95^4 \approx 0.81$. A composite agent which has an additional atomic agent *PayloadBattery* can increase the redundancy of the resource PowerSource by 1. This leads to an increase of the probability of survival for the functionality *LocationImageProvider*, since now two redundant subsystems exist: $P = (1 - (1 - 0.95)^2)^2 \cdot 0.95^3 \approx 0.85$.

Goal Dependant Reliability An agent's reliability is based on information about required functionalities and relevant subsystems, i.e., reliability can be computed with respect to a functionality requirement. The overall goal and objective of an organisation lies in supporting activities of multiple agents. These agents can operate in parallel at different physical locations. The objective for an active coalition structure $CS = \{GA_1, \dots, GA_{|CS|}\}$ of an organisation is described by a corresponding set of functionality sets denoted $FS = \{\mathcal{F}_1, \dots, \mathcal{F}_{|CS|}\}$, and $a2f : CS \rightarrow FS$ allows to map each operative agent GA_i to the required functionality set \mathcal{F}_i . The current redundancy of the organisation is then the minimum achievable level of redundancy:

$$R(FS, CS) = \min_{GA \in CS} R(a2f(GA), \widehat{GA}) \quad (3.39)$$

This redundancy computation is used as safety metric for the planning approach in Chapter 4.

3.5.3 Organisation Performance Metrics

Any performance of the organisation has to be characterised upon analysis of the overall sequence of organisation states. The analysis of the state progression, e.g., as part of simulating the performance and dynamics of the operation of reconfigurable multi-robot systems, and providing a state heuristic for planning with an organisation, can require additional properties which have to be evaluated over time. In coalition games a coalition structure can be described with a single utility. Agents change coalitions to maximise the overall utility. In contrast, MoreOrg uses multiple utility classes. They correspond to the objectives efficacy, efficiency, and safety.

Particularly safety of the organisation is a major concern for space applications. The safety characterisation is based on resource redundancy. Hence, agents with low redundancy shall be avoided during the evolution of an organisation's state. Chapter 4 builds upon this idea and presents an approach to plan with these objectives.

3.5.4 Negative Effects

The default assumption for using composite systems is a linear increase of robustness and functionality. Criticism can be addressed towards this assumption, since an addition of atomic agents to an existing composite agent can also reduce the general functionality: the operational speed of composite agents might be slowed down due to the increase in mass, or due to obstructing morphological changes. Hence, accounting for negative effects in the organisation model is an important feature for practical applications.

MoreOrg allows to account for negative effects based on the addition of maximum cardinality constraints, which reflects the loss of a component. Additional control components which rely on the organisation model can use this expressiveness as follows. An initial definition of atomic agent types exists with the organisation model. The loss of an agent's component changes the agent type correspondingly. Hence, a new agent type needs to be dynamically added as a combination of an existing atomic agent type and a maximum cardinality constraint to account for the loss of this component. Since the same reasoning mechanism applies to all agent types, the effects of a component loss are transparently accounted for.

Mobility is major feature of a robotic agent. Therefore mobility has received special attention within MoreOrg. Any mobile atomic agent with at least one interface to connect to other atomic agents, can act as *transport provider*, i.e., it can operate as marsupial system capable of transporting other agents. Although MoreOrg does not account for geometric constraints when forming composite agents, it permits to define an upper transport limit $cap(\hat{a})$ for each agent. Limiting the transport capacity of an atomic agent can therefore be interpreted as a proactive way to handle negative effects. An extended use of property constraints is an additional way to account for negative effects. Currently, MoreOrg does not use property constraints in the static description of atomic agents types, but only in queries.

3.6 Discussion

The organisation model MoreOrg provides a high-level abstraction regarding the resource structure of atomic agents. Based on a known function decomposition an agent's capabilities can be inferred. MoreOrg uses an existential checking of functionality which is based on identifying the availability of hardware and software components. MoreOrg does not quantify the quality or accounts for a degradation of resources. Hence, the model can further be extended with the quality or degree of suitability of resources in a similar fashion to OMACS (DeLoach 2009). Yet, a prior attribution of the quality might not be feasible for composite systems. A combination with a system which collects and online embeds experiences similar to (Rockel et al. 2013) might be a practical approach. The quality of a component can be added as model property. This information can be directly embedded and used to query suitable agents via MoreOrg's property constraints. In general, the mechanism to identify suitable agents can be used in combination with sophisticated functionality requirements. For instance, property

constraints support the identification of agent types which support a set of selected functionalities and fall below a given weight threshold.

Ontology Usage and Cardinality Constraints An OWL 2 based ontology representation serves as knowledge base for the extensible organisation model. The usage of the general reasoning is limited to rather simple queries such as subsumption and data value assertions. MoreOrg does not exploit the generic reasoner for cardinality constraint-based reasoning. Instead of a consistency checking of ABox instances, the introduced custom reasoning level focuses on an inference of new TBox models, namely the composite agents. This is not a limitation of MoreOrg, but points to unused potential of the included technologies.

Level of Abstraction The chosen abstraction level serves as foundation for a practical approach to reconfigurable multi-robot reasoning which allows to identify the main requirements and needs for such a model. Hence, this thesis explores function decompositions with a maximum dependency tree-depth of two. Although this does not influence the validity of the modelling and planning approach, more complex reasoning structures increase the required computational complexity of the mapping function. Furthermore, MoreOrg relies on the idea that equally typed components can replace each other according to Liskov's substitution principle (Liskov and Wing 1994). Any actual robot that operates with this assumption requires support through adequate high-level (software) control structures. For instance, the dynamic construction of a stereo camera with cameras from two different atomic agents requires a special calibration procedure and a joined data processing infrastructure for both agents.

Shrot, Aumann, and Kraus (2010) make a distinction between strategic equivalence and representational equivalence. In MoreOrg agents of the same type are considered representational equivalent while looking only at the composition in agent space. Full strategic equivalence is achieved, when two agents support the same functionalities. Chapter 4 introduces a mission planning approach that allows for a specification of strategic as well as representational requirements by combining functionality with agent type constraints.

Coalition games look at single agent properties and hide a collection of resources behind a single scalar. In contrast, MoreOrg provides an explicit description of atomic agents, and introduces a number of properties that characterise atomic as well as composite systems. MoreOrg finds a middle ground between an approach like OMACS and purely coalition based theory: it uses multi-agent systems and focuses on physical agent coalitions. Loosely coupled coalitions can still be modelled with MoreOrg: interface types which have no physical counterparts can be associated with each agent. These can either represent actual software interface types or an interface which solely restricts the general ability to cooperate in multi-agent systems.

Functional Saturation Functional saturation and the use of minimal agents reduces the problem of combinatorial explosion. The approach is, however, currently limited with respect to using property constraints. The computation of the functional saturation bound is based on the idea to reduce the number of composite agents that need to be considered for forming a composite agent with the given functionality. Some constraints might require to increase the number of agents and redundancy, e.g., to meet mass, probability of survival, or energy efficiency requirements.

The functional saturation bound is not exact. The current algorithm to compute the bound, looks at each agent type separately. The algorithm might identify the functional saturation for a function f at ten agents of type \hat{a}_0 and ten agents of type \hat{a}_1 . Still, a combination of seven agents of type \hat{a}_0 and four agents of type \hat{a}_1 could suffice. The current bound leaves redundancies in the resulting agent pool. This is accepted in this thesis as a compromise to maintain a simple, yet effective bounding mechanism.

Connectivity Checking The compatibility of interfaces and possible links between two atomic agent are analysed for connectivity checking. Although some morphological configurations will not be possible, connectivity checking does not account for geometric constraints of a composite agent. For composite agents that are composed of a large number of atomic agents the identification of a feasible morphological structure can be a challenge. The organisation model should therefore be improved with the addition of a geometric constraint solver, where the connectivity checking can serve as preprocessing stage.

State Description The safety metric embeds system reliability theory into MoreOrg and it is based on knowledge about the structural composition of atomic and composite agents. The selected metric to describe the probability of survival in MoreOrg is based on redundancy. Using the level of redundancy to describe the safety of a system is complementary to the application of the functional saturation bound which tries to reduce redundancy. The integration of probability of survival illustrates a general mechanism to support safety and fault management strategies with an organisation model. The use of a redundancy is thereby one available metric, while information entropy and failure count are among the potential alternatives. In general, reliability theory serves as basis for the safety metric. But due to the selected abstraction level a complementary empirical validation is still required to identify a realistic and practical measure. Hence, redundancy only offers a starting point for even more sophisticated and empirically validated models.

As result of the previously mentioned abstraction level MoreOrg comes with a limited level of modelling detail. For instance, an application of functionality might come with an agent specific energy consumption or energy consumption profile even. MoreOrg does not consider such detailed quantification of energy consumption of functionalities except for transport. An inclusion of further agent properties can be part of a more precise state description.

3.7 Summary

This chapter introduces Model for Reconfigurable Multi-Robot Organisations (MoreOrg) as an organisation modelling approach dedicated to reconfigurable multi-robot systems. The organisation model considers the modelling levels: atomic agent, composite agent and the overall organisation. The organisation model is the basis to infer functionalities and properties of composite agents by analysing the aggregation of resources as result of merging multiple atomic agents. The use of (composite) functionality is not only based on strong cooperation, which is defined by R. Brown and J. Jennings (1995) through the need for agent synchronisation. It is also based on a tight coupling of agents. Thereby MoreOrg narrows a research gap for multi-robot systems which are capable of physical reconfiguration.

The organisation model uses an ontology-based representation approach in order to exploit international standardisation and to support open extensible agent architectures. The existing generic reasoning is complemented with special reasoning strategies which use a qualitative constraint-based modelling of composite resources. This special reasoning approach is the basis for a dynamic representation of composite agents types.

Since connectivity of atomic agents is restricted, MoreOrg introduces an approach to identify feasible agents. The approach is suitable to validate large agent compositions. A functional saturation bound reduces the combinatorial challenge when dealing with atomic agents. The bound is represented by a set of minimal agents which can support a particular functionality. The application of the functional saturation bound effectively prunes agents which contain redundancies. In effect, MoreOrg provides a compact mapping between functionality and agents and thereby provides key functionality to enable planning for reconfigurable multi-robot systems.

MoreOrg is a merge of best practices and existing knowledge in the areas of multi-agent research, organisation theory, knowledge-based systems and robotics research in order to establish a representation for reconfigurable multi-robot systems. The organisation model serves as general back-end and knowledge database for planning with reconfigurable multi-robot systems in this thesis. MoreOrg is not only useful for automation. It also provides a means to systematically explore options that reconfigurable multi-robot systems offer, e.g., the exploration of connectivity in a reconfigurable multi-robot system. Previously not anticipated morphological constellations can be identified and it implements a constraint and model-based search as alternative to evolutionary approaches. A guided exploration through agent coalition structures can therefore lead to new options for performing robotic missions. In summary, MoreOrg represents the basis to exploit the flexibility of reconfigurable multi-robot systems.

4

Mission Planning

Disclaimer *This section introduces and expands on works which have been published in (Roehr 2018; Roehr and F. Kirchner 2016).*

Up to now reconfigurability with respect to high-level action planning and optimisation has received little attention in research. The existing approaches tackle subproblems, such as planning a single transition between two configuration states or finding an optimal coalition structure. Up to now, no planning approach existed which considers reconfigurable multi-robot systems. To close this research gap this chapter introduces the planner Temporal Planning for Reconfigurable Multi-Robot Systems (TemPl) as a mission planning approach for reconfigurable multi-robot systems. TemPl embeds the organisation model MoreOrg to reason about reconfigurable systems. MoreOrg permits the planner to exploit the flexibility of reconfigurable robotic systems. Planning for reconfigurable systems is a challenging task since an exponentially large number of agents can be created from a given set of atomic agents. The focus on minimal agents, i.e. those agents which have minimal redundancy and still support a required functionality, leads to a significant reduction of the considerable number of agents which need to be considered. Hence, the introduction of effective bounding strategies is one of the central concepts within TemPl.

The following Section 4.1 gives relevant background information on related approaches and technologies which have been included in the planning approach or have influenced the finally implemented planning approach. Section 4.2 formally introduces the mission planning problem and details the developed solution approach. The known existing limitations of the planning approach are discussed in Section 4.5, followed by a summary in Section 4.6.

4.1 Background

The following sections provide the background knowledge for the mission planner TemPl which has been particularly designed for reconfigurable multi-robot systems. TemPl represents a highly integrated approach, which relies on the complementary use of state of the art technologies. While the planning approach does not extend any of the basic technologies, it does provide a unique, interdisciplinary approach involving the following set of technologies: (a) knowledge-based reasoning, (b) temporal reasoning, (c) Vehicle Routing Problems (VRPs), (d) constraint-based programming, (e) linear programming, (f) and heuristic search.

A science driven mission design serves as motivation for the mission planning approach performed with a reconfigurable multi-robot system. A human operator can define the scope and activities of a robotic mission, while the number of available atomic agents is limited. The mission can be either defined by exact activities of atomic agents or by high-level functional requirements. Therefore, TemPl is based on a constraint-based robotic mission specification which collects major mission constraints and defines the general goals. The specification will be used as a combined domain and problem definition to perform planning and optimisation of the robotic mission.

The organisation model MoreOrg as presented in Chapter 3 introduces knowledge-based reasoning into the planning approach. The organisation model provides the means to identify suitable agents which fulfil functionality requirements, and allows to define the properties of atomic agents, composite agents and the overall organisation. The ability to change the coalition structure is the key feature of a reconfigurable multi-robot system. But a coalition structure change requires synchronisation of agents, so that the consideration of time and temporal reasoning is essential for the planning approach. Temporal constraint networks (TCNs) (Rina Detcher 2003) provide an appropriate means to work with time in a qualitative and a quantitative way and have therefore been included into the planner's design. The resulting planning challenge lies in a highly combinatorial problem in combination with side-constraints. Constraint-based programming is a well established technology and suited to tackle problems as the given one.

The overall mission planning problem is strongly linked to the domain of operations research and especially so-called vehicle routing problems: mobile agents can be viewed as vehicles, and immobile agents as commodities or goods which need to be transported. Reconfiguration can then be considered as a transshipment from one to another vehicle. This permits the formalisation as multi-commodity min-cost flow optimisation problem which is solved with linear programming in TemPl. Heuristic (local) search techniques are also commonly applied in the context of VRP to cope with searching through large neighborhoods of a given feasible or infeasible solution (see (Toth and Vigo 2014)). Although local search can be started on either a feasible or an infeasible solution to the mission planning problem, many local search approaches require a feasible solution. For the given planning problem, however, the primary challenge lies in generating at least one feasible solution.

4.1.1 Vehicle Routing Problem

The VRP represents a classical optimisation problem typically dealt with in the area of operations research: a number of customers have a demand with respect to goods or services and a

set of vehicles exists to perform one or multiple tours to satisfy these requests. In its simplest form solving a VRP intends not only to identify a feasible solution, but finding the solution with lowest cost - often defined by the length of the overall route. A typical restriction is a limitation of visits, so that a customer can only be served once. VRPs are a generalisation to the Travelling Salesman Problem (TSP), a classical combinatorial optimisation problem (Ahuja, Magnanti, and Orlin 1993), i.e., the TSP can be found as a subproblem in a VRP.

In the TSP a salesman has to visit a number of customers, each of which resides in a different city. The optimisation problem consists of finding the optimal route, i.e., lowest cost tour, to serve all customers under the constraint that the salesman visits each city (customer) only once. The TSP deals with finding one route only, but in a VRP multiple vehicles have to be considered which can deliver their goods to customers. Therefore, an initial assignment of vehicles to customers is required to subsequently solve a number of TSPs. A graph $G = (V, E)$ can serve as representation for a VRP, i.e., by defining a network flow (see (Ahuja, Magnanti, and Orlin 1993)) where a vertex $v \in V$ represents a customer with a particular demand q_v , and an edge $e \in E$ represents the transition of a vehicle from one customer to another. The graph can be directed when transport cost between two vertices are asymmetric, or undirected if they are symmetric. Additional constraints can be applied to either vertices or edges, e.g., limiting or requiring an in and out-flow of a vertex, or limiting the flow capacity for an edge to encode the maximum transport capacity of a vehicle.

Variants of the VRP allow for a detailed, often application specific formalisation and description of the problem, e.g., in order to cover practical issues such as time-based delivery or the loading constraints of the vehicles. While some of these variants support a systematic analysis of solution approaches, they come with a reduced applicability, e.g., when considering a fully homogeneous vehicle fleet. Toth and Vigo (2014) provide a broad overview. Meanwhile, the following section details the relevant variants for this thesis.

Problem variants and models

The Capacitated VRP (CVRP) is one of the most popular VRP variants. It accounts for a limited transport capacity of individual vehicles. A fleet of homogeneous vehicles is assumed, i.e., all vehicles have the same capacity and operating cost. All vehicles start and end at a single depot. To solve the CVRP a variety of representations has been developed, where each representation comes with limitations for its applicability: directed and undirected two-index vehicle-flow formulations (referred to as VRP1 and VRP2 by Toth and Vigo (2014)) use only single edges between any two vertices - the directed formulation use one edge in either direction. As a major disadvantage, these formulations cannot embed vehicle specific solution information and thus cannot explicitly take vehicle characteristics into account. However, these two-index vehicle flow formulations avoid redundant solutions which result from vehicle permutation. The alternative is a three-index vehicle-flow formulation (referred to as VRP3 by Toth and Vigo (2014)). The third, vehicle-specific index permits the direct identification of a vehicle which is routed along a particular edge. As a drawback VRP3 leads to the computation of highly redundant solutions.

The problem of vehicle identification is similarly encountered in TemPl. Section 4.3.3 refers to the problem in the context of the identification of atomic agent instances. The problem is tackled with a complementary use of symmetry breaking and an explicit multi-commodity min-cost flow formulation (see Section 4.3.5).

Another so-called extensive formulation focuses on identification of the best combination of actually feasible routes. Instead of defining the vehicle-flow and edge variables explicitly, the extensive formulation uses variables to represent complete routes. Additionally, it associates each route variable with a binary variable. This binary variable defines whether a route is part of the solution or not. The main benefit of this representation lies in lower computational bounds and the possibility to embed complex feasibility constraints for each route. The extensive formulation increases, however, the combinatorial challenge. A full enumeration of all feasible routes and combinations thereof is only feasible for very small problem sizes. Hence, an implicit handling of routes is required and routes for candidate solutions have to be dynamically generated as part of the optimisation process, e.g., using column generation (Desrosiers and Lübbecke 2005). TemPl uses a similar dynamic route generation approach. It generates the (full) routes for all mobile agents and partial routes for immobile agents using a constraint-based solution approach. The local search then tries to identify a valid solution including the full routes for immobile agents.

The VRP with Time Windows (VRPTW) is an extension of the CVRP which adds time windows to define when a customer requires its demand of goods to be fulfilled. Time windows can be defined as either hard or soft constraints. The violation of a soft constraint does not invalidate a solution, but involves a penalty and increases the solution cost, e.g., when a good is delivered early or late. In effect, any hard constraint is a soft constraint with infinite cost penalty.

The VRP with Pick-up and Delivery (VRPPD) is also known as Dial-a-Ride Problem (DARP) and represents a public transportation problem: passengers request a ride from one location to another while the pickup and delivery interval can also be constrained. Most of the classical problem instances come with significant limitations for their practical use, e.g., due to assuming a single depot or homogeneous vehicles. Problem-rich variants of higher practical relevance such as Multi-depot heterogeneous fleet VRP with time windows, VRPTT and VRP with multiple synchronization constraints (VRPMSs) exists, but are rarely handled in research. However, these problem rich variants are of particular interest for this thesis. They comprise main characteristics encountered in an application of reconfigurable multi-robot systems including, although not limited to the following constraints: (a) multiple depots, (b) heterogeneous vehicles, (c) time windows, (d) vehicle capacities, (e) vehicle synchronisation, (f) pre-emptive drop-off, (g) and pick-up. The fleet of vehicles and its partitioning into same type vehicles directly corresponds to agent types in the organisation model (see Chapter 3). The organisation model focuses on agent properties such as nominal speed, power consumption, and capacity for transporting other agents. Additional fleet characteristics can involve the locomotion capabilities of agents which can be combined with route characteristics such as traversability.

The VRPTT deals with trucks and trailers, which can be linked to each other in order to form a transport unit. In contrast to classical VRPs definitions, the problem embeds location accessibility constraints, e.g., by accounting for limited manoeuvring space for vehicles or similar. Though this type of constraint is not explicitly handled in TemPl, Section 4.2 shows how these constraints can be introduced or accounted for; showing the increased generality of the problem formalisation. The specialised VRPTT and the more generalised VRPMSs (Drex1 2013) describe problems that deal with vehicles' interdependence. This interdependency arises through the need for synchronisation of tasks or operations which involve more than one vehicle. Since this is a typical characteristic of any coordinated multi-agent operation, VRPMSs are strongly relevant for reconfigurable multi-robot system operation. Due to the interdependence of vehi-

cle operations, individual routes cannot be optimised in an isolated way, but require so-called inter-route constraints. To tackle the problem, Drexler (2013) focuses on the modelling of the problem using a graph-based approach, while initially leaving the development of appropriate solution approaches open. Parragh and Cordeau (2017), Tilk et al. (2018) and Rothenbächer, Drexler, and Irnich (2018) solve the problem using a combination out of branch, price and cut algorithms.

Drexler identifies orthogonal categories for vehicles: the first category comprises autonomous and non-autonomous, and the second category task and support vehicles. The latter are due to the particular application scenario of milk collection, where customer visits are permitted only for a limited set of vehicles. The remaining vehicles can still be used to support the transport, e.g., by creating intermediate transshipment hubs. Drexler further differentiates between active and passive vehicles. Passive vehicles such as trailers provide additional load capacities. For a location change they depend, however, on an active vehicle such as a lorry which can pull trailers. The general modelling approach remains domain specific in parts due to this categorisation which is narrowed to trucks (here lorries) and trailers. Thus, it leaves room for further generalisation particularly in the context of reconfigurable multi-robot systems.

Drexler collects constraints for VRPTTs and VRPMSs which can broadly be categorised into: (1) location access constraints: constrain the parallel or overall access to a location by vehicle number and/or by vehicle type, (2) location transfer constraints: constrain the transshipment of commodities by participating vehicles, by the current and overall amount per location, and (3) time constraints: intervals within which the transshipment of goods must be completed. Drexler splits customers into lorry customers and trailer customers. These subcategories encode a permission of a lorry to perform a visit with or without a trailer - this could also be modelled using location access constraints. The VRPMSs have no limitation on the starting depots for the vehicles, and no constraint exists on the lorry trailer combination which returns to a depot. Autonomous and non-autonomous vehicles are considered in this problem formulation. A final solution has to synchronise the operation of these vehicles.

Inter- and Intra-Route Constraints In basic VRP variants vehicles can operate independently from each other. Each vehicle's route has only to be consistent with existing so-called intra-route constraints. Intra-route constraints can be seen as local constraints. Hence, a solution can be quickly invalidated when a local, i.e. route-based, consistency check fails. Typical intra-route constraints as mentioned by Toth and Vigo (2014) are collected in Table 4.1. In contrast to intra-route constraints, inter-route constraints are global, and can therefore only be verified when all routes for a potential solution have been constructed. Typical inter-route constraints as mentioned by Toth and Vigo (2014) and Drexler (2013) are illustrated in Table 4.2.

4.1.2 Planning

Planning and scheduling are strongly related. Planning tries to identify the actions to be taken to solve a problem, while scheduling tries to identify a feasible (temporally valid) schedule for a known set of activities and a limited set of available resources. Plan optimisation typically targets shorter plans with respect to the number of actions. In contrast, scheduling mainly tries to minimise the so-called makespan, i.e., the overall timeframe for plan execution. Real world problems often require a mixture of planning and scheduling.

Table 4.1: *Intra-route constraints in VRPs as described by Toth and Vigo (2014).*

Category	Constraint	Description
loading	capacity	simple load capacity, weight or mass constraints, as well as space related constraints as 2- or 3-dimensional packing problem
	item clustering	items for the same customer must be shipped with the same vehicle
	item orientation	items must be kept in a restricted alignment
	sequential loading	goods need to be packed such that goods for later customers need not to be rearranged to get the goods for the current customer
	compartments	partitioned overall vehicle capacity, use of compartments based on items' compatibility
numeric route properties	(spatial) distance / link length	length of a route
	duration	travel time for a route including wait times
	traversability cost	a route can be associated with cost for traversal
multiple vehicles	vehicle reuse	reuse might be desired or required due to small vehicle capacities
time windows & scheduling	time windows (hard, soft)	time frame for performing delivery or service with or without penalty
	driving period restrictions	originating from general traffic regulations, e.g., to limit a truck drivers working time
	ridetime constraint	restrict the travel time for passengers that share the ride

Table 4.2: *Inter-route constraints in VRPs as described by Toth and Vigo (2014) and Drexler (2012, 2013).*

Category	Constraint	Description
balancing	route duration	route duration of vehicles should not exceed a given threshold
resource limit	location access constraints	limit number of vehicles at a location
	route properties	limit number of routes that meet a particular property constraint
synchronisation	tasks	execute tasks at designated locations and times
	operation	require presence of specific vehicles at designated locations and times
	movement	movement of commodities or non-autonomous systems requires a mobile, autonomous vehicle
	load	account for vehicles' capacity constraints
	resource	account for a general resource utilisation and in particular vehicle utilisation needs to be accounted

The classical planning problem as described by Ghallab, Nau, and Traverso (2004) does not consider time and represents a state transition system. A state describes facts which are relevant to solve the planning problem. A state can be changed by the application of a suitable action, where an action represents the instance of an operator. An operator specifies precondition, postconditions, and in some representations like SAS+ (Bäckström and Bernhard Nebel 1995) also prevail conditions; preconditions define when an operator is applicable to a planning state and postconditions what changes result from an operator. Preval conditions need to hold for the full timeframe to which the operator is applied. An operator uses a set of variables as parameters, and these variables have to be bound to instantiate the operator. An instantiated operator represents an applicable action. Planning tries to identify a feasible sequence of actions to achieve a goal state, which is defined as a partial or full state description. To find the sequence of actions, forward search or backward search can be used: forward search starts at the initial state and applies actions to modify the state until a goal state has been reached (or the search is aborted). In contrast, backward search starts at the goal state and tries to identify actions which can lead to this goal state and originate at the initial state. Since both approaches suffer from the branching factor during search, heuristics are typically used to guide the search.

Classical planning has been strongly influenced by planners such as Fast Forward (Hoffmann and Bernd Nebel 2001), Fast Downward (Helmert 2006) and LAMA (Richter and Westphal 2010). These planning systems represent a subsequent development that has brought forward and tested various ideas for tackling NP-complete planning problems with heuristic search. Fast Forward initiated this process by introducing a heuristic that is based on an actual planning algorithm itself: Hoffmann and Bernd Nebel (2001) use GRAPHPLAN (Blum and Furst 1997) as core of their Fast Forward's planning heuristic; they relax the planning task for GRAPHPLAN by ignoring delete effect when computing the heuristic value. Hoffmann and Bernd Nebel's planner implementation which is based on a Hill-Climbing search and which uses the GRAPHPLAN-based heuristic proved to be successful. Fast Downward's main contribution was the introduction of the Causal graph (CG) Heuristic based on the use of Domain Transition Graph (DTG). DTGs describe the feasible transitions of a state variable through its value domain, e.g., for a variable x with $D_x = \{0, 1, 2\}$ the DTG depends upon the set of applicable operations and a node in the DTGs represents the value of the variable and directed edges are annotated with the operations that trigger a change from this value to the value of the target node. Edges are annotated with additional conditions of a transition, e.g., such as the particular value of another variable, here y . Such a condition shows the direct dependence of the variable x upon a particular value of y , a relation that is encoded by a connecting edge in the CG. The CG can be used to compute the cost to change the variable x from a value d to a value d' , resulting in an efficient heuristic to estimate the cost of changing variables that have a causal relationship. LAMA meanwhile builds upon Fast Downward and uses additionally a landmark heuristic by identifying an invariant state in the search space, i.e., landmarks represent a grounding of a propositional formula that has to be encountered during a planning process.

Various representation standards have been developed including, but not limited to Planning-Domain Definition Language (PDDL) (M. Fox and Long 2003). Particularly PDDL helped to facilitate the comparison between planning systems on the International Planning Competition (IPC). PDDL allows to represent the planning domain and the planning problem in a standardised way. The expressiveness of PDDL extends to the definition of actions with duration, and can therefore also be used for temporal planning problems. LAMA and FastDownward are PDDL-based planning systems. Helmert (2006) has developed a standard process template

for the planner implementation which involves three steps: (1) translate, (2) preprocess, and (3) search. The first *translate* phase transforms PDDL into the intermediate representation, here SAS+ (Bäckström and Bernhard Nebel 1995) is used. Preprocessing involves the preparation of heuristics, i.e., generation of the CG and DTG, while the search is based on a best first search approach using the prepared heuristics.

A PDDL-based Representation Approach

An application of a PDDL-based planning approach has been considered based on the explicit generation of agent types by Roehr and Hartanto (2015), where the problem domain has been defined with operators shown in 4.1. The domain definition does not account for any capacity constraints, but allows to define atomic and composite agent relationships through the predicate *embodies*. Operator *moveto* leads to a change of location for atomic and composite agents, while *join* and *split* reflect a reconfiguration activity which takes place at a single location. To simplify the operators *split* and *join*, a full decomposition of a composite agent into individual atomic agents is assumed for reconfiguration either as result or as precondition; in practice or rather at execution level this assumption should not turn into a requirement, but dealt with using a dedicated local transition planner. The operator definition requires a set of predicates listed in Table 4.3 (see also (Roehr and F. Kirchner 2016)):

Table 4.3: *Predicates as part of the domain definition.*

Predicate	Description
<i>atomic(a)</i>	agent <i>a</i> is atomic
<i>operative(a)</i>	agent is operative meaning, if agent <i>a</i> is composite it is assembled and otherwise if agent <i>a</i> is atomic it is not part of any composite agent
<i>at(a,l)</i>	agent <i>a</i> is at location <i>l</i>
<i>embodies(c,a)</i>	composite agent <i>c</i> embodies an atomic agent <i>a</i> (if <i>c</i> is operative)
<i>mobile(a)</i>	agent <i>a</i> is mobile and thus can move by itself
<i>provides(a,f)</i>	agent <i>a</i> offers a functionality <i>f</i>

The domain definition accounts for the mutually exclusive sets of operative and dormant agents (see definitions in Chapter 2.4); dormant equals the negation of operative, so that all agents which are not explicitly marked as operative are dormant.

Transformation into PDDL The operator definition in Figure 4.1 can be directly translated into PDDL. The representation uses quantifiers and disjunctions in preconditions and effects. Therefore a planner requires support for Action Description Language (ADL) (Pednault 1987) to be applicable to this domain representation.

For any agent symbol an operator instance aka an action has to be generated as part of the operator grounding process. Therefore, the problem definition requires an exhaustive listing of all available agents, i.e., atomic and composite as well as dormant and operative agents. The need for this explicit representation is a significant limitation of a PDDL-based planning approach. It limits the scalability of an application for reconfigurable multi-robot systems. The listings in Appendix F provide examples of domain and problem samples which have been automatically generated from an organisation model. The transformation example shows, however, that a

moveto(a, l_s, l_t) – move agent a from start l_s to target l_t
preconditions: $mobile(a) \wedge operative(a) \wedge at(l_s) \wedge \neg at(l_t)$
effects: $at(l_t)$

join(c, l) – construct the composite actor c at location l
preconditions: $\forall z \in A : \neg atomic(c) \wedge \neg operative(c) \wedge$
 $((embodies(c, z) \wedge operative(z) \wedge at(z, l))$
 $\vee \neg embodies(c, z))$
effects: $\forall z \in A : at(c, l) \wedge operative(c) \wedge$
 $(\neg embodies(c, z) \vee (\neg at(z, l) \wedge \neg operative(z)))$

split(c, l) – split the composite actor c at location l
preconditions: $operative(c) \wedge at(c, l)$
effects: $\forall z \in A : \neg operative(c) \wedge \neg at(c, l) \wedge$
 $(\neg embodies(c, z) \vee (operative(z) \wedge at(z, l)))$

Figure 4.1: Operations as part of the domain definition, for a set of atomic agent A and location variables l, l_s, l_t (based on Figure 4 in (Roehr and F. Kirchner 2016)).

direct use of PDDL-based planners is unsuited for planning with reconfigurable multi-robot systems even for medium sized robotic organisations.

4.1.3 Hierarchical Task Networks

Hierarchical Task Networks (HTNs) are an alternative planning approach to the classical planning (Ghallab, Nau, and Traverso 2004). HTN planning is based on the search for a suitable decomposition of a high-level task into a sequence of primitive tasks. The decomposition is controlled through methods which are part of the planning domain description. A method defines preconditions, as well as a list of subtasks along with an optional list of ordering constraints; for primitive tasks, the list of subtasks will be empty. HTN-based search can be very efficient compared to classical planning approaches. This results from the exploitation of a hierarchical task structure and the ability to embed domain specific knowledge into methods and tasks. However, HTN planning depends upon a significant modelling effort which is needed to define decomposition methods.

Stock (2016) and Stock et al. (2015) combine HTNs with a constraint-based planning approach in the planner CHIMP, which has also applied in the context of reconfigurable multi-robot systems as part of the project TransTerrA (cf. Section 2.2, (Stock 2016, pp. 110-111)). The application in the reconfiguration context is based on an explicit modelling of reconfiguration tasks, e.g., to consider the exchange of payload items or an explicit pickup of a base camp. Stock suggests to use the planner for detailing a mission execution plan, which will be computed by the planner TemPl. Stock et al. use the framework Meta-CSP (Pecora 2018) to represent and apply constraints which are semantically related in combination as meta-constraint. The constraint network serves as basic representation of the planning problem and in order to combine the approach with HTN decomposition, each application of a HTN method is translated into a meta-constraint which is then applied to the constraint-network.

HTN planning has also been applied in combination with knowledge-based approaches by

Hartanto (2009). Hartanto combines HTN planning and DL by representing a planning domain and problem in a DL language. The problem domain is filtered using DL reasoning, so that facts which are irrelevant to the current planning problem are removed. The approach reduces the planning search space in the preprocessing stage of the planning. Hartanto additionally motivates the usage of a knowledge-based representation of the planning domain with a gain in usability and better accessibility of the achieved modelling to users outside the planning community. KnowRob is a further knowledge-based system developed by Tenorth and Beetz (2013) which is used for robotic planning. The planning approach builds on the Cognitive Robot Abstract Machine (CRAM) lightweight reasoning language (Beetz, Mösenlechner, and Tenorth 2010) which intends to avoid predefined and mostly static control recipes. To increase the applicability of plans, Beetz, Klank, et al. (2011) use KnowRob and CPL to define plan recipes which contain variables. These variables are grounded as close as possible to the time of application of the plan, so that variables can reflect the latest, and hence most accurate information.

4.1.4 Temporal Planning and Dealing with Time

A representation of time is basis for temporal planning. A distinction can be made between quantitative and qualitative representations of time. The qualitative handling of time deals with relative relations between timepoints or intervals, and the commonly applied techniques are point algebra (PA) (Vilain, Kautz, and Beek 2013) and interval algebra (IA) (Allen 1990). PA uses a set of relations $REL = \{>, <, =\}$ in order to qualitatively describe the relation between two timepoints. A TCN (Rina Detcher 2003) collects the constraints between a set of timepoints and allows to check a corresponding network for consistency; the constraint network is a directed graph $G = (V, E)$, where $v \in V$ represents a timepoint and an edge $e \in E$ represents a constraint between its source vertex and the target vertex. Path-consistency checking is often the basis to verify if the overall set of constraints is consistent. In contrast to PA, IA describes relations with a disjunctive combination from a set of 13 relations including *equal*, *before*, *meets*, *overlaps*, *during*, *starts*, *finishes* (and their inversions). IA is more expressive compared to PA since it can handle disjunctive relation, e.g., allowing it to describe two non-overlapping intervals. This cannot be expressed with PA because it requires more than only binary relations between timepoints (see (Vilain, Kautz, and Beek 2013)). Despite this apparent disadvantage of PA it comes with a lower computational complexity, and has therefore been selected for the implementation of the planner TemPl.

Simple temporal networks (STNs) (Rina Detcher 2003) are the basis for a qualitative handling of time. An STN is a graph-based representation of temporal constraints where variables represent timepoints, and weighted edges represent the allowed duration for transitioning from one timepoint to another. Weights on edges which point forward in time, are positive, while weights which point backward in time are negative (see Figure 4.2). Solving an STN and checking for consistency is based on the computation of the distance graph, e.g., using Floyd-Warshall's all-shortest path algorithm (see (Cormen et al. 2009)) and an identification of a negative cycles. If a negative cycle exists, the STN is invalid, and otherwise the result of the constraint propagation can be extracted from the distance graph. Dealing with multiple interval constraints between timepoints leads to general TCNs which operate with a set of STNs. Since finding a consistent set of interval constraints is a combinatorial challenge which suffers from interval fragmentation, an application of approximation algorithms such as Loose-Path Consistency and Upper-Lower Tightening has been suggested by Schwalb and Dechter (1997).

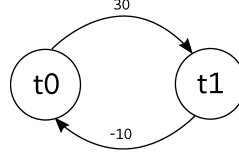


Figure 4.2: A temporal constraint describes the lower and upper bound for the time distance between two timepoints, here a transition between timepoint t_0 and t_1 requires at least 10 and at most 30 (time units).

These algorithms do not solve the constraint satisfaction problem, but reduce the set of disjunctive intervals that need to be considered by an actual solver.

Meiri (1996) combines qualitative and quantitative representations of time into a general TCN representation. The basis of his approach is a translation of qualitative constraints into the qualitative domain which uses transition intervals as constraints. For each of the existing qualitative relations $R = \{<, =, >\}$ and two timepoints t_a and t_b Meiri defines the following implied constraints $X_{t_a \rightarrow t_b}$ for the transition from t_a to t_b .

$$\begin{aligned} t_a < t_b &\implies (0, \infty) \in X_{t_a \rightarrow t_b} \\ t_a = t_b &\implies [0] \in X_{t_a \rightarrow t_b} \\ t_a > t_b &\implies (-\infty, 0) \in X_{t_a \rightarrow t_b} \end{aligned}$$

In order to identify a suitable approach to deal with qualitative time three this thesis has applied three different approaches:

- the author's own PA implementation, based on the algorithms outlined by Dechter (2003),
- an application of the Generic Qualitative Reasoner (GQR) (Gantner, Westphal, and Wölfl 2008), and
- the author's own PA implementation, using Gecode.

The PA based implementation has a worst-case complexity of $\mathcal{O}(n^3)$ and requires already multiple seconds to validate a relatively small network with only 60 timepoints. In contrast, GQR and the custom constrained-based implementation can operate with hundreds of timepoints, and Figure 4.3 shows an exemplary comparison of the performance of each approach for computing a consistency check. The Gecode-based implementation clearly outperforms the GQR implementation.

According to Ghallab, Nau, and Traverso (2004), a temporal database $\Phi = (\mathcal{T}, \mathcal{X})$ is the basis for temporal planning, where \mathcal{T} is a set of *temporally qualified expressions* and \mathcal{X} represents a set of temporal constraints as well as other object variable constraints, e.g., such as binding constraints allowing to set values or enforce equality between objects. A temporally qualified expression $f(o_1, \dots, o_n)@[t_s, t_e]$ describes a fluent which holds for a time interval from t_s to t_e ; the fluent f denotes a condition for a set of object variables o_1, \dots, o_n . For the database to be consistent, a feasible assignment of object variables has to exist, for which the fluents and the set of constraints in \mathcal{X} hold. An example database for the operation of a multi-robot system can look like the following:

$$\begin{aligned} \Phi = (&\{at(a_0, \text{lander})@[t_0, t_1], at(a_1, \text{lander})@[t_0, t_1], \\ &at(a_0, l_0)@[t_2, t_3], at(a_1, l_1)@[t_3, t_4]\}, \\ &\{t_0 < t_1 < t_2 < t_3 < t_4\}) \end{aligned}$$

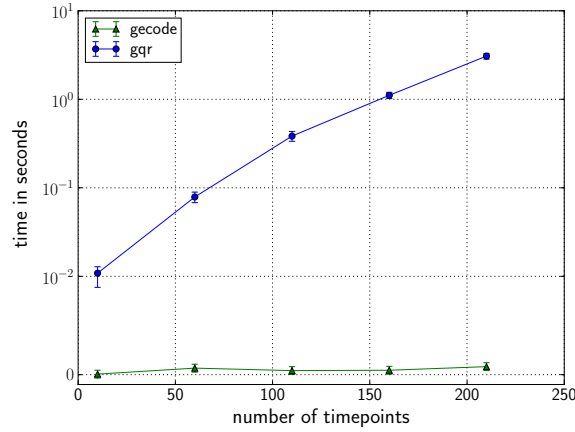


Figure 4.3: Comparison of the consistency checking for a temporal constraint network $N = (V, E)$ with a predefined number of timepoints $|V|$, and constraints between every two timepoints so that $|E| = 0.5 \cdot |V|$.

Two instances agent a_0 and a_1 are required at an initial location *lander* and subsequently are at locations l_0 and l_1 .

The temporal database Φ is a description of the evolution of states. It can be used to represent the initial state of the system as well as intermediate states and the goal state. A planning process has to refine and augment the initial temporal database through an application of temporal planning operators requiring preconditions and effects. In contrast to the classical planning approach applying a temporal planning operator also adds constraints to the temporal database. Since negative effects cannot be encoded, domain axioms have to be defined. For the robotic domain, for instance, axioms have to require a robot to be at one location at a time only.

The temporal database representation is the basis for the mission planning approach developed in the remainder of this chapter. The temporal planning approach as suggested by Ghalab, Nau, and Traverso (2004) still requires an explicit representation of participating agents. Hence, it requires adaptation to be used for a reconfigurable multi-robot system planning approach and dynamically created agents.

4.2 Robotic Missions

Space exploration missions are motivated by science goals and robots as means to analyse remote environmental conditions, e.g., by taking soil samples, performing in-situ analysis of these soil samples or even by preparing the returning of sample to Earth. The existing robotic Mars missions use single rovers such as Spirit, Opportunity and Curiosity (NASA Jet Propulsion Laboratory 2018), so that the robot performing a particular task is clearly defined. As soon as multiple robots are present dynamic task allocation can be used, e.g., based on which robot is available and best suited for the job. So most importantly the scientific tasks and their inter-dependencies have to be encoded into a mission specification; the used resources, here robots, should be dynamically assigned.

The mission specification, which is described in the following sections, is based on this idea

of a task-oriented goal definition and dynamic resource assignment. Furthermore, the task-oriented approach can be mixed with a means-oriented approach as a result of applying temporal planning; the means-oriented approach has its focus on how goals are achieved and uses temporal constraints to restrict possible solutions. Meanwhile the task oriented approach defines intermediate and final states that shall be part of a robotic mission.

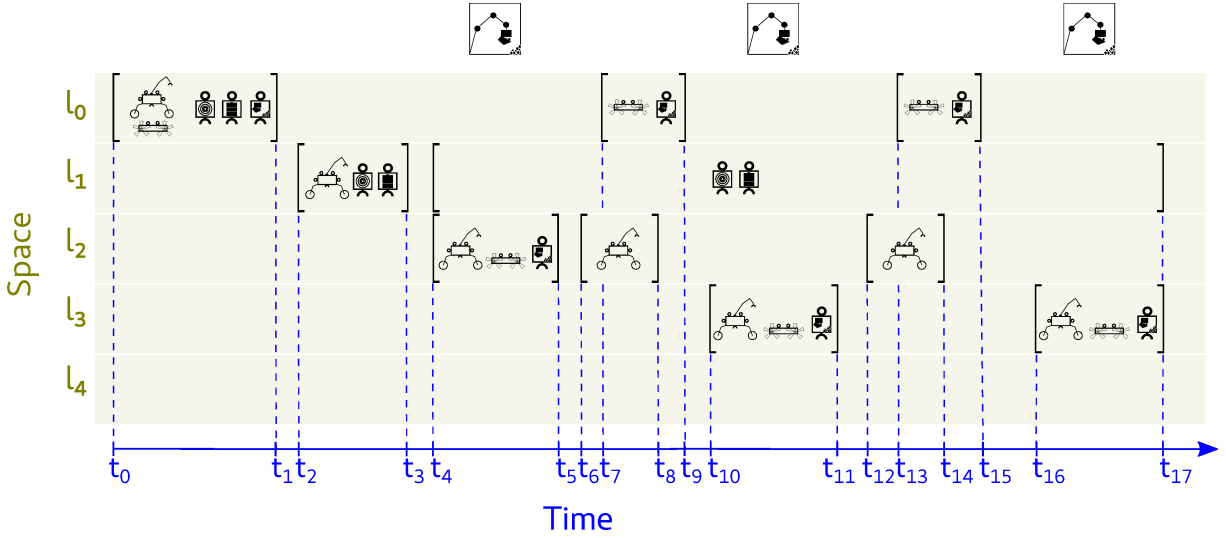


Figure 4.4: Timeline-based illustration of an example mission. Agents are assigned in space and time, while the need for functionality is indicated by pictograms above the top timeline.

An Example Mission Figure 4.4 illustrates the use of a multi-robot system for a sample-return mission: available for the mission are a rover, a shuttle, a soil-sampling module, a power module and a seismograph module. The goal of the mission is to deploy the seismograph module at location l_1 and perform soil sampling. Since only the stationary lander is equipped with a laboratory for automated soil analysis, soil sample have to be returned to the lander. Figure 4.4 illustrates the following sequence of actions: the rover moves towards the first destination l_2 to perform soil sampling; the rover initially carries a power module and seismograph module. While passing the location l_1 the rover deploys the seismograph, which is combined with a power module; the seismograph itself does not comprise a power unit, thus requires an additional power module for operation. Once the sensor deployment has been completed the rover moves towards the soil sampling destination l_2 . The rover is continuously accompanied by a shuttle robot. The shuttle waits for the soil sampling to complete and carries the filled soil-sampling module back to the lander at l_0 for further analysis. The shuttle subsequently takes the cleaned soil-sampling module to meet again with the rover at l_3 , allow to repeat the soil sampling and subsequent return to the lander. During their operation the rover and shuttle can perform parallel activities, e.g., while the rover deploys the seismograph module, the shuttle can explore the environment or create video footage of the deployment. Similarly, the rover can continue exploration while the shuttle returns the soil sampling module to the lander. The illustration shows one specific solution for performing this mission and does not use higher-level abstraction.

Features of Reconfigurable Multi-Robot Missions The example mission demonstrates several aspects of multi-robot operations in general and of reconfigurable multi-robot missions specifically, namely: (1) parallel execution of tasks (2) synchronised execution of tasks, and (3) superadditive effects. The possibility for parallel execution of tasks is an advantage and reason for better efficiency of multi-robot systems compared to a single robot system. The example mission implicitly assigns a servicing role to the shuttle robot to establish a continuous stream of soil-sampling modules. This allows a better exploitation of the specialisation of systems through cooperation. Although cooperation requires the ability to synchronise the activities of robots, the availability of multiple system also opens the opportunity to perform synchronised activities in the first place; a basis for new scientific experiments and observations. Tool exchange can lead to a superadditive effect, and using the soil-sampling module is only one example for space mission. The flexibility of a reconfigurable multi-robot system opens new opportunities for the design of missions. But the increased flexibility and number of feasible agent types demands new management approaches. For space mission, the desire and need for fine-grained control is expected to remain high, although the operational personnel is limited. Hence, a micro-managed multi-robot missions as well as a fully autonomous missions should be considered. Micro-managed missions let operators define resource usage precisely, e.g., an operator might wish to control precisely which robotic system performs a set of tasks. In contrast, a fully autonomous mission definition is only based on functional requirements. The resource assignment is left to the solution procedure.

A natural fit to handle a range of simple to complex requirements is a constraint-based problem representation where a desired mission state with respect to time can be described. The mission specification models the following requirements:

- temporal constraints to design synchronisation,
- resource assignment constraints,
- intermediate and final goals as temporally qualified functional requirements,
- intermediate and final goals as temporally qualified robot assignments, and
- functional property constraints for precise functional requirements.

Section 4.5 further discusses how requirements can be used to model some of the intra-route and inter-route constraints which are found in classical VRPs.

Relation to Existing VRP-based Approaches The presented characteristics for multi-robot missions show a significant overlap to the constraints presented for VRPs, particularly for the VRPTT and VRPMSs. Coordinated operation comes with task synchronisation and the need for linking vehicles for joint operation is equivalent to the general consideration of superadditive effects. Still, a significant distinction remains since the consideration of reconfigurable systems is a generalised approach compared to VRPTT's scenario with trucks and trailers. Embedding the organisation model into the planning approach aims at better generalisation of reconfiguration and handling of complex constraints in general. Thereby, the organisation model serves as basis for planning for a broader range of planning problems.

A further distinction exists with the optimisation criteria. While classical approaches focus on efficiency for cost optimisation, this thesis suggests a multi-objective optimisation function and the combined consideration of efficiency, efficacy and safety (see Section 4.2.2).

One of the benefits of a reconfigurable multi-robot system is the possibility for resource exchange and therefore for failure handling. Systems can fail during operation and although

this might not be critical to the global mission success it can lead to substantial delay and cost penalty. Redundant equipment, which can be used as hot- or cold-swap replacement, can serve as backup and basis for continuing a mission in cases of failure. Therefore a higher redundancy is desirable from a mission safety point of view. As shown in Chapter 3.5 the level of redundancy can be a local agent property or a global organisation property. Overall, redundancy would be typically avoided since it comes with higher cost due to enforced idle time of components by design, and hence a cost trade-off has to be made. In contrast to a monolithic system, a reconfigurable system can adapt redundancy dynamically during a mission. Thus, an application of reconfigurable multi-robot systems could potentially offer a better cost efficient application despite the introduction of redundancy.

4.2.1 Mission Specification & Representation

The overall mission planning problem which arises from the use of reconfigurable multi-robot systems has been cast into a constraint-based mission planning approach. This approach builds upon the definitions for reconfigurable multi-robot systems developed in Chapter 2.4 and the organisation model as outlined in Chapter 3. The definition of a mission combines the domain and problem representation in a temporal database which allows to define the evolution of a mission using a set of constraints. This means, that the initial, intermediate and final state of a mission can be encoded to represent the planning problem. The same representation can be used to describe partial and full solutions.

A human operator has the opportunity to detail a mission for individual atomic agents by defining constraints, e.g., such as on partial paths. Alternatively, a mission can be defined only through functionality requirements; requirements are defined as a set of functionalities which are expanded to a set of suitable agents. The mission specification only accounts for agent roles, i.e., anonymous instances of agent types. Therefore, after a solution to a mission planning problem has been computed, all involved agent roles have to be mapped to the set of real atomic agents. A mission specification is based on so-called spatio-temporal requirements which reflect the use of embodied agents in combination with time:

Definition 4.1 (Spatio-temporal requirement). A spatio-temporal requirement is represented as a spatio-temporally qualified expression (stqe) s , which describes the functional requirements and agent instance requirements for a time-interval and a location:

$$s = (\mathcal{F}, \widehat{GA}_r) @ (l, [t_s, t_e]),$$

where \mathcal{F} is a set of functionality constants, \widehat{GA}_r is the general agent type representing the required agent type cardinalities, $l \in L$ is a location variable, and $t_s, t_e \in T$ are temporal variables describing a temporal interval with the implicit constraint $t_s < t_e$.

Each spatio-temporal requirement represents a persistence constraint, i.e., the requirements have to hold throughout the time interval. The mission specification allows to relate spatio-temporal requirements to the organisation model which defines agent types and functionalities along with the associated properties (cf. Chapter 3.5).

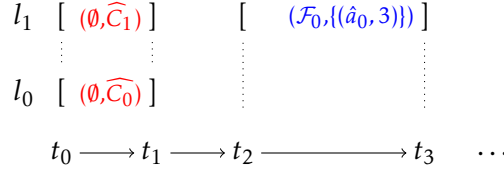


Figure 4.5: A mission specification example consisting of three spatio-temporal requirements: $(\emptyset, \widehat{C}_0)@(\mathbf{l}_0, [t_0, t_1])$ and $(\emptyset, \widehat{C}_1)@(\mathbf{l}_1, [t_0, t_1])$ representing the initial state and $(\mathcal{F}_0, \{(\hat{a}_0, 3)\})@(\mathbf{l}_1, [t_2, t_3])$, where $\mathbf{l}_0, \mathbf{l}_1$ are location variables and t_0, \dots, t_3 are timepoint variables

Definition 4.2 (Mission). A robotic mission is a tuple $\mathcal{M} = \langle \widehat{GA}, STR, \mathcal{X}, \mathcal{OM}, T, L \rangle$, where the general agent type \widehat{GA} describes the available agent types, STR is a set of spatio-temporally qualified expressions, \mathcal{X} is a set of constraints, \mathcal{OM} represents the organisation model, T is the set of timepoints and L the set of locations.

Constraints in \mathcal{X} can refer to stqes as described in Section 4.2.1. The initial state of a mission is defined by the earliest timepoint and binds available agents to their starting depot. The earliest timepoint is $t_0 \in T$ and $\forall t \in T, t \neq t_0 : t > t_0$. Note that this neither requires a single starting location for all agents nor a single final destination, e.g., in contrast to VRP 1-M-1 variants using a single depot.

Graph-based Representation Graphs serve as an intuitive representation for many VRP related problems (cf. (Drex1 2013) and (Ahuja, Magnanti, and Orlin 1993)). A graph representation of the mission specification is also basis for the planning approach. Each vertex represents a space-time requirement, and each edge a set of agent roles. Figure 4.5 illustrates a mission specification, where the starting state is defined by two composite agents \widehat{C}_0 and \widehat{C}_1 which are assigned to the start depots: \mathbf{l}_0 and \mathbf{l}_1 . A single goal requirement demands a functionality set \mathcal{F} , and at least three instances of an agent type \hat{a}_0 at location \mathbf{l}_1 and $t_0 < t_1 < t_2 < t_3$. Figure 4.6 shows an exemplary solution for this mission, where \mathcal{F}_0 defines a transport provider functionality, i.e., a mobile agent type with transport capacity, and three instances of an agent type ‘payload’ are required at location \mathbf{l}_1 within the timeframe of t_2 to t_3 .

Mission Constraints

A mission can be detailed by providing constraints in the constraint set \mathcal{X} . The only initially required constraints are temporal ones to describe the starting timepoint and state. Other constraints are optional, but can be added to the mission specification as part of the planning process to detail the mission and resolve flaws in infeasible solutions. Human operators can use the additional constraints to further restrict solutions characteristics and manually explore new solutions.

Reasoning with agent types instead of agent instances is a first means to reduce the combinatorial explosion, $O(|A|) \leq O(|\theta(A)|)$. Still, to fully control a mission, agent roles, i.e., instances,

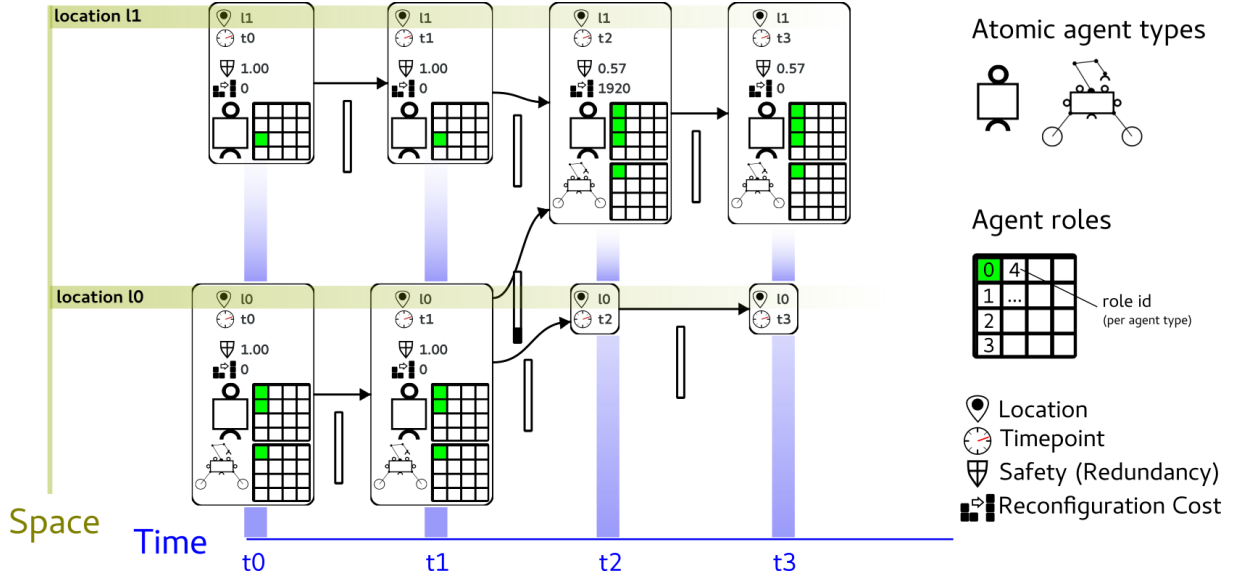


Figure 4.6: A mission solution representation: each vertex in the network represents a space-time tuple describing the agent type assignments. Fillbars on the capacitated edges indicate the total consumed capacity of the transitioning agent. Colour-coded boxes represent unique agent roles: green for fulfilled requirements, grey for an agent role presence without requirement.

have to be considered for timeline construction. Only this timeline construction allows to control the path of individual agents, e.g., such that the same agent performs a given set of tasks. This requirement exists in VRPs as a client delivery preference, for instance when a driver should (re)visit a regular set of clients. For robotic space exploration a sample return mission serves as application example. After sampling a payload cannot be substituted by any other (unused or used) sampling module. Hence, the action execution effectively changes the type of the used sample payload.

Mission constraints augment the set of spatio-temporal requirements and they can be divided into four different categories: (i) temporal, (ii) model, (iii) functionality, and (iv) property. The following paragraphs describe and motivate these constraints.

Temporal Constraints Temporal planning allows to synchronise activities and also allows to perform activities in parallel. While time can be represented qualitatively and quantitatively, the presented mission planning approach uses a combination of a qualitative and quantitative representation of time. Spatio-temporal requirements only require qualitative timepoints to define synchronisation constraints. However, a quantification of time is basis for computing the cost of a mission; the transition of agents between two locations is lower bound by the minimum travel time. A quantification of the temporal network is part of the solution process, where qualitative as well and quantitative constraints are considered. Table 4.4 lists the point-algebra and duration constraints which apply to the qualitative and quantitative time representations respectively.

Model Constraints Model constraints set requirements for agent types and agent roles. They allow bounding the cardinality of agent types so that the combinatorial search problem can be

Table 4.4: Temporal constraints for a mission $\mathcal{M} = \langle \widehat{A}, STR, \mathcal{X}, \mathcal{OM}, T, L \rangle$.

Name	Syntax	Description
temporal relation	$\langle t_n, REL, t_m \rangle$	t_n and t_m are qualitative timepoints and REL is the set of permitted relations, so that $REL \subseteq \{<, >, =\}$ (Dechter 2003)
min duration	$minDuration(t_n, t_m, d)$	sets a lower bound for the duration of a time interval: $t_n - t_m \geq d$, where t_n and t_m are two qualitative timepoints $d \in \mathbb{R}^+$; implies the qualitative relationship $t_n > t_m$
max duration	$maxDuration(t_n, t_m, d)$	sets an upper bound for the duration of a time interval: $t_n - t_m \leq d$, where t_n and t_m are two qualitative timepoints $d \in \mathbb{R}^+$; implies the qualitative relationship $t_n > t_m$

limited according to a least-commitment principle. Equality constraints allow to restrict agent routes partially or even completely. Requiring the minimum equality of a single agent type over the full mission defines the full route for a single atomic agent of this type. Modelling constraints allow to detail a mission to a high degree. In general, constraints have to be considered which apply to the dimensions space, time, agent types, and roles. TemPl implements only a subset of feasible (meta-)constraints and Table 4.5 lists the available model constraints.

Table 4.5: Model constraints, where $S \subseteq STR$ and $A_s^{\hat{a}}$ represents the general agent type requirement of $s \in S$.

Name	Syntax	Description
min cardinality	$minCard(S, \hat{a}, c)$	Minimum cardinality constraint $\forall s \in S : \gamma_{\widehat{GA}_s}(\hat{a}) \geq c$, where $c \geq 0$
max cardinality	$maxCard(S, \hat{a}, c)$	maximum cardinality constraint corresponding to $minCard$ so that $\forall s \in S : \gamma_{\widehat{A}_s} \leq c$, where $c \geq 0$
all distinct	$allDistinct(S, \hat{a})$	$\forall s \in S : \bigcap A_s^{\hat{a}} = \emptyset$
min distinct	$minDistinct(S, \hat{a}, n)$	$\forall s_i, s_j \in S, i \neq j : \left A_{s_i}^{\hat{a}} - A_{s_j}^{\hat{a}} \right \geq n$, where $n > 0$
max distinct	$maxDistinct(S, \hat{a}, n)$	the equivalent maximum constraint to $minDistinct$, so that $\forall s_i, s_j \in S, i \neq j : \left A_{s_i}^{\hat{a}} - A_{s_j}^{\hat{a}} \right \leq n$, where $n \geq 0$
min equal	$minEqual(S, A_r)$	minimum existence of the same agent roles so that $A_{eq} = \bigcap_{s \in S} r(A_s)$ and $A_r \subset A_{eq}$, where $A_r \subset r(A)$, A is the available agent pool for a mission, and A_s is the agent pool that fulfils $s \in S$
max equal	$maxEqual(S, A_r)$	maximum existence of the same agent roles so that $A_{eq} = \bigcap_{s \in S} r(A_s)$ and $A_{eq} \subset A_r$, where $A_r \subset r(A)$, A is the available agent pool for a mission, and A_s is the agent pool that fulfils $s \in S$
all equal	$allEqual(S, A_r)$	the constraint conjunction: $minEqual(S, A_r) \wedge maxEqual(S, A_r)$

Functionality & Property Constraints Agents either comprise a functionality or they do not. In effect, functionality is requested with a maximum cardinality of one, which makes the introduction of a maximum function constraint unnecessary. Note that this is a limitation of the current modelling approach, which is discussed in Chapter 4.5. However, the property of a agent providing a particular functionality can be of importance, and the use of property constraints allows to narrow applicable agents. The so-called `TransportProvider` functionality servers as an examples. This functionality characterises all mobile agents, which offer a trans-

Table 4.6: *Functionality and property constraints.*

Name	Syntax	Description
min function	$minFunc(s, f)$	functionality f to be available at spatio-temporal requirement (str) $s \in STR$, so that $f \in \mathcal{F}^s$, where \mathcal{F} represents the functionality requirements associated with s
min property	$minProp(s, f, p, n)$	constrain the numeric property p_f of a functionality f to be $p_f \geq n$, where $n \in \mathbb{R}$ and $minProp(s, f, p, n)$ implies that $minFunc(s, f)$ holds
max property	$maxProp(s, f, p, n)$	equivalent maximum property value constraint to $minProp(s, f, p, n)$, so that property $p_f \leq n$, $n \in \mathbb{R}$

port capacity. A particular demand can only be handled by systems with enough transport capacity, and a property constraint allows to filter these out, here by requiring a minimum transport capacity. Table 4.6 lists the constraints for functionalities and properties.

4.2.2 Mission Solution & Cost Function

Evaluation and optimisation of solutions to the mission planning problem can only be done with an appropriate optimisation or cost function. Cost in a VRP are most often only measured as the total travelled distance, The objectives of TemPl’s mission planning approach are broader, so that additional cost factors are considered. The cost function therefore intends to support three optimisation objectives: (i) maximisation of efficacy, (ii) maximisation of efficiency, and (iii) maximisation of safety. Any feasible solution and thus any feasible assignment of atomic agents which fulfils all demanded requirements leads to maximum efficacy. A penalty can therefore be considered for all (partially) infeasible solutions. Maximisation of efficiency in TemPl corresponds to a minimisation of the travelled distance in VRPs. But the total travelled distance is only a suitable cost measure for homogeneous robotic teams. In heterogeneous robotic teams individual robots come with distinct energy consumption, which the total travelled distance cannot reflect. Even more so, when robots are combined and energy can be shared among atomic agents in a composite agent. Additionally, reconfiguration cost have to be accounted for. Firstly, changing the overall coalition structure requires time (and energy). Secondly, a change of the coalition structure also poses a risk: a desired configuration might (temporarily) not be reachable for reasons the current model cannot account for or atomic agents might be completely lost during such operation. Hence, reconfiguration is not only reducing efficiency, but also safety. To estimate the cost of changing the state of an agent organisation two related concepts are used: policies and heuristics (as described in Section 3.5).

A solution \mathcal{M}_s to the mission specification \mathcal{M} is a conflict free assignment of agents to spatio-temporal requirements. A qualitative view on time is sufficient to compute this assignment. The relation between all timepoints must, however, be defined with no open aka universal temporal constraints remaining. Time needs to be quantified before the solution cost can be computed. Therefore the qualified temporal network is converted into a quantitative temporal network: time intervals can be constrained based on the estimated travel time of mobile systems between locations leading to a STN (Dechter 2003) from which the transition times can be extracted.

A cost function that balances the three optimisation targets is based on the heuristic cost computations which are listed in Table 4.7. The cost function itself is defined as:

Table 4.7: *Heuristics for cost computation on the solution \mathcal{M}_s for a mission \mathcal{M} .*

Name	Syntax	Description
distance	$d(a, \mathcal{M}_s)$	travelled distance of an agent a in \mathcal{M}_s
operation time	$op(a, \mathcal{M}_s)$	corresponds to the time horizon of the mission; any location change introduces a lower bound for time intervals by assuming a traversal with the mobile agent's nominal velocity $v_{nom}(a)$
energy	$E(a, \mathcal{M}_s)$	$E(a, \mathcal{M}_s) = op(a, \mathcal{M}_s) \cdot pw(\hat{a})$ overall consumed energy by agent a to perform \mathcal{M}_s ; $E(\mathcal{M}_s) = \sum_{a \in A}$ overall consumed energy per mission
safety	$SAF(\mathcal{M}_s)$	$SAF(\mathcal{M}_s) = \min_{s \in STR} saf(s)$ represents the minimal safety level (here: redundancy) of the mission, where $saf(s)$ defines the safety metric associated with an spatio-temporal requirement s based on the available (general) agent and with respect to the required set of resources; currently a redundancy based model is used to estimate the probability of survival based on an agent's set of component required to provide the functionalities in \mathcal{F} , such that $0 \leq saf(s) \leq 1$.
fulfilment	$SAT(\mathcal{M}_s)$	$SAT(\mathcal{M}_s) = \frac{1}{ STR } \sum_{s \in STR} sat(s)$ represents the ratio of fulfilled requirements, where

$$sat(s) = \begin{cases} 0 & , \text{ unfulfilled} \\ 1 & , \text{ fulfilled} \end{cases}$$

$$cost(\mathcal{M}_s) = \alpha E(\mathcal{M}_s) + \beta SAT(\mathcal{M}_s) + \epsilon SAF(\mathcal{M}_s) \quad (4.1)$$

It reflects the balancing of the three general mission aspects: efficiency through the energy cost function, efficacy through checking the level of fulfilment, and safety as a redundancy dependant survival metric; for balancing the parameters α , β and ϵ can be used. Note that safety and fulfilment have a value range $[0, 1]$, so that α should account for normalisation to $[0, 1]$ for the energy cost; α therefore comprises a factor E_{max}^{-1} , where E_{max} is the maximum energy cost which is either allowed or observed in any existing mission solution.

4.3 Spatio-temporal Planning

The mission specification which has been described in the previous sections uses a constraint-based representation, and the following section details the correspondingly constraint-based planning approach of TemPl. TemPl's algorithm is based on the general flow as depicted in Figure 4.7.

The algorithm uses a CSP-based approach to generate plan candidates, which can also represent only partial solutions, which are detailed as part of the candidate optimisation step. Candidate optimisation can also identify infeasible solutions, which can be optionally repaired. Candidate can be continuously generated until one or multiple abort conditions apply: a minimum number of candidates has been generated, a minimum number of solutions with an

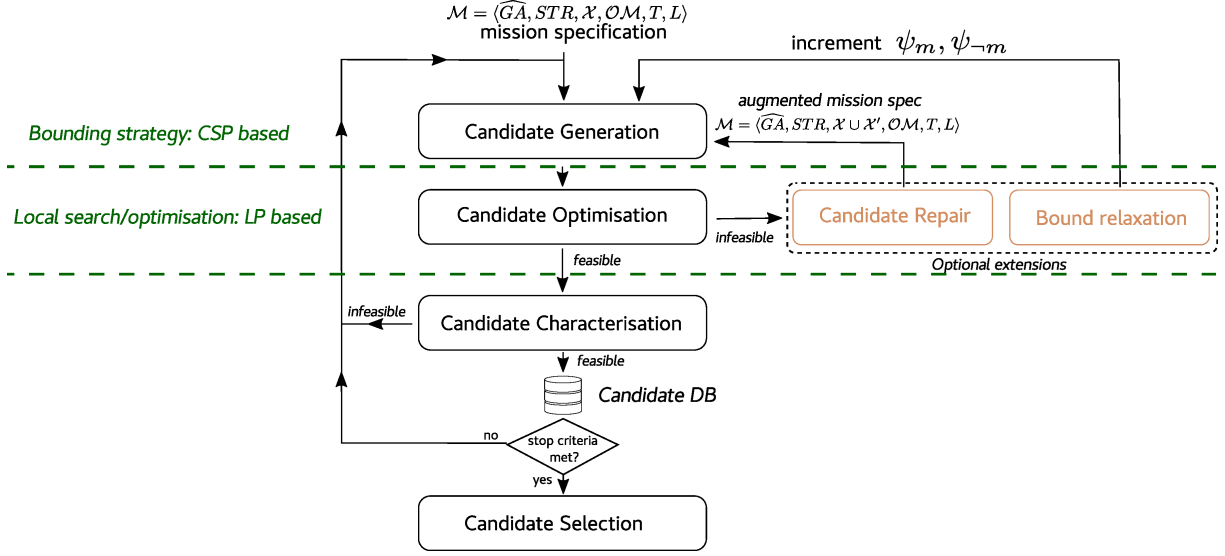


Figure 4.7: Schematic view illustrating the high-level components of TemPl's algorithm to generate and process solution candidates.

efficacy of 1.0 has been generated, or the search space has been exhaustively explored. Due to this layout the algorithm can be seen as anytime planning approach. The algorithm can be decomposed into the following stages:

Candidate Generation

- (1) generation of a time-expanded network and temporal constraint network (without gaps) for the synchronisation for agent activities
- (2) bounding of agent type cardinalities,
- (3) assignment of agent roles,
- (4) generation of agent role timelines

Candidate Optimisation

- (5) local optimisation,
- (6) solution validation including the identification of flaws

Candidate Characterisation

- (7) quantification of reconfiguration time
- (8) quantification of time
- (9) computation of efficacy, efficiency and safety

Candidate generation uses an iterative refinement of a mission until a full assignment of atomic agents to spatio-temporal requirement exists. The stages 1 to 5 involve backtracking and a stage either produces a valid variable value assignment or backtracks to a previous stage. Candidate repair triggers a restart of the search with an augmented mission; candidate repair is an optional strategy. The following sections describe each stage in detail.

4.3.1 Stage 1: Synchronisation of Agent Activities

Time-expanded networks (Fleischer and Skutella 2007; Ford and Fulkerson 1963) $G = (V, E)$ can model flow over time and are used in this thesis to synchronise time-based agent activities.

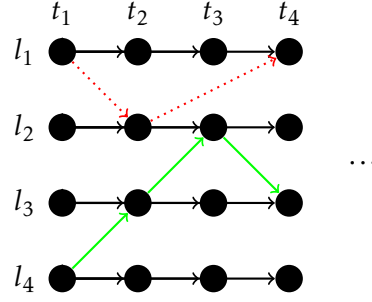


Figure 4.8: Example for a time expanded network (based on Figure 6 in (Roehr and F. Kirchner 2016)): a legal path (solid green line) and an illegal path (dotted red line).

The basis for the time-expansion is a set of temporally ordered qualitative timepoints. A time-expanded network according to Definition 4.3 is used.

Definition 4.3 (Time-expanded network). A time-expanded network for a set of timepoints $T = \{t_1, \dots, t_{|T|}\}$ and a set of locations is a graph $G = (V, E)$ with the following properties: each vertex $v_{l,t} \in V, l \in L, t \in T$ corresponds to a location timepoint tuple: $\forall l \in L, t \in T : \exists v_{l,t} \in V$. The set of edges is restricted: $e \in E \implies e = (v_{t_n, l_i}, v_{t_{n+1}, l_j}), n = 1, \dots, |T|, i, j = 1, \dots, |L|$. Without loss of generality $t_1 \leq t_2 \leq \dots \leq t_{|T|}$. A vertex $v_{t_1, l} \in V, l \in L$ is denoted by *setup vertex*, and an edge $e \in E$, where $e = (v_{l, t_n}, v_{l, t_{n+1}}) \in E, n = 1, \dots, |T|, l \in L$ is denoted by *local transition edge*.

Edges in the time-expanded network only connect vertices that are associated with the next timepoint. This restriction reflects that time progresses towards the final time horizon and cannot move sideways. This restriction reduces the amount of edges that need to be considered in planning; the number of edges can be reduced approximately by a factor $\frac{|T|}{2}$ compared to an approach where each vertex can be connected to any vertex associated to a later timepoint (cf. Appendix G.4).

Definition 4.4 (Space-time point). The tuple $stp = (t, l)$ where t is a timepoint and l is a location is denoted by space-time-point.

As part of the planning process, resource requirements, or rather agent requirements are related to all space-time-points that can be constructed from the all relevant locations and timepoints. Note that for any implementation the relevance of locations and timepoints can be tested upon to avoid an unnecessary increase in computational complexity. The current algorithm assumes a temporally ordered set of timepoints without time gaps, i.e., where all relations between timepoints are temporally constrained and no universal constraint (Dechter 2003) remains. A set of timepoints without time gaps is a requirement to perform immobile agent transport optimisation or rather multi-commodity min-cost flow optimisation on the time-expanded network.

Temporal Constraint Network

TemPl uses qualitative temporal reasoning which allows to define partially ordered sequences of activities. Point algebra is used with the set of relations $REL = \{>, <, =\}$. The generation of a fully constrained set of timepoints is based on a TCN (Rina Detcher 2003). The corresponding CSP is defined by a set T to represent the timepoint variables, a set D to represent the domain values for each timepoint $t \in T$, and a constraint set C with constraints of the form $c = \langle t_n, rel_i, t_m \rangle$, where $n, m = 1, \dots, |T|$, and $rel_i \in REL$. A constraint is fulfilled if the relation described by $c \in C$ between two timepoint variables is fulfilled.

$$T = \{t_1, t_2, \dots, t_{|T|}\} \quad (4.2)$$

$$D = \{D_1, D_2, \dots, D_{|T|}\} \quad (4.3)$$

$$(4.4)$$

The final domain for each variable is restricted to a singleton: $|D_i| = 1$ and permitted values are $D_i \subset \{1, 2, \dots, |T|\}$.

4.3.2 Stage 2: Bounding Agent Type Cardinality

Each spatio-temporal requirement defines agent type and functionality requirements, whereas the planner translates both to role requirements. A solution contains agent role assignments only. In order to initially enforce only a minimum role assignment, lower and upper bounds for the cardinalities of agent type instances are set. The agent type cardinality defines how many instances of an agent type can be created and thus bounds the required number of agent roles to solve the mission. To apply the bounds the minimum agent type constraints are identified for each spatio-temporal requirement based on: (a) the overall resource availability, (b) the resource saturation bound with respect to all demanded functionalities, (c) and the lower bound given through specifically required agent instances. The overall resource availability sets an upper bound not only to a single , but to all mutually exclusive .

Definition 4.5 (Mutual exclusive spatio-temporal requirements). Two spatio-temporal requirements s_0 and s_1 are denoted *mutual exclusive* if and only if they refer to different locations and their associated time intervals overlap.

The $n \times m$ matrix AT as depicted in Figure 4.9 represents a set of CSP variables $x_{i,j}$ where $n = |\theta(A)|$ and $m = |STR|$. A variable $x_{i,j}$ describes the agent type cardinality for a spatio-temporal requirement s_i and an agent type \hat{a}_j . A variable's value domain is based on the size of respective type partition of the available atomic agent set: $D_{x_{i,j}} = \{0, \dots, |A^j|\}$:

After identification of all mutually exclusive spatio-temporal requirements agent type upper and lower bounds can be enforced, according to Equations (4.5) and (4.6):

$$\forall \xi \forall \hat{a}_j \in \widehat{A} : \sum_{s_i \in \theta} x_{i,j} \leq \gamma_{\widehat{GA}}(\hat{a}_j) \quad (4.5)$$

$$\forall \xi \forall \hat{a}_j \in \widehat{A} : \sum_{s_i \in \theta} x_{i,j} \geq \gamma_{\widehat{GA}}(\hat{a}_j) \quad (4.6)$$

$$\mathbf{AT} = \begin{matrix} & \hat{a}_1 & \hat{a}_2 & \cdots & \hat{a}_n \\ \begin{pmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,n} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m,1} & x_{m,2} & \cdots & x_{m,n} \end{pmatrix} & s_1 \\ & s_2 \\ & \vdots \\ & s_m \end{matrix}$$

Figure 4.9: Matrix-based representation of agent type cardinality constraints. Columns and rows are annotated with the corresponding meaning of the variable indices.

, where

- Ξ Set of all sets of mutually exclusive spatio-temporal requirements of a mission \mathcal{M}
- ξ Set of mutually exclusive spatio-temporal requirements, such that $\xi \in \Xi$
- \widehat{GA} General agent representing the agent pool A , and defines the maximum cardinality for each agent type in a given mission \mathcal{M}

Extensional Constraints The combination of functionality constraints and agent requirements is based on a maximum of two agent types.

Definition 4.6 (Maximum of two agent types). The maximum $\max(\widehat{X}, \widehat{Y})$ of two agent types \widehat{X} and \widehat{Y} is defined as agent type \widehat{Z} , where $\forall \hat{a} \in \theta(X \cup Y) : \gamma_{\widehat{Z}}(\hat{a}) = \max(\gamma_{\widehat{X}}(\hat{a}), \gamma_{\widehat{Y}}(\hat{a}))$

For each spatio-temporal requirement $s = (\mathcal{F}, \widehat{GA}) @ (l, [t_s, t_e])$ a corresponding set of minimal agent types θ_s exists, which represents the maximum of a combination of $\theta^*(\mathcal{F}, A)$ and \widehat{GA} , where A is the available set of agents:

$$\theta_s = \theta^*(\mathcal{F}, A) \times \{\widehat{GA}\} \setminus \{\max(\widehat{A}_f, \widehat{GA}) \mid \widehat{A}_f \in \theta^*(\mathcal{F}, A)\}$$

The set of agent types θ_s represents the domain of a spatio-temporal requirement for the CSP. All agent types in this domain are minimal, so that this set cannot be reduced to a common or at least more compact agent type combination. Extensional constraints are therefore used to account for the set of allowed agent types in the constraint-based planning process.

Extensional constraints, also known as table constraints, explicitly define possible value assignment to tuples of variables; the Global Constraint Catalog (Beldiceanu and Demassey 2018)) states this more formally as:

“Enforce the tuple of variable VARIABLES to take its value out of a set of tuples of values TUPLES_OF_VALS. The *value* of a tuple of variables $\langle V_1, V_2, \dots, V_n \rangle$ is a tuple of values $\langle U_1, U_2, \dots, U_n \rangle$ if and only if $V_1 = U_1 \vee V_2 = U_2 \vee \dots \vee V_n = U_n$ ”

The concept of the functional saturation, which has been introduced in Section 3.3.2, is embedded with extensional constraints and reduces the value domain for all which contain functionality requirements. Since extensional constraints lead to an exact assignment of agent type cardinalities, agent type cardinalities are only interpreted as a minimal bound by subsequent stages.

4.3.3 Stage 3: Assignment of Agent Roles

In comparison to VRPs an agent can represent a vehicle as well as an item. All atomic agent instances need to be identifiable in a computed solution to allow for the application of equality or distinction constraints. Therefore agent roles are part of TemPl, and for each role a timeline is computed. The assignment of agent roles to spatio-temporal requirements is represented by the $n \times m$ -Matrix \mathbf{AR} , where $n = |STR|$ and $m = |r(A)|$ (see Figure 4.10). Each item $\mathbf{AR}_{s_i, r_l^{\hat{a}_k}} = y_{i,k,l}$ in this matrix represents a CSP variable, with a domain $D_{y_{i,k,l}} = \{0, 1\}$, which represents the involvement of an agent role in the fulfilment of a spatio-temporal requirements: 0 for no involvement, 1 for agent role is present and contributing to the fulfilment of the requirement.

$$\mathbf{AR} = \begin{matrix} & \begin{matrix} r_1^{\hat{a}_1} & r_2^{\hat{a}_1} & \cdots & r_1^{\hat{a}_k} & \cdots & r_l^{\hat{a}_n} \end{matrix} \\ \begin{pmatrix} y_{1,1,1} & y_{1,1,2} & \cdots & y_{1,k,1} & \cdots & y_{1,n,l} \\ y_{2,1,1} & y_{2,1,2} & \cdots & y_{2,k,1} & \cdots & y_{2,n,l} \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ y_{m,1,1} & y_{m,1,2} & \cdots & y_{m,k,1} & \cdots & y_{m,n,l} \end{pmatrix} & \begin{matrix} s_1 \\ s_2 \\ \vdots \\ s_m \end{matrix} \end{matrix}$$

Figure 4.10: Internal CSP model to deal with agent role constraints. Columns relating to the same agent type are interchangeable. Constraints are applied: (a) to items with the same column index to account for unary resource constraints and mutually exclusive spatio-temporal requirements (red), (b) to items with the same row index, and same agent type to set a lower agent type cardinality bound to fulfil each requirement (blue), and (c) to set an upper agent type cardinality bound for mutually exclusive spatio-temporal requirements (green).

Unary Atomic Agent Usage Atomic agents are embodied and therefore cannot be assigned to any two spatio-temporal requirements which are mutually exclusive. As mentioned in Section 4.3.2, all mutually exclusive spatio-temporal requirements can be identified once the temporal constraint network has been fully assigned. Since all mutually exclusive spatio-temporal requirements require a likewise mutually exclusive resource usage, a unary agent role usage is enforced for all sets of mutually exclusive spatio-temporal requirements:

$$\forall \xi \forall r_l^{\hat{a}_k} \in r(A) : \sum_{s_i \in \xi} y_{i,k,l} \leq 1 \quad (4.7)$$

, where

Ξ	Set of all sets of mutually exclusive spatio-temporal requirements of a mission \mathcal{M}
ξ	Set of mutually exclusive spatio-temporal requirements, such that $\xi \in \Xi$
$r(A)$	Set of all agent roles corresponding to the agent pool A

Relaxation of the Agent Type Cardinality Bound The number of agent roles of a particular agent type is limited by the prior computed agent type bounds (see Section 4.3.2). This bound can be too tight to find a feasible solution. Hence, additional sub-constraints permit a relaxation of this bound:

$$\forall s_i \forall \hat{a}_k : x_{i,k} \leq \sum_{l=0}^{|A^k|-1} y_{i,k,l} \leq b_{i,k} \quad (4.8)$$

where

$$0 \leq \sum_{\hat{a}_k \in \widehat{A}} \psi_{\hat{a}_k} \leq \psi_m, \text{ if } mobile(\hat{a}_k) \quad (4.9)$$

$$0 \leq \sum_{\hat{a}_k \in \widehat{A}} \psi_{\hat{a}_k} \leq \psi_{-m}, \text{ if } \neg mobile(\hat{a}_k) \quad (4.10)$$

, where

s_i	spatio-temporal requirement $s_i \in STR$ for a mission \mathcal{M}
\hat{a}_k	available agent type $\hat{a}_k \in \widehat{A}$
$b_{i,k}$	$\begin{cases} A^k & \text{if } x_{i,k} + \psi_{\hat{a}_k} \geq A^k \\ x_{i,k} + \psi_{\hat{a}_k} & \text{otherwise} \end{cases}$
$\psi_{\hat{a}_k}$	deviation of the agent type cardinality bound from the minimum lower bound \hat{a}_k , where $\psi_{\hat{a}_k} \in \mathbb{N}^0$
ψ_m	allowed bound deviation for the agent type cardinality of all mobile agents, where $\psi_m \in \mathbb{N}^0$
ψ_{-m}	allowed bound deviation for the agent type cardinality of all immobile agent types, where $\psi_{-m} \in \mathbb{N}^0$

The parameters ψ_m and ψ_{-m} control the use of additional mobile or immobile roles with respect to the minimum required roles. The minimum required number for each role is defined through the mutual exclusive requirements, since each role is a unary resource, A setting of $\psi_m = 0$ and ψ_{-m} is equal to enforcing the minimal bound. The total number of used atomic agents is still limited to the number of available atomic agents, as part of the definition of the local bound $b_{i,k}$.

Active Agent Roles A full assignment can be computed as soon as unary resource constraints and agent type cardinality bound constraints have been propagated. Not all available agent roles are required for a solution. Unused agent roles remain part of the spatio-temporal requirement (str) representing the initial state. Any intermediate problem can therefore be reduced to a set of so-called active agent roles:

Definition 4.7 (Active agent role). Any agent role $r_l^{\hat{a}_k}$ that fulfils the following condition is denoted by **active agent role**:

$$\sum_{s_i \in STR \setminus S} y_{i,k,l} > 1 \quad , \quad (4.11)$$

where S denotes all spatio-temporal requirement (str) which encode the initial state.

The identification of active agent roles allows to prune the set of agent roles for subsequent stages.

Solution Redundancy Any atomic agent which exists at the corresponding starting depot of an agent role is a suitable candidate to execute the respective timeline. Therefore, column assignments in the agent role assignment matrix (see Figure 4.10) are interchangeable as long as they belong to the same agent type. Similarly to the three-index modelling (VRP3) for VRPs (see Section 4.1.1) interchangeable columns result in redundant solutions. To reduce the number of redundant solutions, symmetry breaking is applied. For the implementation Lightweight Dynamic Symmetry Breaking (LDSB) (Mears, Garcia de la Banda, et al. 2008) is used as part of the CSP framework Gecode. Symmetry breaking significantly reduces the number of redundant solutions, but it does not guarantee completeness (Schulte, Tack, and Lagerkvist 2018, p. 141), so that redundant solution might still be encountered.

4.3.4 Stage 4: Generation of Agent Role Timelines

The assignment of agent roles to spatio-temporal requirements is not only restricted through mutual exclusion and upper and lower bounding of agent type cardinalities. It must also maintain a valid path in the time-expanded network. The timeline representation is based on an adjacency list, and a timeline is encoded by a set of CSP variables P , with $p_{t,l} \in P$ where $t \in T$ and $l \in L$, T and L denote the set of timepoints and locations. Hence, each variable corresponds to a vertex in the time-expanded network. The value domain representation is encoded as a set of sets $D_p = \{\emptyset, \{0\}, \dots, \{n\}\}$. A matrix based interpretation is provided in Figure 4.11:

A path propagator has been implemented to enforce the constraints of the time-expanded network. The path propagator guarantees that all value assignments represent a valid path in the time-expanded network and thereby creates the necessary precondition to perform flow optimisation. Path propagation is only performed for agent roles referring to mobile systems; these define the feasible (transport) paths in the flow network. This leaves timelines of immobile agent roles partially unassigned, but a full routing of immobile agents is the result of the subsequent step of flow optimisation. Local transitions come at not cost for all systems. But for relocation only the routes of the mobile agents can be used.

4.3.5 Stage 5: Local Optimisation

A transport network can be generated when the agent roles corresponding to mobile agent types have been assigned and their timelines have been created. The capacities of the edges in this transport network are defined by the capacity of the mobile agents travelling that edge.

$$\begin{array}{c}
l_1 \\
l_2 \\
l_3 \\
l_4
\end{array}
\begin{array}{c}
t_1 \quad t_2 \quad t_3 \quad t_4 \\
\left(\begin{array}{cccc}
\{\}^0 & \{10\}^4 & \{\}^8 & \{\}^{12} \\
\{\}^1 & \{\}^5 & \{\}^9 & \{\}^{13} \\
\{4\}^2 & \{\}^6 & \{14\}^{10} & \{\}^{14} \\
\{\}^3 & \{\}^7 & \{\}^{11} & \{\}^{15}
\end{array} \right)
\end{array}$$

Figure 4.11: Example of a matrix based interpretation of a timeline corresponding to a time-expanded network which is constructed for four locations and four timepoints. Superscript values (in grey) indicate the CSP variable index, and value assignments (in blue) are the index of the variable corresponding to the next space-time point in the timeline.

It might not be desirable to perform transitions with each atomic agent in a final solution, yet, a worst case performance relies on that option. The assignment of immobile agent roles is translated into a flow optimisation problem using commodity demand and supply; each immobile agent role directly corresponds to a single commodity. Due to this correspondence, the maximum single agent commodity demand and supply is 1. The resulting transshipment problem can be formulated as a multi-commodity min-cost flow problem (MMCF). Based on the formal description of MMCF by Kennington (1978) the following flow-based representation of the optimisation problem is used:

$$\min \sum_{k,m} c_m^k x_m^k \quad (4.12)$$

, where

$G = (V, E)$ Graph G with vertices V and edges E

$K =$ number of commodities

$k = 1 \dots K$

$m = 1 \dots |E|$

$e_m =$ edge $e_m \in E$

$x_m^k =$ flow for commodity k in edge e_m

$c_m^k =$ unit cost for commodity k in edge e_m

subject to

$$\sum_{e_m \in A_n} x_m^k - \sum_{e_m \in B_n} x_m^k = \begin{cases} S_k^+ & \text{if } n = s_k \\ -S_k^- & \text{if } n = t_k \end{cases}, \forall n \in V, k \quad (4.13)$$

$$\sum_{e_m \in A_n} x_m^k - \sum_{e_m \in B_n} x_m^k \geq S_k^n, \forall n \in V : n \neq s_k, n \neq t_k \quad (4.14)$$

$$\sum_{1 \leq k \leq K} x_m^k \leq q_m, \forall m \quad (4.15)$$

$$l_m^k \leq x_m^k \leq u_m^k, \forall m, k \quad (4.16)$$

, where

n	$n = 1 \dots V $
v_n	vertex $v_n \in V$
s_k	start (supply) vertex of commodity k , $s_k \in V$
t_k	target (demand) vertex of commodity k , $t_k \in V$
S_k^+	supply of source vertex s_k
S_k^-	demand of target vertex t_k
S_k^n	minimum trans-flow requirement for commodity k through vertex n , $S_k^n \geq 0$
B_n	set of incoming edges of vertex n
A_n	set of outgoing edges of vertex n
q_m	upper bound for total flow through edge m
u_m^k	upper bound for flow of commodity k through e_m
l_m^k	lower bound for flow of commodity k through e_m

The optimisation target defined by Equation (4.12) is cost minimisation. The unit cost c_m^k can be set per commodity and per edge. This representation leaves maximum control over the cost factor, so that unit cost can be accounted for as well as sophisticated cost measures. The constraint (4.13) balances the outgoing and incoming commodities per vertex at origin and destination node. This enforces a flow from commodities that originate at s_k and are demanded at t_k .

The mission planner solves an assignment problem based on constraint propagation before the MMCF optimisation takes place. After propagation of the constraints, all assigned variables for the immobile systems have to be interpreted as hard requirements, which can only be fulfilled by routing the respective immobile system accordingly. Typically, only the initial supply and final demand are used for flow optimisation in order to define the problem. However, to encode a detailed mission design into a flow network, control of partial routes is required. To control the path of a commodity a transflow constraint (4.14) is added. This inequality defines a lower bound S_k^n for transitioning flow through a vertex n . If no transition flow requirement exists then $S_k^n \geq 0$.

The upper capacity bound for an edge is defined by Constraint (4.15), whereas Constraint (4.16) defines the upper and lower bound for commodity flow across an edge. Bundle constraints can generalise the flow bound setting to combinations of commodities.

$$\bar{l}_m^{\mathcal{K}} \leq \sum_{k \in \mathcal{K}} x_m^k \leq \bar{u}_m^{\mathcal{K}} \quad \forall e_m \in E, \mathcal{K} \in \bar{\mathbf{K}}_m \quad (4.17)$$

$$\bar{l}_n^{\mathcal{K}} \leq \sum_{k \in \mathcal{K}_n} x_m^k \leq \bar{u}_n^{\mathcal{K}} \quad \forall v_n \in V, e_m \in B_n, \mathcal{K} \in \bar{\mathbf{K}}_n \quad (4.18)$$

, where

$\bar{u}_m^\mathcal{K}$	upper bound for flow of commodity k through e_m
$\bar{l}_m^\mathcal{K}$	lower bound for flow of commodity k through e_m
$\bar{\mathbf{K}}_m$	set of sets of commodities with constrained flow through e_m
$\dot{u}_n^\mathcal{K}$	upper bound for the inflow of vertex n with respect to a commodity combination \mathcal{K}
$\dot{l}_n^\mathcal{K}$	lower bound for the inflow of vertex n with respect to a commodity combination \mathcal{K}
$\dot{\mathbf{K}}_n$	set of set of commodities with constraint inflow into vertex v_n

The bundle constraints (4.17) and (4.18) limit the flow for a subset of commodities and can therefore replace constraints (4.15) and (4.16). The bundle constraints give additionally control over commodities of the same type, so that minimum and maximum cardinality constraints for agent types can be maintained throughout the local search. The finally implemented flow modelling approach relies on the usage of a single start depot s_o and a single final depot s_d , so that $s_k = s_o$ and $t_k = s_d$ for all commodities k . To guarantee the initial mission setup with agents at the demanded locations, additional constraints on the outgoing edges of s_o are required to control the starting depots' outgoing flow:

$$u_m^k = S_k^n \quad , \text{ where } \forall v_n \in V : e_m = (s_o, v_n) \quad (4.19)$$

As a result, the depot vertex s_o only has outgoing edges to the initial so-called setup vertices (see Definition 4.3). Only by bounding the commodity flow to the exact demands of the setup vertices, the planning problem is correctly represented. The use of VRP 1-M-1 (Toth and Vigo 2014) based modelling for the flow network definition simplifies the management of the overall mass balancing constraints. As a side-effect, all resources are routed along the full mission timeline, even though only partial requirements might exist for these resources. The formulation increases the size of the linear problem, but considers the reuse of commodities.

4.3.6 Stage 6: Solution Validation

The planning process has to guarantee the feasibility of a solution. The preceding flow optimisation stage returns either a feasible or an infeasible solution to the local optimisation problem. For any infeasible solution, the existing flow violation is extracted from the resulting flow network when possible. Some LP Solver might not support the extraction of intermediate result as a result of using so-called presolvers.

Three types of flaws can be extracted: *min-flow violation* for a missing minimum flow, *trans-flow violation* for a missing transitional flow, and *flow-balance violation* so that a flaw v is a tuple: $v = \langle t, l, r(A'), \Delta, i, o \rangle$, where t is a timepoint, l a location, $r(A')$ the set of all agent roles with a demand at space-time point (l, t) , Δ the difference between commodity demand and actual supply, and i and o refer to the inflow and outflow respectively.

The combination of the constrained-based planning approach and the integration of a local optimisation using linear programming comes with some limitations. The constrained-based planning approach generates a feasible assignment with respect to a space-time point, and guarantees the availability of functionality by assigning suitable agents. Yet, a solution which

is feasible according to the LP Solver, does not have to be a feasible solution for the mission planning problem. Transitions between different locations can still involve infeasible coalition formations. Hence, after local search a transition between two locations can involve a set of agents which can neither form a single feasible agent, nor a feasible set of agents to allow transport of all involved agents. Section 4.4.1 provides an example. The solution validation therefore includes a test for suitable coalition structures (see Section 3.4). For a valid transition each set of agents A_e that requires a location change, has to form a coalition structure CS^{A_e} so that $\forall A \in CS^{A_e} : mobile(\bar{A}) = 1$; the set of atomic agents can form any coalition structure as long as all formed agents are mobile to perform the transition.

4.3.7 Stage 7-9: Candidate Characterisation

In order to compare solution a quantification of cost according to Section 4.2.2 is required. A quantification of transition times is a required precondition for determining the solution cost or rather performance of the robotic organisation. But reconfiguration is an additional cost factor, since changing the coalition structure of an organisation requires additional operation time. A reconfiguration cost function has been already introduced in Chapter 3.5, and each space-time point can be associated with a reconfiguration cost, considering a reconfiguration from all incoming agents to the required agent, and also from the required agent to the outgoing agents. These cost are combined with the estimated transition times between two space-time points, which form the edge constraints of an STN. Since duration constraints are applied only at this stage, an infeasible plan can be detected, e.g., when encountering a negative cycle in the STN's distance graph. If the planner identifies a feasible solution efficacy, efficiency and safety can be computed. Otherwise the planner backtracks and turns to the next role agent assignment.

4.3.8 Optional Extensions

Heuristic based-search approaches often start from a feasible solution. But finding an initially feasible solution for a given mission problem can be costly. This depends upon the applied bounding strategy. The planning algorithm tries to minimise the use of atomic agents, and the initially tried lower bound is based on an concurrency analysis. The resulting number of agents can be still insufficient to solve the planning problem. To address that issue two optional extension have been evaluated. Extension 1 exploits encountered infeasible solutions and starts a refinement process. The refinement process involves an informed repair strategy of an infeasible solution. When encountering a flaw, one or more new constraints which target the elimination of this flaw are added to the mission specification. Each flow violation has a related incoming transport edge from the previous space-time point assignment for the agent role to the space-time point where the violation has been identified. Adding a min-distinction constraint between the affected space-time point and the previous on the route of the affected agent role. Hence, a min distinction constraint enforces an additional inflow to the current space-time point, from a different source space-time point. Alternatively, an additional functionality request is added for a *TransportProvider*, i.e., a mobile system with transport capacities: $minProp(s, \{TransportProvider\}, tcap, \geq i + \Delta)$, where i is the total inflow and δ the total flow violation for this space-time point. Combinations of flaw repairs are tested in a planner sandbox. The best combination - considering the reduction flaws - is applied. The planning

process is then restarted with the updated mission specification. Extension 2 takes advantage of the previously mentioned control parameters ψ_m and ψ_{-m} to control the agent type cardinality bound. By adapting or rather incrementing this bound, more (mobile or immobile) atomic agents are considering for solving the mission planning problem. This extension is similar to an incremental fleet size adaptation strategy found in combination with minimal fleet size optimisation approaches.

4.3.9 Implementation Notes

The constraint-based planning approach has been implemented using the CSP framework Gecode (Version 6.0.0) (Schulte, Tack, and Lagerkvist 2018), which offers low level and high level modelling capabilities. The implementation of the planner takes advantage of the following features: (a) integer constraints, (b) set constraints, (c) symmetry breaking, (d) value propagation, and (e) value branching. Each stage of the planning approach requires a particular selection of value branching strategies, which can significantly influence the performance of the planning approach. This thesis does not provide an evaluation on the best branching strategies for the given planning approach, but outlines the design of the overall planning approach. In addition to using the inbuilt features, custom constraints propagators have been implemented to support the temporal-expanded network and evaluate location access restrictions. The multi-commodity min-cost flow problem is constructed using a custom implementation of a front-end for graph-libraries (Roehr, Munteanu, et al. 2019). This library allows to represents graph with any of the following backends: Boost Graph Library (Siek, Lee, and Lumsdaine 2000), LEMON (Dezső, Jüttner, and Kovács 2011), or SNAP (Leskovec and Sosič 2016). The multi-commodity min-cost flow problem is translated into a linear program, and encoded into a CPLEX LP (LP Solve Community 2018) representation. Appendix G lists an example output. The use the standard format CPLEX LP guarantees interoperability with other solvers and permits to use a range of LP solvers including GLPK (Free Software Foundation 2015) and SCIP (Achterberg et al. 2008) with TemPl. To encode the mission planning problem an XML-based representation format has been developed which is detailed in Appendix G.

4.4 Evaluation

This mission planning approach combines the application of a constraint-based approach to generate a local optimisation problem which is subsequently solved with linear integer programming. In general, the application of a constraint-based planning approach with the solver Gecode allows for a complete search. As a generalisation of the TSP the VRP is, however, an NP-hard problem. Therefore the mission planning approach can only be the basis for a heuristic planning approach. The size of the planning problem is influenced by the number of timepoints $|T|$, locations $|L|$ and the set of active mobile and immobile agents which have to be considered for a solution. The following examples provide the required column and row numbers of the linear problem to solve the minimum cost flow; this serves as indicator for the size of each mission problem.

To illustrate general aspects of the planning approach a selection of simple examples is provided first. Subsequently, a complex space mission scenario is evaluated to demonstrate the scalability. For the organisation model only the most relevant information are embedded into

the illustrations. More detailed information about the atomic agent properties can be found in Appendix A.

4.4.1 On Demand Transport

Given a mission $\mathcal{M} = \langle \widehat{GA}, STR, \mathcal{X}, \mathcal{OM}, T, L \rangle$ with the following requirements:

$$\begin{aligned} \widehat{GA} &= \{(\hat{a}_0, 2), (\hat{a}_1, 2)\} \\ STR &= \{(\emptyset, \{(\hat{a}_0, 1), (\hat{a}_1, 1)\}) @ (lander, [t_0, t_1]), \end{aligned} \quad (4.20)$$

$$(\emptyset, \{(\hat{a}_1, 1)\}) @ (base1, [t_2, t_3]), \quad (4.21)$$

$$(\emptyset, \{(\hat{a}_0, 1)\}) @ (base2, [t_4, t_5]) \} \quad (4.22)$$

$$\mathcal{X} = \{t_0 < t_1, \dots, t_4 < t_5\}$$

$$\mathcal{OM} = \{mobile(\hat{a}_0), tcap(\hat{a}_0) = 10, \neg mobile(\hat{a}_1), tcon(\hat{a}_1) = 1, \dots\}$$

$$T = \{t_0, \dots, t_5\}$$

$$L = \{lander = (0, 0, 0), base1 = (1000, 0, 0), base2 = (0, 1000, 0)\}$$

The atomic agent type \hat{a}_0 corresponds to an instance of robot type Sherpa, and the agent type \hat{a}_1 corresponds to an instance of a payload item. A payload item is immobile but requires relocating from location *lander* to location *base1*. Figure 4.12 illustrates a flawed solution when a minimum role bound is enforced such that $\psi_m = 0$ and $\psi_{-m} = 0$ (see Equation 4.8) and when candidate repair is disabled.

With this setup the planner can find a solution, but has to explore the search space exhaustively in worst case. The encountered solution includes a single flaw: Constraint 4.21 is only be partially fulfilled. As a result of enforce a minimum role bound the not all four available atomic agents, but only two are considered for the solution. No concurrent activities for multiple agents of type a_0 or a_1 have been identified. Hence, the respective upper role bound per agent type is set to 1.

Figure 4.12 also illustrates how safety is computed with respect to the local requirements. The safety at the start depot is set to 1.0 per default. At space-time point $(base1, t_3)$ only a single payload item is required. A robot Sherpa is also available and can partially support the requirements, so that redundancy exist; in this case for the passive and active interfaces. Hence, the additional Sherpa robot leads to a local safety measure of 0.99. Reconfiguration cost have be accounted for at space-time point $(base1, t_3)$. The payload item remains at location *base1* while a Sherpa proceeds to *base2*, so that the initial coalition has to split.

Additional roles of the mobile agent type \hat{a}_0 , such that $\psi_m = 2$ in Figure 4.13, increase the redundancy level of the solution and can also increase the possibility to find a solution at all. The corresponding search statistics to the two solution approaches are listed in Table 4.8.

TemPl performs planning up to and including solution validation (see Section 4.3.6) only by using a qualitative temporal network, so that no metric information is considered. Hence, it produces a solution candidate where all relative synchronisation constraints holds. Solution candidate characterisation as described in Section 4.3.7 performs a qualitative analysis. Transitions between space-time points are constrained by the travel time of (the slowest) agents; reconfiguration cost increase the required travel time. The resulting simple temporal network is minimised to find an assignment. Here (in seconds): $t_0 = t_1 = t_2 = 0, t_3 = 2000, t_4 = t_5 = 6388.43$

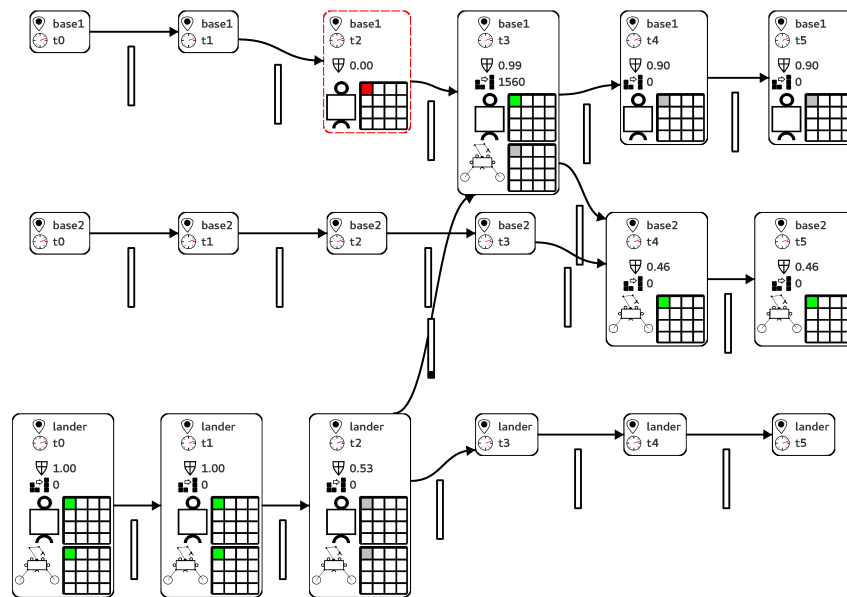


Figure 4.12: Flawed solution with the missed fulfilment of a requirement (red), while some requirements are fulfilled (green). All assignment of resources without requirement are marked in grey.

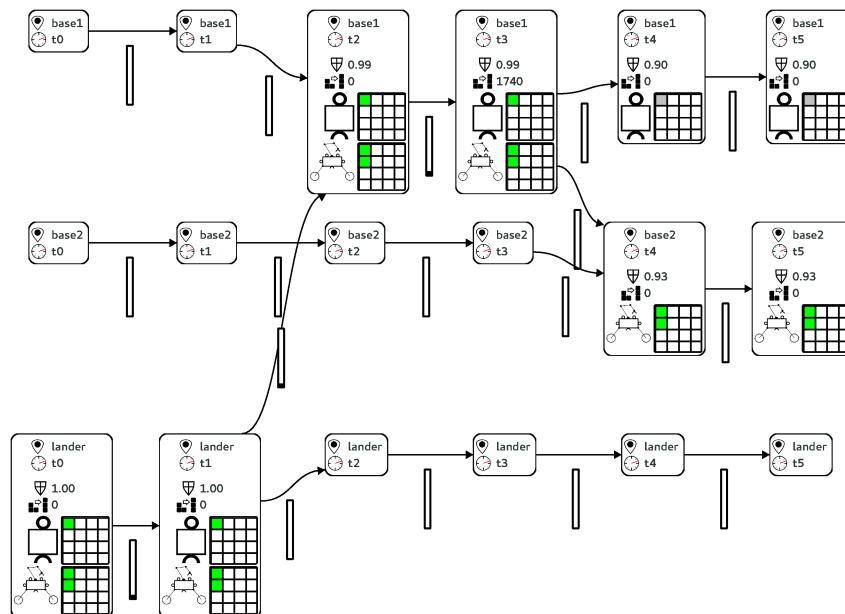


Figure 4.13: Solution for a simple mission with $\psi_m = 2$.

Table 4.8: Search statistics for a simple mission with minimal and extended agent assignment bounds.

Simple mission	$\psi_m = \psi_{\neg m} = 0$	$\psi_m = 2, \psi_{\neg m} = 0$
available atomic agents	4	4
required atomic agents	2	3
overall runtime	0.129 s	0.228 s
per solution runtime	0.129 s	0.228 s
<i>Constraint-based search</i>		
# of executed propagators	178	255
# of explored nodes	13	12
max search tree depth	12	11
# of performed restarts	0	0
# of generated nogoods	0	0
<i>Flow optimisation</i>		
# of constraints (rows)	79	79
# of variables (columns)	23	23

Table 4.9: Solution properties for a simple mission with minimal and extended agent assignment bounds.

Simple mission	$\psi_m = \psi_{\neg m} = 0$	$\psi_m = 2, \psi_{\neg m} = 0$
efficacy	0.67	1
efficiency	0.26 kWh	0.48 kWh
safety	0.46	0.90
time horizon	6388.43 s	6568.43 s
reconfiguration cost	1560 s	1740 s
travel distance	2414.21 m	2414.21 m

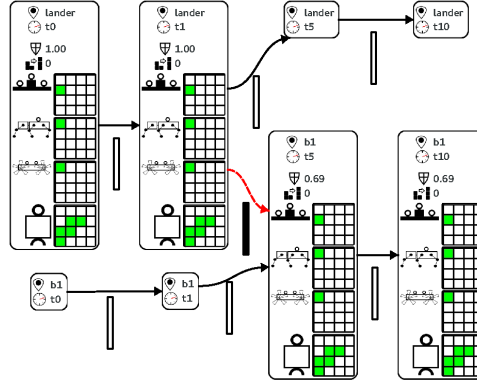


Figure 4.14: No suitable coalition structure can be found for a transition, leading to an edge flow (red).

for the solution with $\psi_m = \psi_{\neg m} = 0$. Table 4.9 lists the computed solution properties. Qualitative constraints apply only during the solution analysis, so that for any max duration constraint can still lead to detecting an infeasible solution. However, this is only true for any max duration constraint, since the addition of min duration constraints only stretches the solution with respect to time.

4.4.2 Constrained Coalition Formation

Section 4.3.6 describes solution validation which involves the feasibility check for all transitions between different locations. The solution validation step therefore must ensure that a (sub) coalition structure exists for the set of all participating agents, where all agents are mobile. Figure 4.14 illustrates an example, where this validation fails, so that the solution candidate has to be rejected. This example is based on the following mission specification:

$$\begin{aligned}
 \widehat{GA} &= \{(BaseCamp, 1), (CREX, 1), (CoyoteIII, 1), (Payload, 5)\} \\
 STR &= \left\{ (\emptyset, \{(BaseCamp, 1), (CREX, 1), (CoyoteIII, 1), (Payload, 5)\}) @ (b_1, [t_0, t_1]), \right. \\
 &\quad \left. (\emptyset, \{(BaseCamp, 1), (CREX, 1), (CoyoteIII, 1), (Payload, 5)\}) @ (b_1, [t_5, t_{10}]) \right\} \\
 \mathcal{X} &= \{t_0 < t_1 < t_5 < t_{10}\} \\
 OM &= \{\neg mobile(BaseCamp), \neg mobile(Payload), mobile(CREX), mobile(CoyoteIII), \dots\} \\
 T &= \{t_0, t_1, t_5, t_{10}\} \\
 L &= \{lander, b1\}
 \end{aligned}$$

The min-cost flow optimisation for this specification produces an agent assignment which is valid with respect to the capacity flow restrictions of the mobile agents. In the presented example, two mobile systems, namely CREX and Coyote III, are available for transporting other agents, but neither of the mobile systems can join with a BaseCamp: the mobile agents as well as the BaseCamp comprise only male interfaces. Hence, a transition from $(lander, t_1)$ to $(b1, t_5)$ is not feasible for all agents.

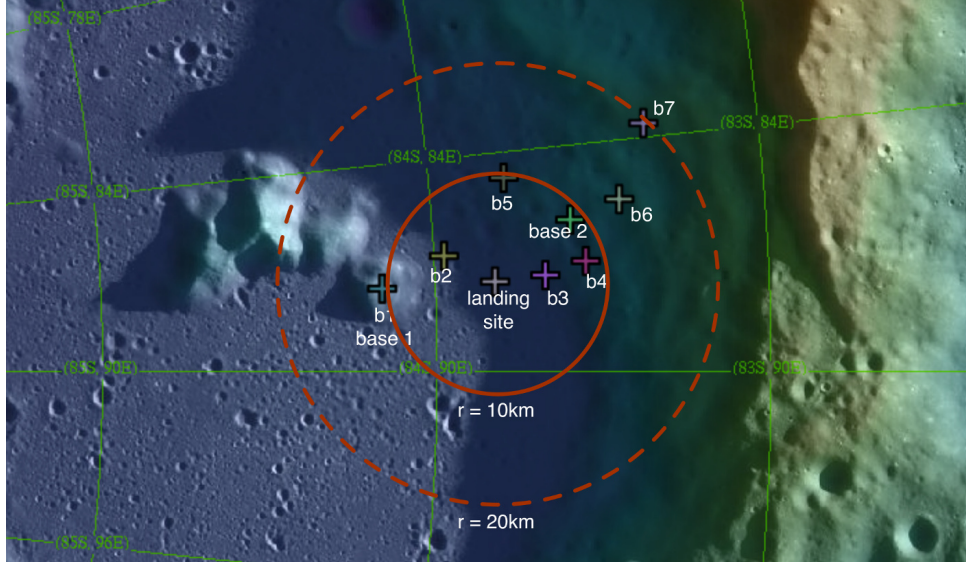


Figure 4.15: Possible locations with science goal defined in the project TransTerra for a lunar robotic exploration mission (extracted from Workpackage Document E2100.1).

4.4.3 Space Mission

Space and other robot missions will come with an increased complexity compared to the previously illustrated examples. The project TransTerra has anticipated a multi-robot space mission on the lunar surface and Figure 4.15 highlights a selected set of locations for science goals. These target science goals are cast into a respective mission specification, which is listed in the following. Note that each location is defined by longitude and latitude in the specification. TemPl uses Mercator projection (PROJ Contributors and Open Source Geospatial Foundation 2018) to convert these coordinates.

$$\begin{aligned}
 \widehat{GA} &= \{(BaseCamp, 5)(CREX, 2)(CoyoteIII, 3)(Payload, 16)(SherpaTT, 3)\} \\
 STR &= \{ \{ \{ (BaseCamp, 5), (CREX, 2), (CoyoteIII, 3), (Payload, 16), (Sherpa, 3) \} @ (lander, [t_0, t_1]), \\
 &\quad (\emptyset, \{ (Payload, 3) \}) @ (lander, [t_5, t_{10}]), \\
 &\quad (\{ LocationImageProvider, EmiPowerProvider \}, \{ (Payload, 3) \}) @ (b_1, [t_2, t_3]), \\
 &\quad (\emptyset, \{ (Payload, 1) \}) @ (b_1, [t_3, t_{14}]), \\
 &\quad (\{ LogisticHubProvider, LocationImageProvider, EmiPowerProvider \}, \{ (Payload, 3) \}) @ (b_2, [t_2, t_3]), \\
 &\quad (\emptyset, \{ (BaseCamp, 1) \}) @ (b_1, [t_4, t_7]), (\{ LocationImageProvider \}, \{ (Payload, 3) \}) @ (b_4, [t_6, t_7]), \\
 &\quad (\emptyset, \{ (Payload, 3) \}) @ (b_4, [t_8, t_9]), (\emptyset, \{ (Payload, 1) \}) @ (b_6, [t_{10}, t_{14}]), (\emptyset, \{ (Payload, 3) \}) @ (b_7, [t_{12}, t_{14}]) \} \\
 \mathcal{X} &= \{t_0 < t_1, \dots, t_{13} < t_{14}\} \\
 \mathcal{OM} &= \{ \neg mobile(BaseCamp), mobile(CREX), tcap(CREX) = 2, mobile(CoyoteIII), tcap(CoyoteIII) = 4, \\
 &\quad \neg mobile(Payload), mobile(Sherpa), tcap(Sherpa) = 10, \dots \} \\
 T &= \{t_0, \dots, t_{14}\} \\
 L &= \{lander = (lat : -83.82009, long : 87.53932, moon), b_1 = (lat : -84.1812, long : 87.60494, moon), \\
 &\quad b_2 = (lat : -83.96893, long : 86.75471, moon), b_3 = (lat : -83.66856, long : 87.42557, moon), \\
 &\quad b_4 = (lat : -83.54570, long : 87.09851, moon), b_5 = (lat : -83.82009, long : 84.66000, moon), \\
 &\quad b_6 = (lat : -83.77371, long : 84.70960, moon), b_7 = (lat : -83.34083, long : 84.64467, moon) \}
 \end{aligned}$$

TemPl has been used to search for solutions to this mission scenario with a setting of ψ_m in the range of 0 to 2. The search has been split into epochs with a maximum allowed planning time of 60 s. After 60 s search has been reinitialised in order to escape from local minima. Planning has been stopped after either after memory has been exhausted or when a total planning time of 20 min has been exceeded. A higher setting of ψ_m requires additional agents to be used for the planning approach, so a higher computation time is to be expected. While the setting with ψ_m results in a stable range below 4 s per solution candidate, the required time increases for $\psi_m = 2$ to up to 12 s. The Linear Program (LP) problem sizes remain, however, at a comparable level, despite the bound relaxation as shown in Figure 4.17b. The comparison of solutions based on efficacy, efficiency and safety as introduced in Section 4.2.2 is shown in Figure 4.16. A detailed analysis as provided in Appendix G.2 indicates that an increase of ψ_m helps to identify safer solutions at the cost of efficiency.

Figure 4.18 illustrates a feasible solution for this mission, where the planning parameters are set to $\psi_m = 2$ and $\psi_{-m} = 0$. Visual inspection of the solution shows further optimisation potential at $(b7, t9)$ and $(b6, t5)$. This potential could be exploited by applying further improvement heuristics (cf. (Toth and Vigo 2014, p.135)) in future works.

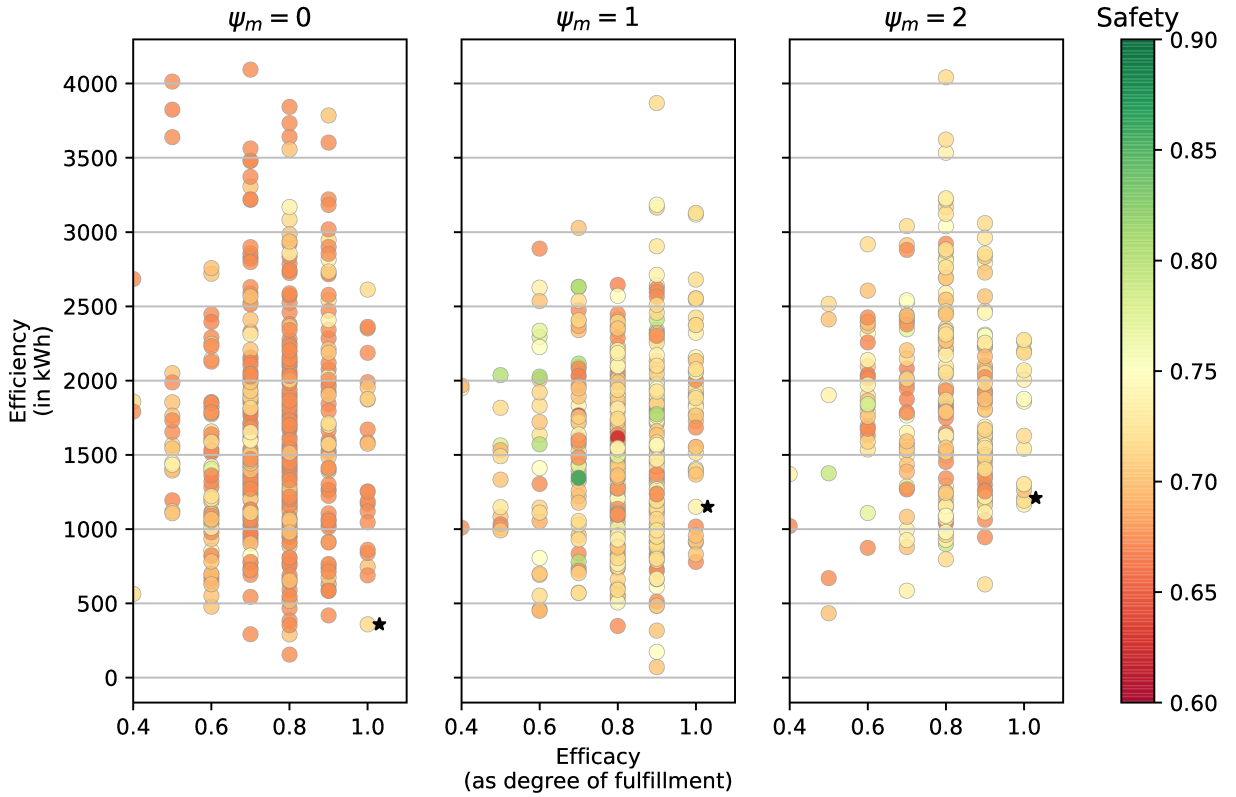
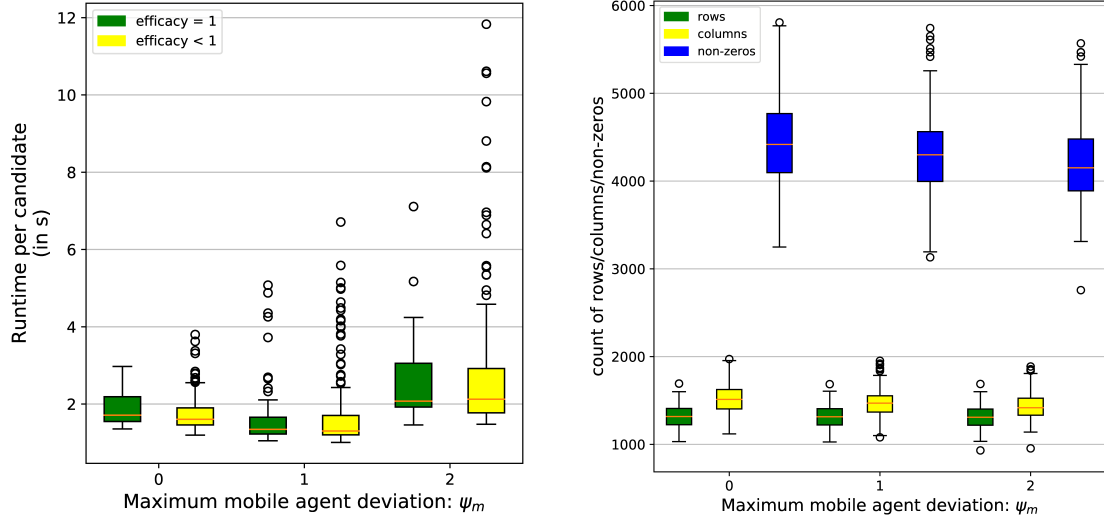


Figure 4.16: Solution landscape compared for different bound settings. Black star-shaped markers identify the current local best solutions, where the cost function is parametrised with $\alpha = -1.0$, $\beta = 100.0$, and $\epsilon = 10.0$.



(a) Required computation time for a solution candidate.

(b) Size of the underlying solved LP problems.

Figure 4.17: Performance and LP problem size comparison with respect to ψ_m for the example space mission.

4.5 Discussion

Tackling the Combinatorial Challenge Non-temporal and temporal classical planning approaches have a limited applicability regarding planning for reconfigurable multi-robot systems since they rely on an explicit and static representation of planning domains. Although very effective heuristics exist in guiding forward search, classical approaches still require operator instantiation which can be prohibitive when considering all feasible agents in a reconfigurable system. The planning approach implemented with TemPl initially avoids the full instantiation by relying on a compact, cardinality based description of agent types. Furthermore, the mission specification is based on a resource-oriented requirement definition. Functionality constraints are combined with explicit agent type constraints to formulate mission constraints on the basis of an available mapping between functionality and agent types. This mapping is provided by the organisation model MoreOrg. The computation of the functionality mapping has worst case computational cost of $\mathcal{O}(2^{|A|})$ when all agent types have to be exhaustively evaluated. Knowledge about the relationship between structure and function can, however, be exploited to prune irrelevant types from the search space. This observation has led to the development of the functional saturation bound in this thesis. Eventually, the application of the functional saturation bound is the key element to efficiently identify suitable agents which can satisfy mission requirements.

Distinction to VRPs and Classical Planning Compared to existing VRP based approaches the mission planning problem embeds most of the known properties, such as time windows, capacity, heterogeneity and fleet size, but also comes with distinctive features. While the properties are mainly treated separately in VRPs, TemPl offers a combined approach. Additionally, TemPl is not only accounting for commodity demand, and hence the presence of single sys-

tems, but combines demand for commodities with the demand for functional properties of agents. For that purpose, required functionalities are resolved to actual agents. Thereby a functionality driven mission design becomes feasible, while the set of required agents is selected from a pool of available agents.

Classical planning approaches typically define the achievable goal as a particular world state, or by one or more tasks that have to be performed. The relation to the former action based approaches exists through the organisation model which encodes the domain description into the organisation state. A relation to the task planning approaches can be established, as soon as a mapping from a task definition to functional requirements or agent roles exists. As an extension, this task model can also be embedded into the organisation model using a similar approach to modelling functionalities.

VRPs's reason quantitatively with time and use soft and hard time constraints. This limits the options for coordination of activities; to state that two customers shall receive their goods at the exact same time hard time constraints must be used. TemPl's usage of qualitative temporal constraints enables a partial ordering of requirements and leads to a generically applicable synchronisation for agent activities. As long as no duration constraints are involved a valid plan computed by TemPl represents a reusable synchronisation template. Effectively, existing solutions can then be used as a whole or even in parts as synchronisation templates.

A multi-pickup multi-delivery problem is part of TemPl's mission planning problem. The initial motivation for multi-pickup and multi-delivery is the planning for a long-term multi-robot operation where agents are reusable. Although atomic agents are reusable, in real missions they might also be stateful. This chapter has illustrated an exemplary sample-return mission with limited reusability of a sampling module. To partially address this issue TemPl allows to constrain the route of agent roles using min/max/all equal constraints. However, TemPl is, however, not a generic planner and compared to PDDL-based it cannot generically handle domain and problem descriptions. Furthermore, it does not consider temporary usage restrictions based on an agent state, e.g., when energy level is too low.

Reconfigurable multi-robot systems have a flexibility to exchange resources and balance the resource distribution. Thereby, an organisation's redundancy level can be maintained to serve as a safety buffer. The developed planning approach accounts for the organisation's redundancy state and characterises the safety of a plan by the lowest redundancy level. In effect, TemPl illustrates an approach to deal with inter-route constraints by enabling a mechanism to trade-off solution efficacy, efficiency, and safety.

Scalability The planning approach embeds multi-commodity min-cost flow optimisation as local search after a (partial) plan has been generated. The multi-commodity min-cost flow optimisation is translated into a linear integer program, which can subsequently be solved with any available linear integer program solvers including GLPK (Free Software Foundation 2015), coin-or/CBC, coin-or/CLP (Lougee-Heimer 2003) or SCIP (Achterberg et al. 2008). GLPK and SCIP use for example different solution strategies. GLPK applies the simplex algorithm to solve and optimise the multi-commodity min-cost flow problem. Without using presolving, even infeasible solutions can be inspected and used as input for a local repair approach. This is not directly possible for SCIP which uses a combination of constraint-based programming and mixed integer programming, but provides a plugin mechanism to use column generation.

The size of the multi-commodity min-cost flow problem as linear program depends upon the

size of the temporally expanded network, as well as the number of mobile and immobile agents which need to be routed. Therefore the scalability of the planning approach is currently limited by the used LP solvers. An improvement could be achieved using an implicit representation of the search space, as it has been done for the organisation model. This means that the current local optimisation strategies have to be adapted to using a column generation approach which allows for a dynamic and scalable optimisation approach. Alternatively, heuristic search approaches can be considered, e.g., such as Tabu Search, Very Large Neighbourhood Search (VLNS) or destroy-repair search algorithms in general. When these approaches require an initial (feasible) solution to start from, TemPl can still be used for boosting other algorithms.

Expressiveness of the Mission Specification TemPl's mission specification is a combination of constraints which are typically tackled by special variants of VRP formulations. As a result a mission specification allows to define: spatio-temporal requirements for functionality and agent presence, partial or full paths of atomic agents, and high-level synchronisation between multiple atomic agents. The presented constraints are only a subset of applicable (meta-)constraints and they limited to persistence constraints. The current mission specification likewise model event constraints as follows: Assuming a mission \mathcal{M} which uses a time interval $[t_0, t_1]$; the requirement of an agent type \hat{a}_0 to appear at location l at any time within this interval requires the addition of the following constraints: $STR' = STR \cup (\emptyset, \{(\hat{a}_0, 1)\})@ (l, [t_s, t_e])$ and $\mathcal{X}' = \mathcal{X} \cup t_0 \leq t_s, t_e \leq t_1$. The mission specification can currently not express transition constraints, so that the provisioning of functionality is not guaranteed throughout a transition.

The current constraint-based mission representation does not support the definition of coalition structure constraints for a single location. For instance, limiting the cardinality of the functionality to maximum one results from this limitation. Instead such constraints have to be modelled with additional locations at the cost of a larger space-time network.

The planner TemPl considers a subset of the intra-route and intra-route constraints as presented in Section 4.1.1 mainly focusing on resource limitations and synchronisation. Efficiency and safety are non-functional properties and used as optimisation criteria. They could also be considered for use as spatio-temporal, intra-route, or inter-route constraints. Such an extension would permit an more detailed problem representation with respect to non-functional requirements.

Usage of the Organisation Model The organisation model MoreOrg and its ability to dynamically represent agents are the basis for the mission planning approach. The organisation model serves as knowledge base to describe agents and resource models. It also serves as domain specific reasoner for composite systems. Although MoreOrg uses a generic resource representation, the mission specification uses two subconcepts: agent and functionality. This separation is not strictly necessary and could be further generalised by permitting resources in general in spatio-temporal requirements.

The special needs of reconfigurable multi-robot systems have been accounted for by introducing heuristics and policies, e.g., to select the transport provider in a composite agent. TemPl uses the defined heuristics and policies to compute the cost of a solution. While the current set of heuristics and policies is clearly domain specific and based on some strong assumptions, it points to further research opportunities to improve generalisation.

Multi-objective Optimisation TemPl has been developed to serve as basis for a safe, effective and efficient operation of reconfigurable multi-robot systems. As such, the mission planning problem turns out to be a multi objective optimisation problem where specialised optimisation strategies could be applied. The safety target competes with the target of efficiency, and hence requires balancing, where the primal objective remains effectiveness. Future research requires a detailed evaluation of the multi-objective optimisation problem, where this thesis provides a basis to study reconfigurable multi-robot system and develop new application strategies. A safety analysis of a timeline which is discretised and evaluated only at given timepoints neglects the transition, which take a big share in the overall mission. A team of agents which always (including transitions) operates in a closer range, will be quicker to help each other, e.g., to exchange failing components (cf. Chapter 3). Thus, such state should be honoured with a higher safety level. For this reason, the overall state progression of the organisation model has to be considered and interpolated for a better computation of a safety estimate for the mission.

Transferability The mission specification and overall organisation model has been developed with a focus on reconfigurable multi-robot systems. TemPl offers a generic solution approach which contains several challenges of classic VRPs including CVRP. Most of these problems use restrictions, which have to be explicitly encoded into a mission specification for TemPl. The CVRP, for instance, can be modelled as described in the following.

The definition of the CVRP requires an organisation model which describes two atomic agent types \hat{a}_v and \hat{a}_c , where \hat{a}_v represents *vehicles* and \hat{a}_c *commodities*. The ability to link vehicles and commodities is encoded through two interface classes. The concepts of male and female interfaces EmiActive and EmiPassive are reused. While a vehicle has only one interface EmiPassive, each commodity has one EmiActive and one EmiPassive. This setup prevents coalitions of two or more vehicles, but (initially) allows to attach an arbitrary number of commodities to a vehicle. A vehicle type has, however, a problem specific transport capacity $tcap(\hat{a}_c) < UB_{tcap}$. The mission specification assigns all available agents defined by the agent pool \widehat{GA} to an initial depot location, so that $s_{init} = (\emptyset, \widehat{GA}) @ (l_{depot}, [t_0, t_1])$, where t_0 represents a setup vertex. The vehicles are also assigned to a final depot accordingly. Since no time windows are given, synchronisation of vehicles is not needed. The standard TSP constraints also demand that each vehicle visits a customer/location only once. Hence, to translate the standard CVRP into a mission description for TemPl requires the addition of the following constraints: $allDistinct(S, \hat{a}_c)$ and the $maxAccess(l_x, \hat{a}_v, 1)$, where $maxAccess$ represents a location access constraint only for mobile agents. Note that location access restriction can also be encountered in VRPMSs (Drexler 2013). Solving a CVRP with the more generic mission planning approach is less efficient compared to a specialised solution approach. The use of the temporally expanded network introduces unnecessary side constraints and idle times for vehicles. Additionally, for a comparison with existing benchmarks the cost function needs to use covered distance of the vehicles instead of the here suggested energy consumption.

4.6 Summary

Automated planning is a key component to achieve the autonomous operation of reconfigurable multi-robot system. It provides a means to exploit the high degree of flexibility of these systems and eventually contributes to implementing complex adaptive, yet controllable systems.

To support this goal, this chapter outlines the mission planner TemPl which uses the organisation model MoreOrg to exploit the flexibility of reconfigurable multi-robot systems. A mission specification represents the planning problem for reconfigurable multi-robot system and it builds upon a temporal database representation. Since robots are embodied systems, the representation does not rely on a classical representation using only temporal qualified expressions, but adds locality as mandatory qualification leading to an overall representation of space-time. Although classical and temporal planning approaches can be applied to the mission planning problem, they are of limited applicability and scalability due to the possible number of combinations when dealing with modular robotic systems. To deal with this combinatorial challenge, the presented planning approach relies upon the organisation model MoreOrg which provide a dynamic agent and organisation domain representations. This approach is a necessary precondition for a practical planning approach with reconfigurable multi-robot systems that account for agent combinations. The use of this dynamic agent representation proves to be a distinctive advantage in comparison to an explicit instantiation for planning domain representations in classical approaches.

This chapter describes a constraint-based formalisation to generate plan candidates which are refined or and optimised using local search, here by solving an extended multi-commodity min-cost flow optimisation problem. The candidate generation relies on the functional saturation bound provided by the organisation model to focus on relevant resource assignments. This leads to a mission planning approach which takes heterogeneous systems, vehicle capacities, time windows, vehicle synchronisation and fleet optimisation into account. The planning approach is based on approaches found in the research area of VRPs. As suggested by Drexel (2012), TemPl combines multiple areas of research, but most importantly uses operations research and robotics research to plan with reconfigurable multi-robot systems. Hence, TemPl is an interdisciplinary planning approach. It abstracts the problem definition and models developed by Drexel (2012) for VRPTT and VRPMSs and provides a generalised approach to account for reconfigurable systems in planning, although limitations remain.

The flexibility to share resources in a reconfigurable multi-robot system permits a robotic organisation to dynamically optimise its safety properties. Especially for space applications the consideration of safety and risk minimisation has a high priority. Since space is the targeted planning domain, TemPl embeds safety or rather probability of survival as optimisation objective. It complements the typically used objectives efficacy and efficiency to create a risk aware planning approach. Planning under consideration of safety can be a contributor to achieving long term autonomous robotic operations.

In summary, TemPl introduces the first temporal mission planning approach which is dedicated to reconfigurable multi-robot systems. It is expected to set a stimulus to develop highly modular, yet, safely adaptive multi-robot systems which can achieve mission goals constrained by efficiency and safety requirements.

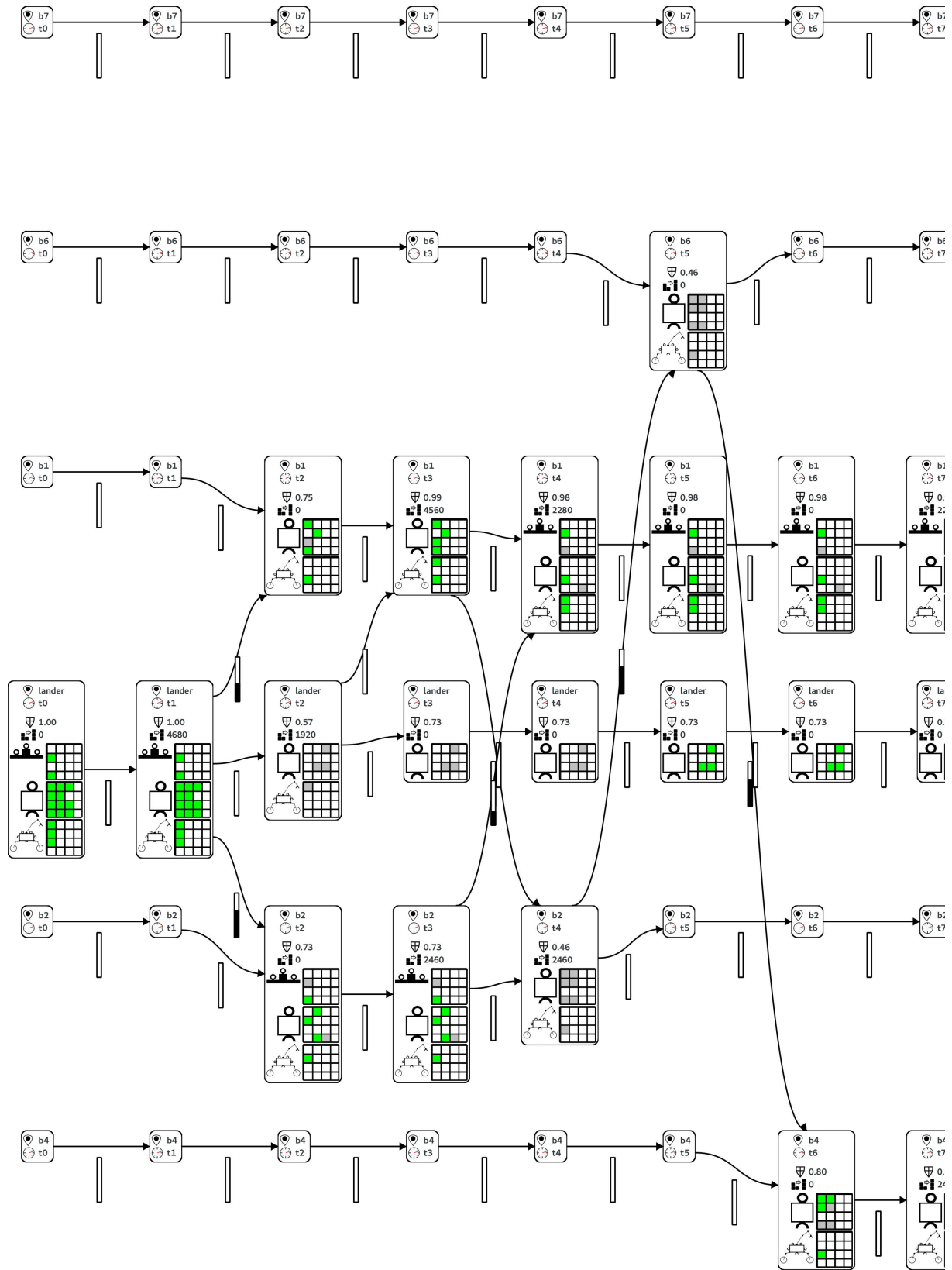
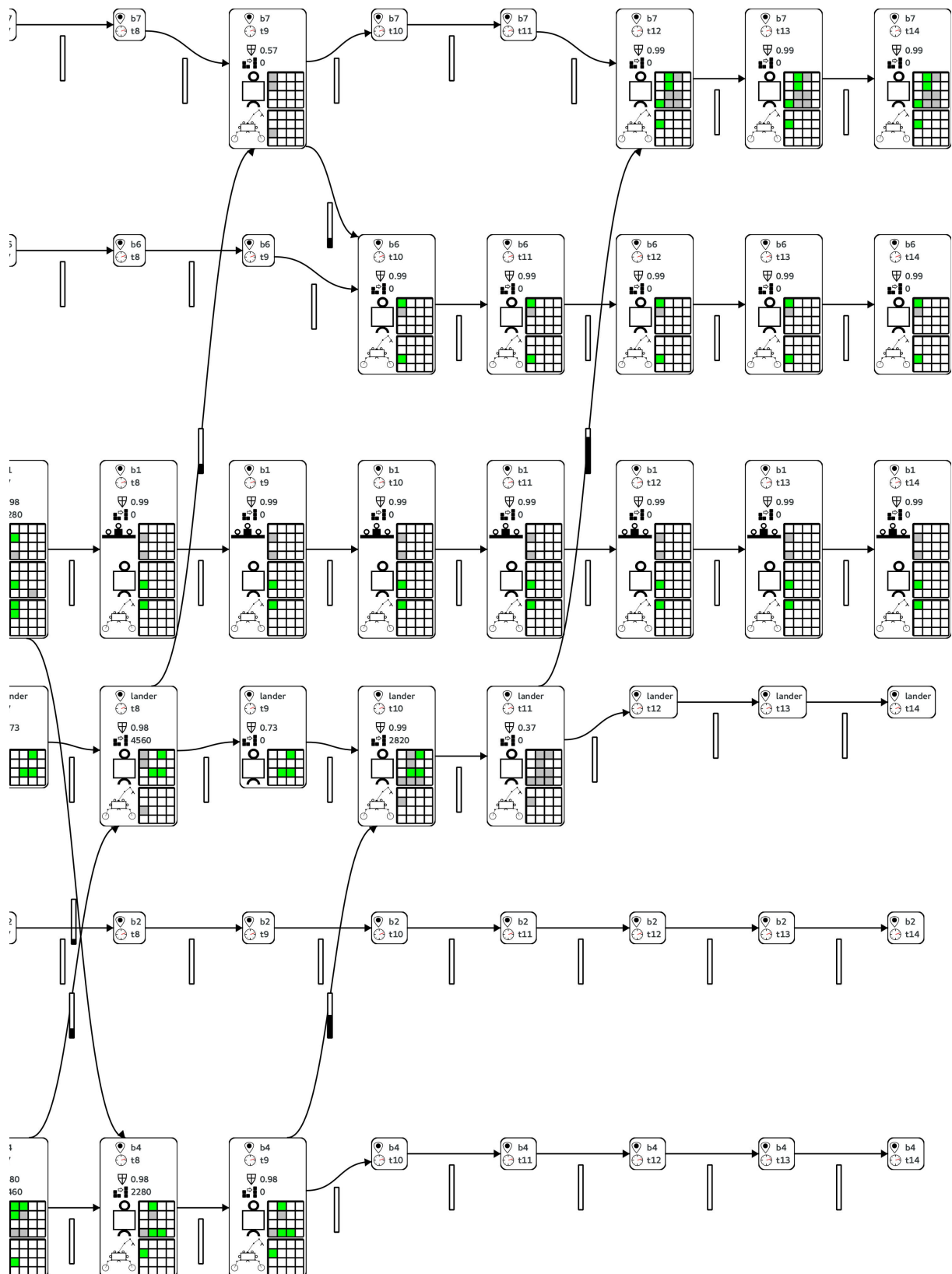


Figure 4.18: Identified solution for the exemplary space mission after 20 min of search ($\psi_m = 2, \psi_{-m} = 0$)



5

Operational Infrastructure

Disclaimer *This chapter introduces and expands on works which have been published in parts in (Joyeux, Schwendner, and Roehr 2014; Roehr, Cordes, and F. Kirchner 2014; Roehr and Herfert 2016; Roehr and Willenbrock 2018).*

A multi-robot system represents a distributed system. Therefore it inherently offers a higher degree of robustness compared to single robot systems. A single failing agent does not lead to a fully dysfunctional system, so that a single point of failure can be avoided. To make sure, however, that this assumption holds, a physically distributed multi-robot system requires a likewise distributed control approach.

The exploitation of reconfigurability in multi-robot systems has requirements. Capable modular hardware is an obvious precondition, yet, the full exploitation of distribution and reconfigurability is only possible with a suitable operational infrastructure. The infrastructure has to support distributed, dynamically appearing and disappearing agents. It is also the basis to control reconfiguration and adapt software to hardware changes. A coalition structure can change as a result of system optimisation or following a mission plan leading agents to change their operational state between operative and dormant. Dormant agents do not leave the overall organisation, but any reconfiguration can still be disruptive in the sense, that one or more atomic agents remains temporarily or permanently unpowered. The coalition structure also changes when additional atomic agents enter the organisation. For supporting an incremental mission approach new agents have to be transparently included into the existing agent organisation. Establishing standards for communication and knowledge representation is therefore a necessity for inter-robot communication, knowledge sharing and cooperation.

Software as well as hardware have to be managed for an application of a reconfigurable, heterogeneous team of robots. Firstly, software and hardware development efforts have to be based on modularisation to facilitate maintenance and enable the reuse of components. Secondly, existing space missions have shown that maintenance and software upgrades of robots are inevitable for long term operations. The reason for software updates is manifold ranging from the fixing of bugs, addition of new features and updating and extending the general knowledge basis of a system. Reconfigurable multi-robot systems even allow for a limited way to upgrade the hardware of remotely operating robotic systems - which is the basis for an incremental space mission design discussed in Chapter 2.3. A hardware upgrade is likely depending upon a software update. Although software updates can be applied more easily compared to hardware updates, they have to be synchronized for all agents, e.g., to maintain interface compatibilities. This latter aspect is a challenge that is generally encountered in software development and essential for maintaining a multi-robot system.

This chapter describes the reference implementations of core parts of a modularly designed operational infrastructure for a reconfigurable multi-robot system. The infrastructure is based on integrating state-of-the-art technologies, so that individual elements come with marginal novelty. It represents, however, a unique combined work which is focussed on reconfigurable multi-robot systems. Hence, the following sections focus on the essential requirements and characteristics of this architecture to support reconfigurable multi-robot systems.

Section 5.1 provides background information about existing space architectures and architecture templates as well as robot architectures, and communication systems. Section 5.2 deals with the design and implementation of a distributed multi-robot communication architecture which can support the dynamics of a reconfigurable multi-robot system. The communication architecture addresses the challenges for inter and intra-robot communication including the handling of unreliable communication channels. Section 5.4 outlines the control architecture and control approaches to deal with reconfiguration. This chapter concludes with a discussion in Section 5.5 and a summary in Section 5.6.

5.1 Background

The requirements for a reconfigurable multi-robot system architecture are derived from the basic technical needs for operating a robotic system, as well as the intended application scenarios involving incremental missions, planetary exploration and the management of logistic chains. Hence, design decisions have been taken with respect to maintaining a modular and extensible software and agent architecture. This background section looks at architectural templates as basis for the actual implementation. Due to the importance of distributed communication for the actual operation of a reconfigurable multi-robot system, the background section also covers mobile ad-hoc networks.

5.1.1 Architectural Templates

To implement a multi-robot control architecture an abundant set of software architectures and architecture templates exists with varying degrees of applicability and level of details. The following sections list a selected set of reference architectures which are related to the implementation of a control architecture for reconfigurable multi-robot systems.

FIPA Abstract Architecture

The FIPA Abstract Architecture (Foundation for Intelligent Physical Agents 2002a) provides a template for the software architecture of physical multi-agent systems with focus on agent communication. FIPA intends to establish agent interoperability by defining communication standards and is therefore of particular interest for an application with heterogeneous multi-robot systems. The architecture specifies services and message formats which in combination define a so-called Agent Communication Channel. This Agent Communication Channel interconnects agents for a speech-act based information exchange (Searle, Kiefer, and Bierwisch 1980). This communication channel is established through several main services: message transport service, agent directory service, and service directory service. A message transport service acts correspondingly to a post office. First, it receives an envelope also referred to as letter, which contains a message. Then it uses the information of the letter to identify the recipients. In case the current message transport service is directly associated with the recipient it delivers the letter. Otherwise the letter is forwarded to recipient's associated message transport service. An agent directory service permits to identify the message transport service which is responsible for the recipient agent: each agent registers itself in this directory with a communication endpoint; here the related message transport service.

Table 5.1: Structure of a message according to the FIPA Standards (Foundation for Intelligent Physical Agents 2002b).

Category	Parameter	Description
Type of communicative act	performative	Classification into: <i>inform</i> , <i>request</i> , <i>propose</i> , <i>not_understood</i> , ...
Participant(s)	sender	Sender of the message
	receivers	One or more receivers of the message
	reply-to	One or more receivers of the message - allows for redirection
Content of message	content	Main payload of the message with arbitrary content, which can be encoded in different ways
Description of content	language	User-defined language of the content, e.g., label on how to interpret content
	encoding	Encoding (bit-efficient, xml, string) of the content, e.g., use to reduce message size
	ontology	Ontology that should be used when interpreting the content
Control of conversation	protocol	Interaction protocol selection to validate the protocol conformance of communication
	conversation-id	Identify a particular dialogue between agents, which follows an interaction protocol
	reply-with	Conversation label
	in-reply-to	Continue a labelled conversation
	reply-by	Request a reply within a certain time frame

The exchange of information is based on messages, which are standardised according to the fields listed in Table 5.1. The fields *sender*, *recipients* and *content* are mandatory. A FIPA mes-

sage can contain arbitrary payload in the *content* field. To communicate how to process and understand the data in the content field, the *content language* field allows to provide a label. Given that a shared understanding exists on the label, agents can thereby identify whether and how they are able to process the content field. Since the general concept of FIPA communication is based upon speech-act theory (Searle, Kiefer, and Bierwisch 1980), FIPA message communication assumes that agents can enter one or more conversations on particular topics with each other. This approach is a distinctive element compared to robotic frameworks which often neglect this element of conversation or rather stateful communication. The usage of speech-act theory is reflected in the consideration of interaction protocols and conversation. Interaction protocols allow a high-level structuring of communication based on so-called performatives, so that a default structure of a conversation between two agent can be described even without knowing the actual content of the messages. The use of conversation which follow a particular interaction protocol therefore allows for the potential validation of distributed agent communication, e.g., to achieve auction based agreement (Weiss 2009). The FIPA standards suggest the use of content languages such as FIPA-SL (Foundation for Intelligent Physical Agents 2002a) or KIF (Genesereth 1998), but custom content languages can also be defined. A number of implementations of the architectural template exist with a focus on software agents and Java Agent DEvelopment Framework (JADE) (Bellifemine, Poggi, and Rimassa 1999) has been the first reference implementation. Other approaches are JackTM (Winikoff 2005), FIPA-OS (Poslad, Buckle, and Hadingham 2000) and Mobile-C (Chen, Cheng, and Palen 2006). All reference implementations are JAVA-based except for Chen, Cheng, and Palen's Mobile-C, which is based on C/C++. Mobile-C's focus is however on software agents and migration of software agents across different physical machines. Hence, despite the popularity of FIPA standards in the field of multi-agent systems, an adoption of these standards in the area of physical agents is rarely seen.

CCSDS Reference Architecture

The Consultative Committee for Space Data Systems (CCSDS) is an organisation which has been formed by a number of space agencies to establish standards across the space industry with focus on communication and data systems. Similarly to the FIPA Abstract Architecture the CCSDS suggests a Reference Architecture for Space Data Systems (RASDS) for an application in spaceflight systems. A comparison between the FIPA and the CCSDS approach exists (CCSDS MOIMS SM&C and IEEE FIPA Comparison 2007) and the comparison shows several similarities between the two approaches, including the message transport facilities and the use of interaction patterns. The comparison suggests that FIPA's approach is more general and flexible, whereas CCSDS's approach is more restrictive due to its known application context and focus on space applications. In contrast to FIPA, the CCSDS standards consider access control, quality of service, and require a message abstraction layer. Meanwhile, FIPA standards consider a fully distributed communication system, aka peer-2-peer communication, something which is of no concern for the CCSDS communication system. Hence, FIPA standards are better suited for an application in a multi-robot scenario.

The comparison shows that FIPA and CCSDS approaches are closely related and partially overlapping. FIPA based applications are expected to be adaptable for integration into a space related software system. Hence, using a FIPA-based implementation approach is a comprise to maintain a level of flexibility for novel, research oriented application, while at the same time targeting a future space application.

Functional Reference Model

As part of the identification of development and design techniques for space mission Putz and Elfving (1991) have suggested a set of reference models including the so-called Functional Reference Model (FRM), the Application Reference Model (ARM) and the Operations Reference Model (ORM). The FRM has been developed to establish a methodology to design space control system with the help of a generic template which can fit all kind of control architectures. Figure 5.1 depicts the basic structure of this template, which is split into three vertical and three horizontal layers. The vertical decomposition leads to a hierarchical control layout composed

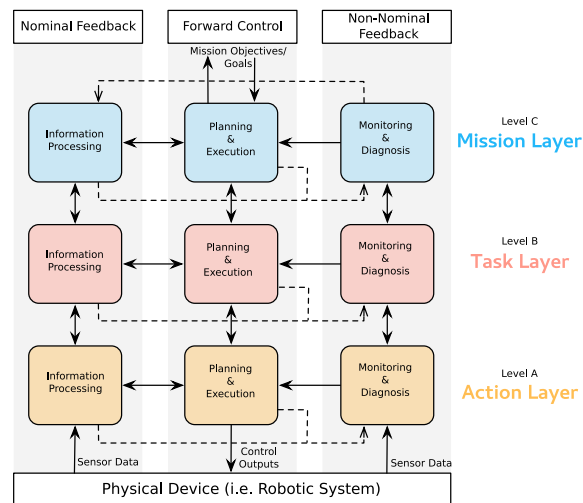


Figure 5.1: The basic structure of the *FRM* as described by Putz and Elfving (1991, p. 94).

of three execution, planning and control layers for the mission (C), its tasks (B) and resulting actions (A). Assuming an application of this reference model for robot control, layer A provides the lowest level of granularity and highest level of detail. Its implementation corresponds to the actual control system on a robotic system, and an action corresponds to the execution of one or more control functions of a robotic system, e.g., opening or closing a gripper. More complex robot activities or rather tasks require the sequencing of multiple actions, which are part of layer B. An overall robotic mission is based on the sequencing of a set of tasks at layer C. Nominal (sensor) feedback allows to establish sensor-based control loops, while a non-nominal feedback channel allows to extract significant information from the nominal and forward control channel, e.g., to identify strategies for failure handling. Each layer can provide local failure handling, but when the layer fails to identify and apply the failure handling approach, it has to delegate the error handling to the next upper (commanding) layer.

"The FRM, because of its extremely generic nature, is necessarily vague in details" (Putz and Elfving 1991, p. 25), and since it is only an architectural reference model, it does not provide implementation details. However, the FRM has been complemented with the ARM and ORM. The ARM details a FRM-based control architecture by focusing on the implementation of a particular application, and Putz and Elfving give multiple application classes as examples among which are: control of robot systems and control of surface roving vehicles' motion. Working out the ARM for a particular application class corresponds to the identification of application specific control loops, actions, and action sequencing. The overall development approach follows a rather strict hierarchical decomposition approach to identify and design the capabilities

of individual robots with respect to a specific space mission. This approach still leaves the detailed design of software interfaces open to the developers who implement the architecture.

Finally, the ORM groups operational aspects along three dimensions. The first dimension allows to assign the performance of a function to either a human or a machine. The second dimension distinguishes between an online performance and an offline performance of a tool. The third dimension considers the application of functions on-board/on-orbit or on-ground, e.g., while the former requires space qualified software development, the latter does not. A classification along these dimensions defines the requirements for systems or persons who perform and/or trigger an activity. Hence, this classification clarifies needs and responsibilities, and facilitates the identification of development needs, e.g., for the interface design and space qualification of components.

5.1.2 Robotic Frameworks

Developing software in any domain comes with a significant number of repetitive tasks which can be automated or supported by standardised workflows and a set of utilities. In robotics many frameworks exist which intend to facilitate software development.

The most popular framework in this area is the Robot Operating System (ROS) (Quigley et al. 2009). ROS has established a software development ecosystem around a publish-subscribe communication architecture. A ROS system consists of a set of so-called nodes, which can publish data under a particular topic name and to receive data a node subscribes to a topic. Hence, the data of a node publishing data on a particular topic can be received by any other node that subscribes to the same topic. A ROS-based communication architecture requires a ROS Master (node) for the publish-subscribe mechanism to work. The ROS Master demands a centralised communication approach, although workarounds exist to connect multiple ROS Master nodes to allow an application with multi-robot systems. The functionality of a ROS-based robotic system is represented as a network of nodes, where each node can encapsulate a single data processing algorithm. The simplicity of the communication architecture permits fast prototyping and enables a wide range of applications. The use of a publish-subscribe mechanism, however, can be restrictive and does not easily allow to control the design of the communication network since it cannot be defined how exactly data should flow in the communication network. ROS 2 is successor of ROS, and it tackles a number of shortcomings of the initial implementation of ROS. However, to achieve a fully distributed system it relies on implementations of the so-called Data Distribution Service (DDS) (Object Management Group 2018), which comes with a service discovery mechanism for distributed systems.

The robotic framework Robot Construction Kit (Rock) (Joyeux, Schwendner, and Roehr 2014) uses a similarly designed network of components, but in contrast to ROS the robotic framework Rock offers a model-based development approach, direct control over the runtime state of communication nodes, and direct control over the communication network - ROS requires to change the internals of a node to adapt the communication network. A so-called oroGen component in Rock corresponds to a node in ROS. Each oroGen module is an Orocos RTT (Soetens 2012) based component. Orocos RTT has a well defined component model and each oroGen component is generated from a model specification which defines named and typed input and output ports, along with configuration properties and update frequencies. To establish a communication network with Rock, an additional level of supervision allows the direct specification of connections between components, i.e., how an output port of one component is con-

nected to another component's input port. Robot Construction Kit (Rock) offers the tool *syskit* for so-called supervision. The tool builds upon the component model and allows the offline design and validation of complex communication networks. The communication flow of activated networks will be correct by construction due to the model-based approach. The network design can use abstraction of component interfaces, so that network templates can be created with collection of port types as interfaces for later added oroGen components. The reference systems of the projects RIMRES and TransTerra described in Chapter 2.2 have been implemented with Rock and have embedded *syskit* for the supervision level (see (Roehr, Cordes, and F. Kirchner 2014)). Controlling the design of the dataflow network is a necessary precondition for dealing with reconfiguration and the robotics framework Rock is well suited to support reconfiguration.

Another approach which intends to wrap existing robotic frameworks is found in D-Rock (DFKI GmbH Robotics Innovation Center 2018). D-Rock provides a model-based software development and relies on a extensible hardware and software component database in order to facilitate the design and implementation of robot in an end-to-end fashion. D-Rock supports developers with an end to end workflow which allows to create the initial physical design of a robot and enables a user to map existing software components onto the existing hardware. Thereby, the general robotic development process can be performed with greater consistency and supporting the reuse of software on various target systems. D-Rock provides a higher-level model abstraction for the existing frameworks such as Rock and ROS.

5.1.3 Distributed Communication & Ad-Hoc Networks

The application of distributed systems is often initially motivated by the avoidance of a single point of failure. To maintain this feature in a multi-robot system firstly a fully distributed communication architecture is required. ROS and Rock, however, depend on the availability of a centralised component. ROS requires a ROS Master, and Rock a centralised CORBA naming service. To establish a fully distributed communication system a variety of high-level communication patterns has been applied in the context of multi-robot and multi-agent systems including broadcasting-based (Parker 2006), blackboard-based (BBN Technologies 2004), and cloud-based (Eich et al. 2014) approaches. Since these approaches are typically found at the application level of the ISO/Open System Interconnect (OSI) reference model, they share the same basic communication stack for the layers 1-4, including the well known protocol IP, ARP, and ICMP at layer 3, and TCP or UDP at layer 4 of the ISO/OSI reference model.

A distributed system with temporary changing communication nodes, however, requires a mobile ad-hoc network. Wireless mesh networks can account for the dynamics of such systems, so that any two systems can communicate with the help of intermediate agents, even if the two systems are not directly connected. One meshing solution is offered by the protocol B.A.T.M.A.N Advanced (Open-mesh 2012). Since the protocol operates on layer 2 of the ISO/OSI reference model, it can transparently add meshing support to any IP-based communication. The protocol has been successfully evaluated by Seither, König, and Hollick (2011) for an application in office environments and has been actively used and developed by the Freifunk community (Förderverein Freie Netzwerke e. V. 2018). The use of a wireless mesh network has been pre-evaluated in the project RIMRES, and established and actively applied for the robotic team in the project TransTerra. The IEEE standard 802.11s for mesh networks has been finalised in 2012, and requires the Hybrid Wireless Mesh Protocol (HWMP) as default

routing protocol; the standard currently enters commercial systems. While 802.11s has a limitation for using up to 32 nodes, B.A.T.M.A.N has been validated with 80 nodes (Lüssing 2013). Both protocols operate at layer 2 and come with a comparable performance as shown by Singh and Talasila (2015): while the B.A.T.M.A.N Advanced protocol comes with a higher data throughput for up to three hops, the IEEE standard 802.11s shows a better performance for transport of four and more hops.

In a dynamic network, agents have to be able to identify other agents or generally available services. A mechanism for service discovery is required (see (Karim Talal and Rachid 2013) for a comparison of available solutions). Distributed mechanisms for service-discovery can be implemented using so-called zeroconf solutions (IETF Zeroconf Working Group 2013) such as Avahi (Poettering, Lloyd, and Estienne 2012). Avahi is a commonly used on Linux-based systems to provide a way to dynamically discover services. Avahi announces the appearance of each service as well as its removal, and its event-based notification has served as basis for the development of a dynamic and distributed service discovery mechanism which has been integrated into Rock as part of this thesis.

5.2 A Distributed Communication Architecture

The implemented communication architecture for a modular and distributed multi-agent system builds upon a set of existing standards and open-source libraries. The FIPA Abstract Architecture and more specifically its Agent Communication Language specification serves as basis for a generic and flexible approach to inter-robot communication. Since it is not necessary to provide a full implementation of the FIPA Abstract Architecture to support distributed communication, the following sections describe the necessary subset only. This subset includes the message transport service, service directory service, the agent communication language and support for using interaction protocols. The implementation of services is publicly available as part of Rock (Roehr 2013a). Figure 5.2 depicts the high-level communication architecture, which builds upon a number of existing technologies as listed in Table 5.2 to create a robust communication system for inter-robot communication.

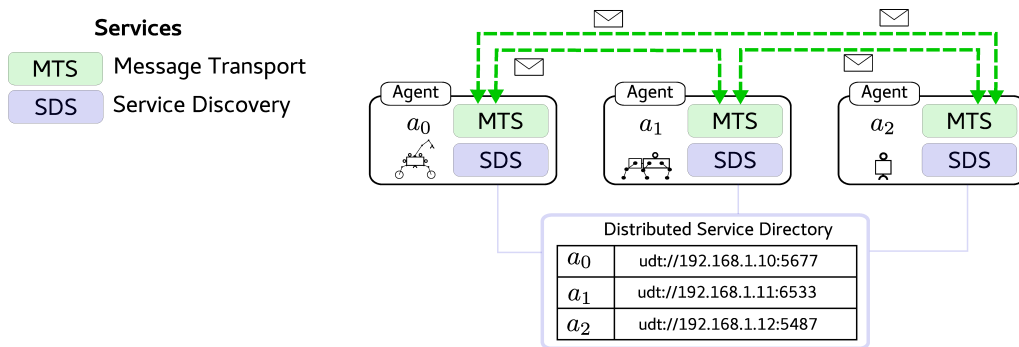


Figure 5.2: The FIPA-based messaging infrastructure for a multi-robot system (adapted from (Roehr and Herfert 2016)).

The FIPA Abstract Architecture (Foundation for Intelligent Physical Agents 2002a) defines yellow and white pages as core elements of an agent architecture. While white pages allow to identify the communication endpoints for services and agents, yellow pages provide a means

Table 5.2: *The essential network technologies involved in the communication layer, sorted according to the seven layers of the OSI model (ISO/IEC 1996).*

OSI Layer	Protocol Name	Usage description
Application	FIPA ACL, CORBA	inter-robot communication: FIPA-message based, intra-robot communication: CORBA-based
Presentation		
Session		
Transport	TCP, UDP, UDT	Transport of CORBA and FIPA messages
Network	IP, ARP, multicast DNS	Defining system addresses, forward and reverse name resolution for IP-based communication and multicast DNS for service discovery (using AVAHI)
Data Link	B.A.T.M.A.N., 802.11 b/g/n	Mobile ad-hoc networks, mesh-based communication
Physical	802.11 b/g/n	Wireless communication

to search for a particular service or agent based on a set of criteria. These services are referred to as Service Directory Service (SDS). The main functionality which is offered by this service is a dynamic registration and deregistration of services (or agents) and a query interface for associated agents. Although a SDS can be implemented as a centralised component, this would create a single point of failure and should therefore be avoided. The service discovery service has therefore been established as a distributed database, which is synchronised across all instances and which is locally available and accessible for each agent. The synchronisation is done via an event-based notification schema, so that dynamically appearing or disappearing agents are announced to other instances of the Distributed Service Directory Service (DSD). Since a non-nominal shutdown of an agent prevents a proper announcement, a keep-alive mechanism allows to monitor the availability of other agents. The DSD is an essential element to support the full distribution of a robotic system, and allows a dynamic extension of the multi-robot system, i.e., the DSD is the basis for the dynamic identification of operative agents.

The implementation of the DSD relies on Avahi (Poettering, Lloyd, and Estienne 2012), where the implementation of the DSD has been split into a C++-library to interface Avahi's functionality (Roehr and Makreshanski 2010) and a high-level interface for the actual DSD as part of the FIPA Service library (Roehr 2013a).

5.2.1 Message Transport

The message-transport service (MTS) is a core element of the FIPA infrastructure and it handles letters, i.e., FIPA messages which are wrapped into an envelope. A set of MTS establishes a high-level communication bus, which can transport arbitrary content. The envelope provides only the information to the MTS which is needed to take the correct routing decision for this letter, so that it can be transmitted to a given recipient. The use of an envelope allows to avoid a potentially costly processing of large messages or transport of encrypted messages. Since message as well as envelopes have different representations which are listed in Table 5.3, this can have a significant influence on the performance of the communication. This is shown in Section 5.3.

Table 5.3: *FIPA representations.*

Element	Representation types
message	bit-efficient, XML, string
envelope	bit-efficient, XML

The MTS either delivers a letter to its locally attached client agents, or redirects a letter to another MTS, which serves as communication endpoint for the recipient agents. The information about the communication endpoint is extracted from the DSD, which maintains a mapping between agents and communication endpoints. The protocol for the transport of letters between two agents is exchangeable, such that the communication endpoint represents a transport protocol specific access to an MTS. The DSD contains all active communication endpoints for a single MTS. To deliver a sender agent's message to another client, the client agent firstly wraps a message into an envelope and forwards the resulting letter to the local MTS. The MTS requires at least one given recipient. This recipient can also be specified by a regular expression which is matched against the entries in the DSD. Thereby, a simple mechanism for broadcasting and multicasting is introduced which is not defined in any standard. This feature can be exploited for a topic-based communication in a multi-agent network.

The implementation supports two types of transport protocols: the connection oriented Transmission Control Protocol (TCP) and the User Datagram Protocol (UDP)-based high-bandwidth protocol UDT (Gu and Grossman 2007). The standard UDP protocol is connection less and packets are limited to a maximum size of 65535 bytes. UDP is used for communication of data which has to be rather most recent, than complete. Although UDT uses UDP it still allows for a reliable transport of messages and these message can exceed the maximum UDP packet size.

Interaction Protocols & Error Reporting

Standard communication approaches in robotic frameworks use a simple forwarding of data. Stateful communication can only be achieved with a custom solution or add-ons to these frameworks. The use of interaction protocols as suggested by FIPA adds a general way to structure communication using the performative field of FIPA messages. An interaction protocol therefore describes a message flow using a state transition system $\Sigma = (S, P, R, \gamma)$, where $S = \{s_0, s_1, \dots\}$ is the set of state in a conversation, $P = \{accept-proposal, agree, \dots\}$ is the set of performatives, and $R = \{r_1, r_2, \dots\}$ is the set of conversation roles. When the protocol is matched with a real conversation, each role maps to an agent and the minimal protocol definition requires at least a sender and receiver label. Appendix H shows some details on the representation of interaction protocols. The use of interaction protocols allows to verify distributed communication online, e.g., such as agreement procedures. It can thereby increase the robustness of distributed control approaches. For that purpose a conversation monitor has been added (cf. (Roehr and Herfert 2016)) which validates the conformance to a requested interaction protocol. Each protocol's flow can be interpreted according the current agent's role in this conversation.

Errors in the communication path can be reported back to a sender, so that the framework establishes a non-nominal feedback channel for high-level communication similar to the FRM. Figure 5.3 shows the nominal message flow, which involves the following steps at the sender's

side: encoding of the message in one of the representation formats (XML, String or bit-efficient), wrapping of the encoded message into an envelope, encoding of the resulting letter in one of the representation formats (XML or bit-efficient) and handover to a MTS. The MTS decodes only the envelope to extract receiver information and to add its stamp, before trying to deliver it to the MTS which has been registered for the receiver. The transport between two MTS requires letter serialisation according to the representation format. At the receiver site, the letter is decoded, so that the encoded message can be extracted and decoded.

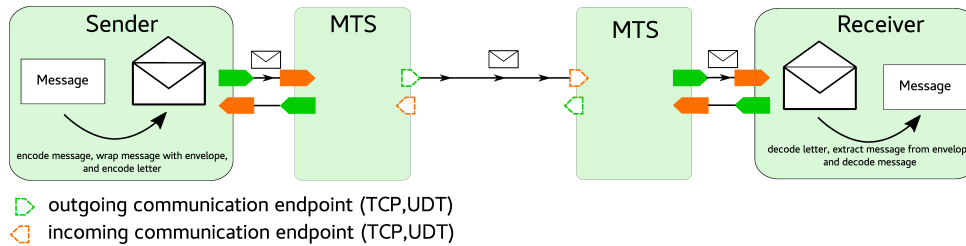


Figure 5.3: *The nominal message flow. Note that for each agent there might be multiple components which act as sender or receiver.*

Figure 5.4 illustrates the reporting of a failed delivery, i.e., when the intended receiver of a message is not available. Since the number available agents in a distributed system might change ad-hoc, an error reporting infrastructure is mandatory for a robust control approach. Corresponding interaction protocols can account for the dynamic removal of agents.

Since letters could start looping between two MTS, the FIPA standard prevents a looping of letters by marking each outgoing letter with an MTS specific stamp. Already stamped letters can be safely discarded by an MTS. Before a sender is notified about a failed delivery, all active transport protocols are tried. Only if all transports protocols have failed an error is reported back to the original sender of the message. When the originator has also become unavailable in the meantime, the error report is discarded by the MTS which detects the originator to be missing.

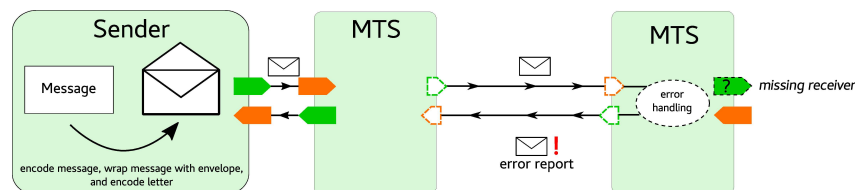


Figure 5.4: *Error reporting back to a message's sender is automatically triggered by an MTS, when the message delivery fails.*

The MTS implementation ideally connects all MTSs to a peer-2-peer communication network, but the communication range of individual agents might be limited so that a direct connection between some agents cannot be established. The implementation of a routing mechanism across multiple MTS is one feasible approach to create a mesh-based communication. The selected solution, however, does not enable mesh based communication at the application layer (layer 7), but on the data link layer (layer 2) by using the protocol B.A.T.M.A.N. Advanced (Open-mesh 2012). Since this protocol operates at layer 2 of the ISO/OSI network model it can be transparently added to any IP-based communication.

Channel-based Communication

The standard FIPA ACL accounts for topic-related discussions between multiple agents, and a particular topic can be identified via the message field *conversation id*. This approach allows agents to filter communication, and facilitates to correctly redirect or process communication. This conversation-based filtering is part of an internal processing chain, but the MTS implementation allows an additionally separation of communication into communication channel. Each MTS allows the registration of so-called local receivers. A local receiver typically resides on the same physical machine as the MTS. Each receiver can be viewed as particular communication processors within a single agent. One agent can have multiple communication processors which are all identifiable by name. The communication endpoint for each communication processor is defined by a named output port of the MTS. This output port is dynamically created upon association with an MTS. A corresponding service entry is added to the DSD. The naming of the output ports of the MTS should follow a policy: *<agent-name><suffix>*. This policy is suggested since the MTS allows for multicasting based on receiver name patterns, e.g., all messages to *'*-processor-0'* are forwarded to all agents which have an MTS port with a matching name. Hence, the communication system can be used to define an arbitrary number of communication channels and topic groups.

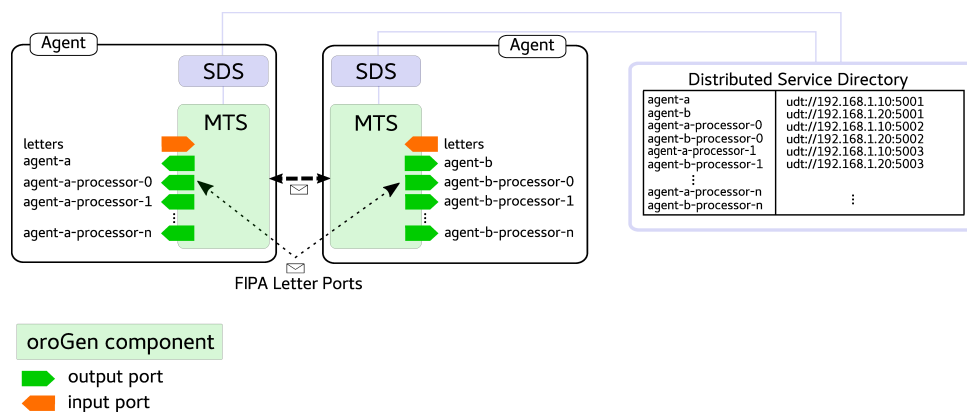


Figure 5.5: Channel-based communication with MTS.

5.2.2 Distributed Sensor Network

Most robotic systems have a set of sensors and the sensor data is usually processed locally, i.e., within the physical boundaries of each robot. However, the exchange of sensor and telemetry data between two and more robots can be required, e.g., to operate a multi-robot team as a distributed sensor network. The suggested distributed communication architecture uses a FIPA-based communication bus to establish inter-robot communication, but the architecture relies on a multiplexing/demultiplexing approach to generically support the exchange of data samples between multiple agents. Figure 5.6 depicts the essential components to support data exchange between two systems.

The *telemetry provider* component provides the main functionality to multiplex sensor data into a single container package, and demultiplexes a single container package so that data of particular sensors can be routed to the correct consumer. A container package can contain one

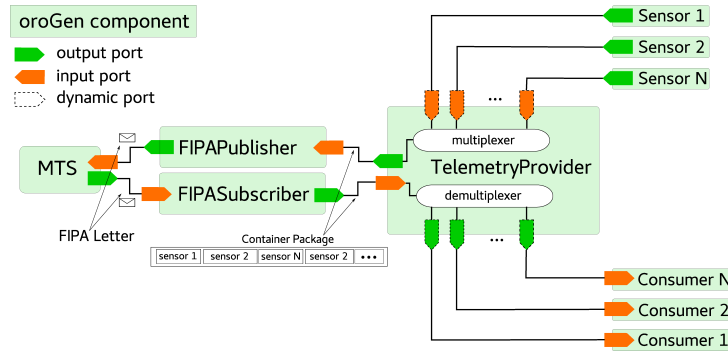


Figure 5.6: Component network structure to support a decentralised publish-subscribe based communication for sensor data inter-robot sensor-data/telemetry exchange.

or more stream samples from a set of sensors, such that also stream fragments can be transmitted via the high-level communication bus. The *FIPAPublisher* prepares a container package for transport across the FIPA-based communication bus, and therefore wraps it into a FIPA Letter. The set of receivers has to be predefined, but due to the multicast ability of the MTS a topic based communication system can be created which can account for the dynamic extension of the sensor network. A sensor data flow can be set-up dynamically, since the *telemetry provider* component supports the dynamic creation of input and output ports to maintain a generic communication approach. This feature allows to adapt to dynamically changing flow networks, which can result from physical reconfiguration and adding new hardware, or from temporary local changes of the setup to adapt to application requirements. The robotic framework Rock has been selected for implementation of the communication architecture and Rock allows an external control of the data flow, in contrast to ROS. Hence, the presented communication architecture maintains control over the full data flow, since the receivers of a message can be exactly specified if needed, and thereby achieves a generic approach to designing data-flow across multiple agents. Chapter 6 shows an application of this architecture for supporting distributed SLAM.

5.3 Evaluation

The distributed communication architecture implementation has to support all agents in a heterogeneous robotic system to be fully applicable. For that purpose the implementation has been validated on the PC-based robotic systems as well as on the ARM-based platform which has been used in all payload items of the reference systems. The performance evaluation has been performed on a PC with an Intel CORE i7 2.1 GHz, 12 GB RAM and a Gumstix with ARM Cortex-A7 CPU 720 MHz, 512 MB RAM; the evaluation is based on using software timers. All payload items and the base camp use a Gumstix Overo Fire with ARM Cortex-A7 (Gumstix 2015). This computing board can perform local sensor processing, but mainly supports the use of a camera in the bottom interfaces. This camera enables visually guided docking processes as described in Chapter 6.

The evaluation of the communication system is split into several parts. Firstly, an evaluation of the core library to perform the encoding and decoding of FIPA messages and letters. Secondly, an analysis of the communication properties of the inter-robot mesh communication which

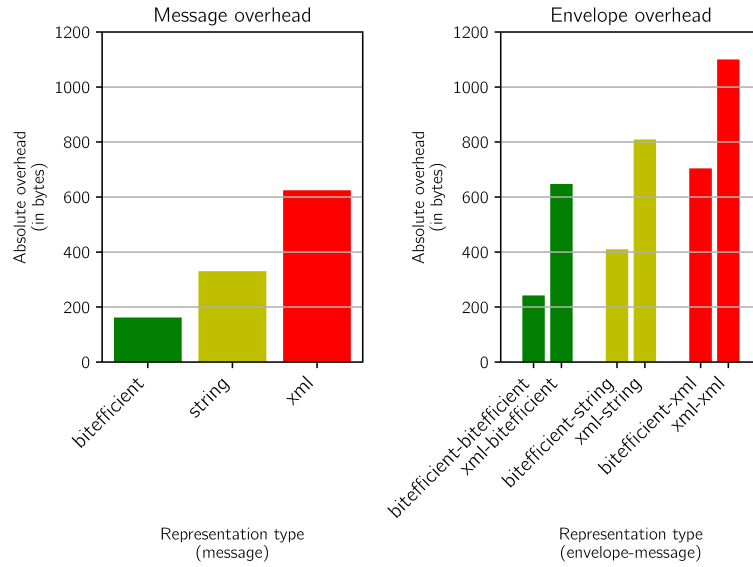


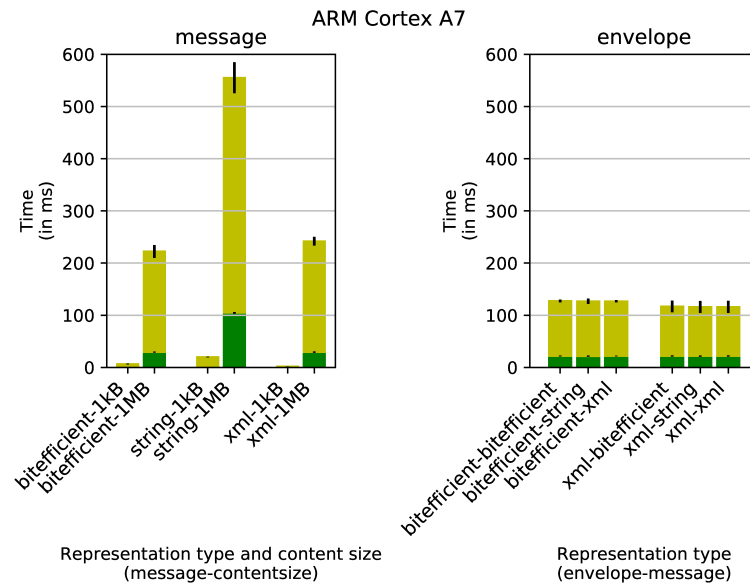
Figure 5.7: Net message and envelope overhead for a message with the content field containing 1 byte and 99 bytes in the remaining message fields as listed in Table 5.1 (adapted by permission from Springer Customer Service Centre GmbH: Springer Nature (Roehr and Herfert 2016) © Springer International Publishing Switzerland 2016).

is using the FIPA-based communication in combination with the mesh-protocol B.A.T.M.A.N. Thirdly, a practical analysis of the usage of conversation protocols for multi-robot coordination.

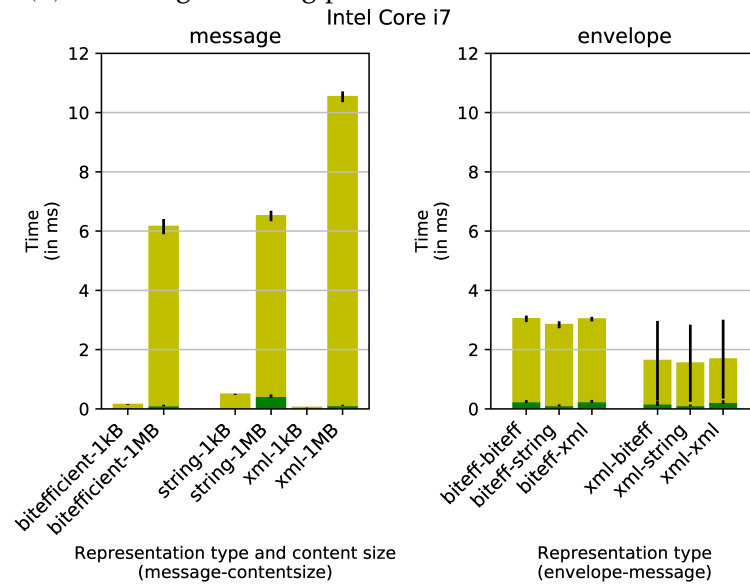
5.3.1 Message-based Communication

The FIPA standard defines three different representation types: XML, String and bit-efficient. The author's library offers the first publicly available implementation of the FIPA standard for bit-efficient encoding. The bit-efficient standard is of particular interest, since it can save communication bandwidth by using additional computing power for encoding and decoding. Therefore, it can serve bandwidth limited space applications.

Protocol Overhead Each representation type comes with a particular overhead. Figure 5.7 highlights the differences between the representation types for the individual message and the combination of message and envelope encoding. The message overhead o_m is computed as $o_m = \frac{m-p}{m}$, where m is the size of the encoded message, and p is the sum of all message field sizes. The envelope overhead o_e is computed as $o_e = \frac{e-m}{e}$, where m is the size of the encoded message, and e is the size of the encoded envelope. This evaluation shows that bit-efficient message encoding is best suited for environments with limited bandwidth. It comes with a slight performance drawback for small messages compared to the XML representation, but performs better for messages with large content. The string representation shows the worst performance for large messages on the Gumstix, while it remains ahead on XML on the PC. Figure 5.8 illustrates the performance for different combinations of envelope and message representations and it shows, that message decoding is the costly part of the process. Benchmarking the envelope encoding shows, however, that the internally carried messages are left untouched. Especially



(a) Encoding decoding performance on ARM Cortex A7.



(b) Encoding decoding performance on Intel i7.

Figure 5.8: Encoding and decoding performances. The colour coding indicates the performance of encoding in dark green and decoding in light green (adapted from (Roehr and Herfert 2016)).

the performance on the ARM-based system illustrates this effect, which can be exploited when messages pass through an MTS, without the need to process the full message content. Bit-efficient encoding for envelopes comes with a performance drawback, but it also comes with a significantly smaller standard deviation.

5.3.2 Mesh-based Communication

A wireless mesh network offers inherently more robustness compared to a network with a central access point. Additionally, the operational range of the multi-robot system increases, since routing across multiple communication nodes is possible. The team of agents can disperse further than the communication range of a single access point while maintaining communication between all agents. But even if communication is disrupted agents can still locally co-operate, which is a significant advantage especially in hazardous and unknown environments. Hence, a mesh-based communication can benefit most multi-robot system operations. Meshing solutions which operate at Layer 2 of the OSI reference model transparently support the implemented communication architecture, but can still introduce an overhead, e.g., in terms of protocol overhead. To validate the general approach and to illustrate the general benefit of applying a mesh network, it has been compared against a communication setup using a central access points. The wireless communication network in a multi-robot system can be established by the addition of specially prepared access points. For the project TransTerrA OM2P access points (Open Mesh Inc. 2018) have been used in combination with the open router operating systems openWRT (OpenWRT/LEDE Community 2018). All OM2P access points have been flashed with a system image of openWRT 15.05 (Chaos Calmer) which has been customised to support the usage of Avahi for service discovery (by enabling so-called reflector mode) and the routing protocol B.A.T.M.A.N. Advanced (latest tested version 2017.4). IPs are statically assigned to each robot, and the wireless network operates with 802.11n at a maximum channel width of 40 MHz.

The performance of the distributed communication architecture has been evaluated to characterise different setups for a multi-robot communication network; the benchmarking tool part of the FIPA Service implementation for Rock (Roehr 2013b). The setup relies on a single producer of FIPA letters and client agents which echo letters back to the producer. The total time of the dialogue is measured based on timestamped letters and for varying message sizes. The benchmark is split into a prepare and a measurement stage. For the prepare stage, the sender hands a single letter which is addressed to "echo-.*" to its local MTS and waits for responses to identify all echo agents on the network - a timeout of 30 s applies. In the measurement phase, the sender sends a letter to the multicast group and awaits responses from all identified echo agents. The process is repeated for an increasing message content size. Depending on a number of epochs the overall process is also repeated - apart from the echo agent discovery. The experiment validates the implemented distributed communication in general including the service discovery approach using a typical application scenario. Multiple agents are taking part in a conversation, and also respond to a query in parallel. The access point in a central AP-based setup is not only a single point of failure but can also be a bottleneck. All transferred data has to pass the central access point; the access point has to receive and resend the data to its destined client. This approach reduces the available bandwidth compared to a mesh-network, where direct communication is possible between two agents. A mesh network suffers from the same effect as soon as the hop distance between the communication nodes exceeds one, i.e., when an additional node is required for relaying. Figure 5.10 depicts the results for a total of 20 epochs

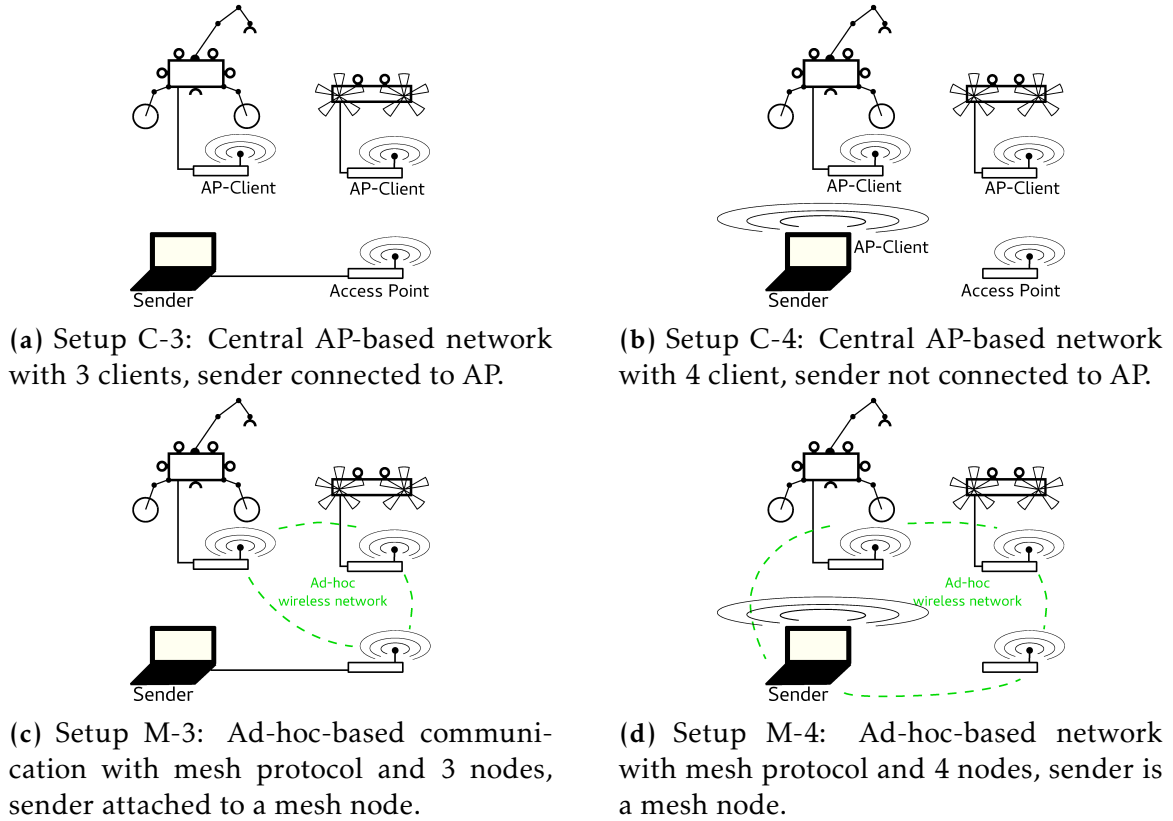


Figure 5.9: Wireless communication network setups which have been used for comparison of a centralised vs. a mesh-based communication setup.

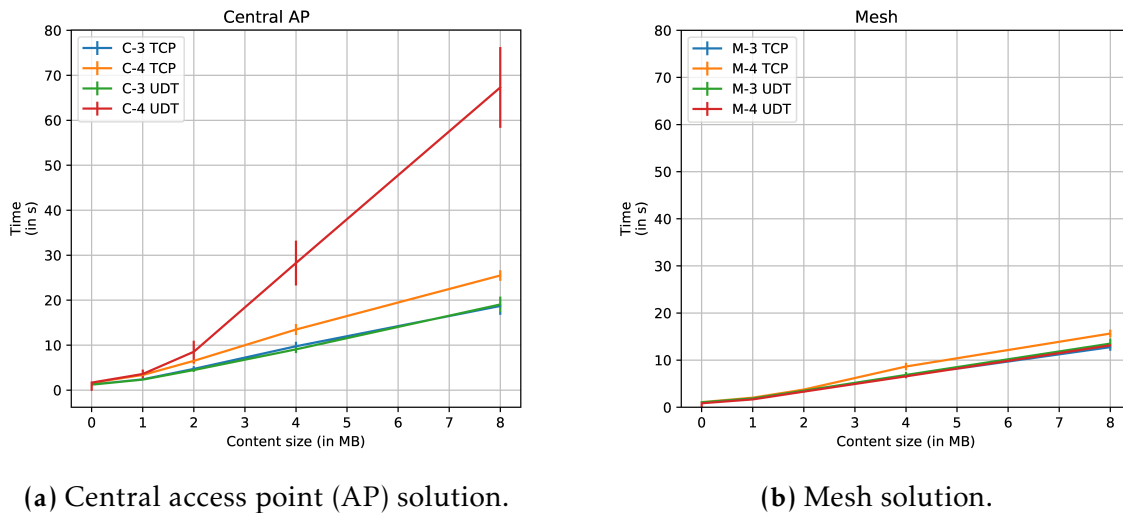


Figure 5.10: Comparison between a central AP-based solution and a mesh-based solution implemented with OM2P routers using the operation system openWRT with three communicating agents.

Syntax	Description
ACTION <NAME-OR-ID>	Control action by name
[EXEC ABORT STOP PAUSE RESUME STATUS DESCRIBE]	Mandatory control keyword
[[<KEY-0><VALUE-0>]... [<KEY-N><VALUE-N>]]	Optional list of named arguments
PROBE	Request an empty reply
INTERACTION REQUEST	Request interaction
[FROM <SUBSYSTEM>]	Optional identification of the requesting subsystem
[MSG <CONTEXT>]	Context or reason of the request

Table 5.4: *Extract of a content language.*

and messages with a content size of up to 8 MB. A clear drawback can be seen when using a central AP-based solution in combination with UDP-based Data Transfer Protocol (UDT) (C-4 UDT). Only client are attached to the access point they can achieve a comparable performance to a mesh-based solution (C-3). The evaluation of the mesh-based communication is detailed in Figure 5.10b. The experiments show an advantage to use UDT compared to the use of TCP (M-4 UDT vs. M-4 TCP). However, all experiments M-4 and M-3 are expected to show a similar performance as result of the direct communication approach. The delta to M-4 TCP is likely to be caused by a difference in the network stack; M-3 relies on the wireless network stack of the OM2P routers, while M-4 involves the laptop's wireless network stack.

5.3.3 Multi-Robot Coordination

Reconfigurable multi-robot systems are highly flexible, but due to their distributed nature require a more generic control approach. The FIPA communication standards have been also selected for implementation to due their general approach for conversation control - five message fields are reserved for the control of conversations in combination with interaction protocols and content languages. The actual execution of reconfiguration is part of the fourth stage of Dunin-Keplicz and Verbrugge's approach to teamwork (B. M. Dunin-Keplicz and Rineke Verbrugge 2010) and a general reconfiguration execution recipe can assume an already identified target reconfiguration state and the participating resources. But all required or affected resources or rather agents have to be reserved to performed the reconfiguration process. Additionally, a MOISE+ like approach to reconfiguration is required to support a centralised control approach, i.e., using a local reconfiguration manager. This manager has to lock resources, control the performance of the reconfiguration and finally release again all resources, i.e., this approach relies on a locally centralised control schema for which agents need to be able to exchange control commands and provide feedback. Hence a distributed locking approach is required.

Inter-robot communication can be based on content languages - Roehr, Cordes, and F. Kirchner (2014) suggest one to control robot actions. The definition of a human-readable content language enables robot-to-robot interaction, while simplifying human-to-robot interaction and debugging. The developed content language defines action commands following the syntax shown in Figure 5.4. The initial response is either a message with an action identifier for additional polling of the status, or a message with content NO_SUCCESS or SUCCESS. Controlling an action with the content language as shown in Table 5.4 assumes an action state machine, and the docking experiments performed in the context of the multi-robot team of RIMRES (Roehr,

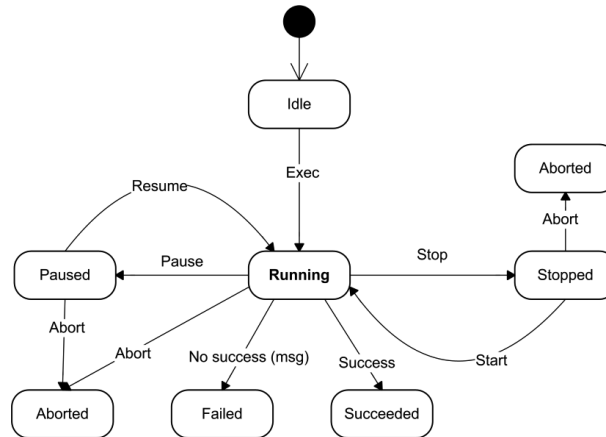


Figure 5.11: State machine for performing an action on a robot

Cordes, and F. Kirchner 2014) are based on the state machine as depicted in Figure 5.11. The content language allows to control and monitor the action from one robot by another, and has been the basis for performing reconfiguration of a distributed multi-robot team (see Chapter 6).

To apply distributed locking with a reconfigurable multi-robot system two algorithms have been implemented and evaluated (see (Roehr and Herfert 2016)): Ricart-Agrawala's (Ricart and Agrawala 1981) non-token-based and Suzuki-Kasami's (Suzuki and Kasami 1985) token-based algorithm. Both algorithms had to be adapted in order to deal with a dynamically changing set of agents or agent-failure. The distributed locking approach consists of the following stages: (1) probing, (2) discovery of the owner of a resource, and (3) performing the actual locking. Corresponding interaction protocols and content languages have been defined to facilitate the development and enforce the compliance with the algorithm at runtime. Figure 5.12 illustrates the protocol for the (extended) Ricart-Agrawala's algorithm.

5.3.4 Standardisation for Interoperability & Maintenance

Reconfigurable multi-robot systems offer their full potential only if they come with a control architecture which supports extension and interoperation. For this reason the application of the FIPA standard has been fostered for the inter-robot communication, and the interconnection of the FIPA communication architecture to the JAVA-based reference architecture JADE (Bellifemine, Poggi, and Rimassa 1999) has been achieved as shown in (Roehr and Herfert 2016). The interoperability of these two architectures, however, concerns mainly the explicit exchange of FIPA letters. So while it allows for communication across both architectures it requires a subsequent agreement on content languages and interaction protocols to achieve interoperability of agents. A minimum consensus exists through the use of FIPA's *not-understood* performative, which is the standard response if a message cannot be interpreted. The intra-robot communication has also been subject to standardisation and it is based on the model-based data flow design of the framework Rock (cf. (Joyeux, Schwendner, and Roehr 2014) and (Schwendner, Roehr, et al. 2014)).

Standardisation is not only essential for interoperation and the design of the control architecture, but also for the general maintenance of (multi-)robot systems. The management cost of multi-robot systems scales linearly with the number of systems, and cost can be reduced by

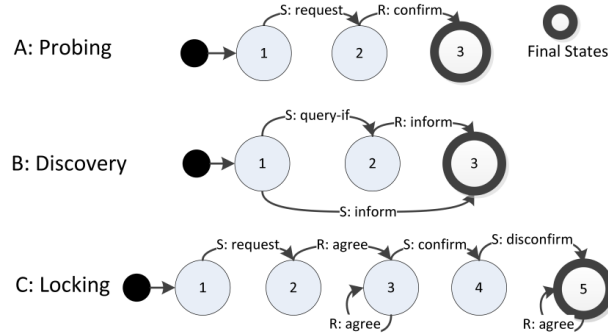


Figure 5.12: Three separate interaction protocols that are part of the (extended) Ricart-Agrawala’s algorithm; A: probing of inter-dependant agents, B: discovery of the resource owner (owner information is sent as broadcast), C: the message flow when performing the actual locking algorithm. Failure handling is embedded into default transitions (which are not depicted) and S denotes the conversation initiator’s role and R the recipient’s role (reprinted by permission from Springer Customer Service Centre GmbH: Springer Nature (Roehr and Herfert 2016) © Springer International Publishing Switzerland 2016).

maximising the reuse software. The reference system as introduced in Chapter 2 is a heterogeneous multi-robot team which requires support for amd64 and ARM-based systems. Particularly the use of ARM-based system requires a custom workflow to support the deployment of the software architecture. To create and unify the workflow for the system architectures in use, a packaging solution has been developed based on Debian’s binary packaging format. The resulting toolkit automated packaging for autoproj (apaka) (Roehr and Willenbrock 2018; Roehr, Willenbrock, and Joyeux 2018) enables the automated generation of binary Debian packages to ease maintenance of reconfigurable multi-robot system. Package releases can be created and transparently installed and used in Debian-based operating systems. Establishing a shared, standardised software basis is important for multi-robot systems and the packaging system allows the synchronisation of a software state across multiple systems. It facilitates the (re)distribution of a software stack and allows to create a verifiable, reproducible software installation. As a result it contributes to the robustness of a multi-robot system. Apaka tackles a software maintenance problem not limited to robotics and enables framework maintainers to create and manage software releases by reusing existing know-how from the Debian community. This packaging solution has been successfully applied in the context of robotic field testing by Sonsalla, Cordes, L. Christensen, Roehr, et al. (2017b) and complements the standard workflow in Rock (Roehr, Willenbrock, and Joyeux 2018) and D-Rock (DFKI GmbH Robotics Innovation Center 2018).

5.4 Control Architecture

Compatible hardware interfaces enable reconfiguration for a team of robots in the first place. A full exploitation of reconfigurable multi-robot systems, however, additionally requires software support. Firstly, general control algorithms have to allow a physical reconfiguration by joining two interfaces; this has to include the general docking process of two and more robots. Secondly, internal software structures have to be adapted to account for the newly defined scope of the robot, i.e., link the functional network of formerly two agents, which now form

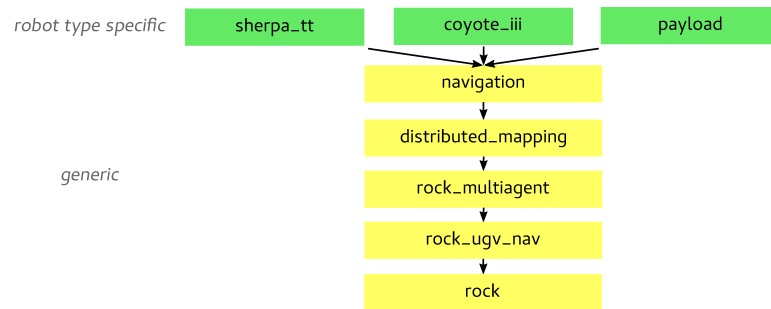


Figure 5.13: *Interdependencies of bundles: hierarchical function design is achieved by modularising functionalities. Component network templates and generically implemented actions are collected in so-called bundles. This approach leads to standardisation and facilitates the reuse of component network templates.*

one. The context of planetary space exploration requires some default functionality for autonomous operation: activation and use of a distributed communication system as outlined in Section 5.2 and the provision of simultaneous localisation and mapping as basis for autonomous navigation and exploration abilities. The actual performance of the control architecture is illustrated in Chapter 6. This section details elements which are particularly relevant for achieving reconfigurability and establishing control loops involving multiple robots.

5.4.1 Hierarchical Model-driven Design

The organisation model outlined in Chapter 3 introduces the usage of agent types as one means to deal with the combinatorial challenge. The consideration of agent types is also fundamental for the development of the software for the multi-robot team since it allows to maximise reuse and minimise maintenance. A team of robots might be heterogeneous, but robots might still share the same code basis. To facilitate the reuse of component networks, e.g., such as the components networks describing the communication system and autonomous navigation approaches SLAM, Rock’s hierarchical design approach is adopted. The general functional decomposition is based on the design of intra-robot component networks; the term component refers here to the use of Orocos RTT modules. Each component network represents a user-modelled dataflow which provides a desired functionality and can be designed as general template to enable a particular functionality. While each component in the network can be enabled, disabled or reconfigured, so can the complete component network be dynamically enabled and disabled. Therefore, each component network is characterised by the overall data flow as well as the individual parametrisation of its components. Hence, any reconfiguration or re-parametrisation of a single component can be considered a new component network.

The design of a monolithic and static dataflow setup, as it is often the case for implementations with ROS, is avoided and the use of component network templates increases the flexibility to react to hardware changes and perform software reconfiguration. The set of networks is collected in so-called bundles (see also (Joyeux and Rehrmann 2012)). A robot-type agnostic set of bundles, e.g., as depicted in Figure 5.13, guarantees a high degree of reusability.

Table 5.5: *Operation modi for reconfiguration.*

Operation mode	Description
cooperative	master and slaves, all agents active and communicating
uncooperative	master and slaves, slaves inactive and not communicating
continuous	Maintain power connection to agent b during a coalition structure change: $\{\{a\}, \{b, c\}\} \rightarrow \{\{a, b, c\}\} \rightarrow \{\{a, b\}, \{c\}\}$
disruptive	Cut power and data connection to agent b during a coalition structure change: $\{\{a\}, \{b, c\}\} \rightarrow \{\{a\}, \{b\}, \{c\}\} \rightarrow \{\{a, b\}, \{c\}\}$

5.4.2 Controlling Reconfiguration

Physical reconfiguration is a transition between two coalition structures and it requires a sequence of actions to join atomic agents and split composite agents. Different operation modi as shown in Table 5.5 have to be considered for a reconfiguration, where the desired operation modus combines cooperation and continuity. But a cooperative approach requires active support of the participating agents to establish a coalition, which might not always be available. The continuous availability of power for devices might either be a safety or an essential operational requirement, since some agents require an external power source. Due to failing components or no available power a continuous operation of an agent might not be achieved during all transitions.

Uncooperative reconfiguration

A (mainly) uncooperative reconfiguration process is encountered when an immobile payload has to be attached to a manipulating agent. For an uncooperative reconfiguration the roles of an active master and one or more passive slave agents can be identified. A slave agent can behave uncooperatively if it is an autonomous agent with another agenda, or it is unable to support reconfiguration. The former situation might also arise in a fully cooperative team settings, when the slave agent has not been informed about the reconfiguration. The latter can be due to permanent or temporally missing capabilities of the slave agent. The master has to attach one of its interfaces to a compatible interface on the slave agent in order to physically join both systems. Hence, the master agent requires sufficient Degrees of Freedom (DoFs) in order to perform an uncooperative reconfiguration. The general *uncooperative join* can be separated into the steps listed in Table 5.6, when the slave agent at least remains stationary.

To initiate an uncooperative reconfiguration the master has to detect the slave and the slave's interface to which should be docked. If this interface cannot be accessed directly, additional actions have to be considered in order to make it accessible, e.g., by changing the pose of the slave or removing any obstructing structures. The final docking step joins the interfaces and establishes the mechanical link, as well as the power and data link if possible. When the slave is a single atomic agent the reconfiguration ends with the uncooperative join. An optionally, subsequent split might be required when the slave is a composite agent. This split can be either cooperative or uncooperative. A *cooperative split* can be performed, when the power and data connection between both, now connected agents can be successfully established. The split can be triggered by the separating slave agent, e.g., commanding to open a female interface.

Table 5.6: Action sequence for an uncooperative reconfiguration.

#	Action description
1	master detects pose of slave
2	master approaches slave
3	master detects pose of slave's target docking interface
4	master actively repositions slave (optional)
5	master docks an interface to the slave's target docking interface

An *uncooperative split* has to be performed when the power and data connection cannot be established or can only be partially used, e.g., a redundant rescue system that is part of the reference systems' EMI design can still be used to power the bottom interface. The rescue system is a hardware feature (available for the EMI in TransTerra), so that an uncooperative split might only be optionally available.

Uncooperative reconfiguration is considered for the reference systems by the pickup of a payload through the master robot SherpaTT. The target interface is the male, top interface of a payload. SherpaTT's manipulator is used for pickup of payloads. Male interfaces are equipped with visual markers to facilitate the detection and to support a visual servoing process, which allows to attach the master robot's manipulator. General, a payload is assumed to reside within the manipulator's workspace. Ideally it is also within the field of view of the camera when the visual servoing process starts; otherwise an object search has to be initiated first. The visual servoing process guides the manipulator to a predefined reference position which is relative to the target interface. The docking can be completed with a blind docking action from this known relative pose. Blind docking as last step is required, since the field of view and the focus of a camera is limited. Chapter 6 provides further details on the setup and performance of the visual servoing process.

Cooperative reconfiguration

Cooperative reconfiguration can be performed with two active agents. Depending upon the DoF of both agents, one or even both agents can actively contribute to the approach. The use of a shared scene script for the cooperative reconfiguration as provided in Table 5.7 can avoid the need for long negotiation phases. Such a script is similar to the uncooperative reconfiguration. It involves the assignment of a master and a slave role to manage the process. Step 4 in this scene script accounts for some flexibility to allow the slave to command the master for repositioning. Firstly, agents might differ in their ability to reposition with different accuracy and precision due to their DoFs, e.g., we used the legged system CREX for the docking approach in (Roehr, Cordes, and F. Kirchner 2014) due to a better pose control. Secondly, agents might differ in their sensing abilities either due to missing sensors or due to a limited field of view. Step 5 of a cooperative reconfiguration might involve uncooperative reconfiguration as substeps, e.g., pick up or rather extraction of payloads from an agent coalition.

Cooperative reconfiguration takes advantage of the distributed communication architecture, since agents have to communicate to command or take orders. The control approach which has been evaluated with the reference systems assumes an already performed consensus of the master and slave role in a cooperative reconfiguration.

Table 5.7: *Action sequence for a cooperative join.*

#	Action description
1	master detects pose of slave
2	master approaches slave or directs slave to approach master
3	master detects pose of slave's target docking interface
4	master/slave repositions or directs slave/master to reposition
5	master docks an interface to the slave's target docking interface

Handling of reconfiguration effects

Chapter 6 evaluates a reconfiguration of the reference systems (see also Figure 6.19). The manipulator of the robot SherpaTT features a camera to use a visually guided docking process to attach to payloads. The manipulator's camera is blocked, however, as soon as the manipulator attaches to a payload. Therefore, all payload modules are equipped with an additional camera in the bottom interface, so that visual guidance can still be used to dock the payload module to other interfaces.

To continue docking with an already attached module, two agents have to establish a data and a power connection. These connections are ideally created transparently as soon as the interfaces are brought together (and locked) so that no higher-level control layer has to become active; this feature is part of the EMI design by Wenzel, Cordes, and F. Kirchner (2015). Additionally, software reconfiguration has to be triggered. For a continued support of visual docking, a new component network has to be activated which embeds the camera of the newly attached device. Therefore, an identification of the connected device is required and the set of configuration parameters, including camera calibration parameters and dimensions of the module, have to be known in order to prepare the new component network. A significant part of the data flow network which represents the functionality to perform visual servoing remains unchanged. Only the image provider and camera related configuration parameters change. The static part of the data flow network can therefore be modelled as template. The template is fully parametrised and instantiated as soon as the attached camera is known. To pick a payload up, first the bottommost camera attached to the end effector is identified. Subsequently associated configuration parameters for the camera are looked up from a database. Finally, the data flow network with the camera related parametrisation is activated.

For the selected scenario the camera parameters have been previously shared with the master, here the robot SherpaTT, which attaches to the payload. A fully generalised implementation requires to request this information from the attached agent after the power and data link have been established.

Error handling Reconfiguration comes with a risk of failure and introduces additional points of failures. For instance, attaching the manipulator to a payload was not free from failure. Firstly, reconfiguration failed during the mechanical docking process. Secondly, spontaneous outage and loss of the data link occurred after the manipulator attached to the payload. The error handling strategy involved to power down and power up the attached payload to recover from this error; this could be done by using the power management system which connects all EMIs. Hence, the following local failure handling routine become part of the reconfiguration process:

- (a) poll for the availability of the new agent and wait for a maximum of 60 s,
- (b) restart (power off/on) the attached module when no agent can be contacted.

This special error handling routine was sufficient to deal with the observed faults.

Coalition structure changes represent transitions between two stable organisation states. Such change is a point of failure in any reconfigurable multi-robot system. While failures such as the one described above can be completely avoided by quality assurance processes, a need for general failure handling strategies remains. Apart from a set of local, domain specific failure handling routines this thesis has, however, focused on the implementation of nominal capabilities needed for operation.

5.5 Discussion

The operation of a reconfigurable multi-robot system requires a general control and communication architecture to allow an exploitation of flexibility. This thesis advocates a FIPA-based inter-robot communication approach for a use in multi-robot systems. The developed implementation of the FIPA message standards (Roehr 2010) fully supports bit-efficient encoding and is the first publicly available implementation of FIPA's final standard for the bit-efficient message encoding. The FIPA Services Library (Roehr 2013a) provides the core functionality for message transport and service discovery services. It can be used to extend existing robotic frameworks; a reference implementation is provided for Rock.

A generic communication standard is seen as cornerstone for reconfigurable systems, so that an incremental and dynamic evolution of robotic teams becomes reality. However, the semi-formality of FIPA has been the subject of criticism (Weiss 2009). Nevertheless, its general practicality has been acknowledged. The gained experience with reconfigurable multi-robot systems confirms this approach. Especially establishing sound multi-robot interaction protocols is an important feature when operating larger robot teams. The open access to the implemented distributed architecture permits the robotics community to exploit these benefits.

The application of inter-robot communication channels is foreseen for low-frequency data and has been applied to low-frequency inter-robot control loops (see Chapter 6 and (Roehr, Cordes, and F. Kirchner 2014)). The inter-robot communication approach can be compared to a human-like information exchange which is flexible and generic. However, the infrastructure is also prepared to support higher-bandwidth and subsystem communication using generically implemented (de)multiplexing components (see also the experiments in Chapter 6).

The developed infrastructure introduces an overhead to achieve abstraction and standardisation. Interaction scenarios can be solved at lower communication cost or with better performance by creating special message types and dedicated software components. The possibly better runtime characteristics of an application specific solution has to be traded against the generic and reusable communication approach which the developed infrastructure provides.

The implementation and application of the communication infrastructure as well as the development with Rock show the benefit of modular architectural design templates, standardised interfaces, encapsulation and the ability to easily replace and combine core functionalities. This likewise reflects desirable design criteria for reconfigurable hardware. Generally, the design of robot-type agnostic software component networks and standardised communication is

essential to establish an extensible operational infrastructure for a reconfigurable multi-robot system. Hence, the implemented control architecture relies on the design of component network templates - partially instantiated at higher levels of abstraction, and fully instantiated in connection with a particular robot type. This approach permits to establish network design patterns that can be reused for the team of robots.

This chapter describes the technological basis which is evaluated in the context of robotic missions in Chapter 6. The presented approach focuses on nominal operations, but illustrates local failure handling strategies. It points, however, to the particular importance of effective and generic failure handling strategies for reconfigurable multi-robot systems. Therefore, the distributed communication architecture presented here can be the basis for further standardisation efforts which improve the scalability, interoperability and failure handling capability of a multi-robot architecture.

5.6 Summary

This chapter presents the core elements of an operational infrastructure for a reconfigurable multi-robot system. The infrastructure is the result of an iterative development process for increasingly sophisticated reconfigurable multi-robot teams. The operational infrastructure is composed of a distributed communication architecture which is designed to satisfy special need of reconfigurable multi-robot systems. The architecture is, however, generic and can be applied to any multi-robot or multi-agent system which has to maintain its distributiveness with a transparent and decentralised communication approach.

The distributed communication system is based on an existing set of standards, thereby achieving interoperability and extensibility. A message transport service and a distributed service discovery service are the essential components to form the distributed communication architecture and they establish high-level, inter-robot communication. Due to a modular architecture design, each component can be substituted with an improved service implementation. The components enable a channel-based inter-robot communication. This approach allows to maintain conversation control across multiple agents and can therefore be used for a model-based data flow design in a distributed, yet, dynamic multi-robot setup.

The operational infrastructure is a prerequisite for cooperative and uncooperative reconfiguration in a real reconfigurable multi-robot system. This chapter illustrates control elements and challenges for operating a reconfigurable multi-robot system. Furthermore, approaches to facilitate the maintenance of reconfigurable multi-robot systems have been identified and implemented. The developed maintenance approaches reduce required manpower and increase efficiency. More importantly though, they allow maintainers to achieve a consistent setup of software components across the members of a multi-robot system.

The general development effort towards an operational infrastructure shows, that flexible, yet still verifiable communication and an adaptive control approach are fundamental requirements to exploit the flexibility of reconfigurable multi-robot systems.

6

Field Testing

Disclaimer *This chapter introduces and expands on works which have been published or described in parts in (Brinkmann, Cordes, et al. 2018; Brinkmann, Roehr, et al. 2018; Sonsalla, Cordes, L. Christensen, Roehr, et al. 2017b).*

This chapter evaluates the performance of an autonomous operation approach with a real reconfigurable multi-robot system. The previous chapters describe the main design elements and considerations to achieve automated planning for a reconfigurable multi-robot system. Hence, the approach assumes that the high-level mission planning problem is solved and a particular action plan for execution exists. The level of abstraction, however, of this action plan is relatively high. The definition of such actions, for instance, assumes a mobile robot to be able to autonomously move to a target destination. In the case of a reconfigurable system, it even assumes the ability to join two robotic systems. Looking at action at this level of abstraction, hides many details. At a lower level, such the implementation of robot actions comes with significant challenges. An encapsulation of an action implementation, however, has to ensure the validity of the abstraction levels.

Real robots have been evaluated in two major experimental setups: Firstly, in an outdoor environment with a focus on achieving the semi-autonomous operation of a multi-robot team. Secondly, in a controlled indoor environment with a focus on the fully-autonomous use of coalition structure changes. This practical evaluation of reconfigurable multi-robot systems validated the general operational infrastructure and the autonomous execution of uncooperative and cooperative reconfiguration approaches.

Section 6.1 describes the performance of a multi-robot mission during a test campaign in Utah, USA. The focus is on the application of the operational infrastructure to establish robot-to-robot communication and satellite-based remote control from Bremen, Germany. Furthermore,

subsystem validation with respect to uncooperative docking procedures and the development of extension modules are discussed in the context of the test campaign in Utah. Section 6.2 describes the incremental implementation and evaluation of a sample mission involving fully autonomous reconfiguration. The chapter concludes with a discussion in Section 6.3 and a summary in Section 6.4.

6.1 Field tests in Utah

The two mobile robots SherpaTT and Coyote III have been deployed in a Mars-like environment close to the Mars Desert Research Station in Utah, USA in November 2016. Figure 6.1 shows the testing area from different perspectives. In the test area an overall four week long test campaign was performed and it completed with an analogue simulation of a multi-robot space mission after performing the validation of subsystem functionalities of each robot.

6.1.1 Analogue Multi-Robot Mission

The analogue multi-robot mission was performed on 16 November 2016 and it used a semi-autonomous control approach with the mission control centre being located in Bremen, Germany. The motivation for performing the mission was twofold. Firstly, demonstrating the feasibility of the remote control approach of a multi-robot space mission. Secondly, evaluating the infield performance of the multi-robot team consisting of the robot SherpaTT and Coyote III. The operation infrastructure presented in Chapter 5 has been used for the operation of the robots and in parts for the inter-site communication. The following sections present the experimental setup and an evaluation which focuses on communication characteristics.

Operations

Figure 6.2 depicts the setup for operations: the local control centre in Utah, USA acted as communication proxy between the deployed robots and the mission control centre. Human operators in the local control centre prepared the robots for the start of the mission. During the mission the local operators were not involved in commanding the robots. Each robot received target waypoints from the mission control centre in Bremen, through the local control centre. Target waypoints were identified by the operators in the mission control centre based on environment maps generated from sensor data of the robots. Planthaber, Mallwitz, and E. A. Kirchner (2018) setup the communication between Utah and Bremen based on a satellite link. To establish this communication link, the core elements of the distributed communication infrastructure described in Chapter 5 have been used as basis for the data transport. Figure 6.3 illustrates the data processing chain: the data transfer was based on the UDT protocol. Since a large delay had to be accounted for establishing a connection, the internally set default timeout for establishing a UDT connection had to be raised to 20 s. The local control station and the robots used a wireless network IEEE 802.11n and a central access point. Note that a central access point based communication infrastructure was selected for the analogues mission (instead of a mesh-based setup): the satellite modem required a very particular network setup which inflicted with using a mesh-based communication. Post analysis identified a configuration error in the mesh setup, which could lead to looping network packages.

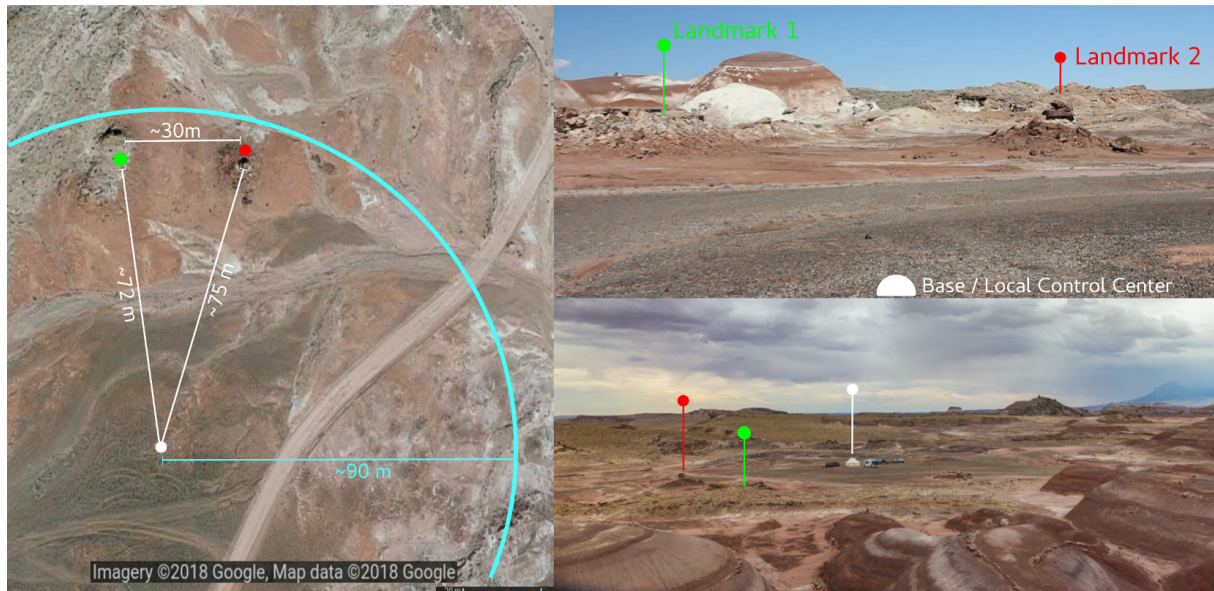


Figure 6.1: Left hand side: general overview over the testing site in Utah with the respective main landmarks and approximate distances (Source: manually annotated Imagery ©2018 Google, Map data ©2018 Google), right hand side: view onto the site from additional perspectives.

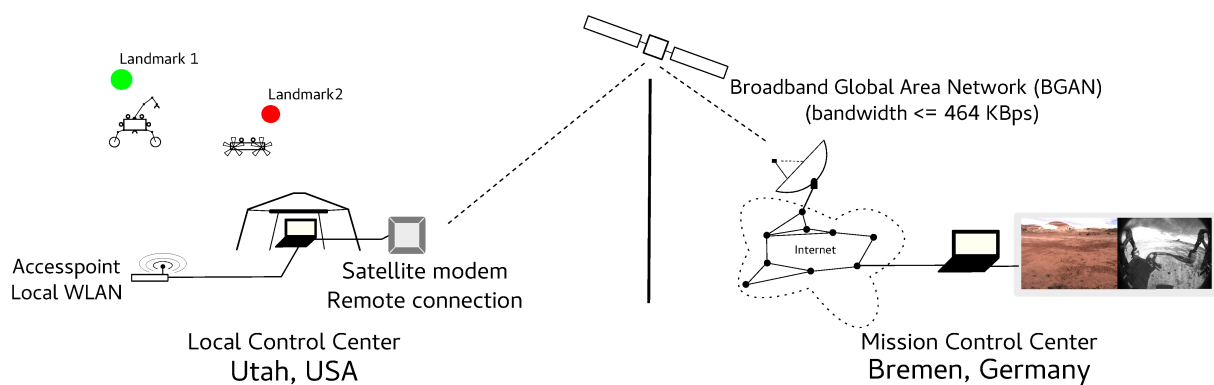


Figure 6.2: The general mission control setup, including a local control centre which redirects the data streams between mission control and robots.

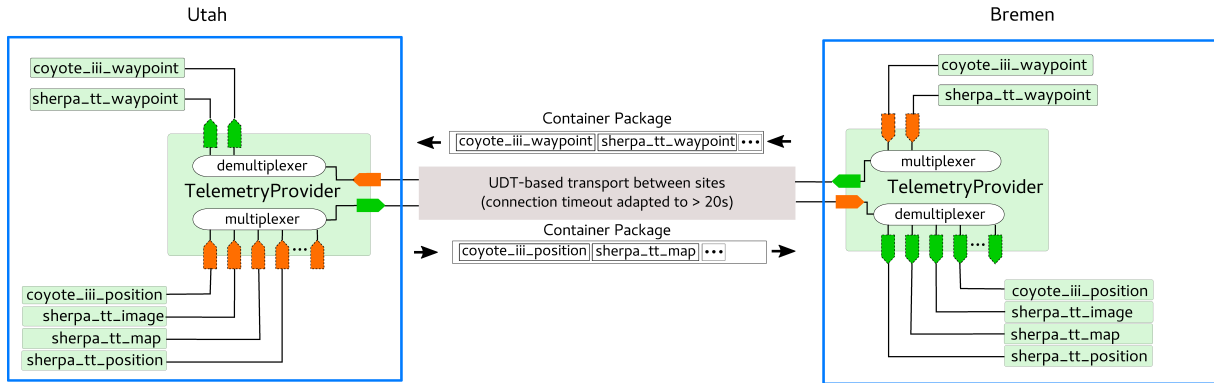


Figure 6.3: Reuse and setup of the demultiplexing and multiplexing infrastructure components for the inter-site communication communication.

Robot autonomy

The analogue mission involved the robots SherpaTT and Coyote III. The mission neither used payloads nor a coalition structure change. Each robot is able to build its own environment map using a distributed SLAM approach and can navigate autonomously towards a given waypoint. Hence, mission operators could request an environment map from the robots as well as images of the environment, so that suitable waypoints could be identified. A set of approximate target science points has been communicated to the mission operators by the local team prior to the actual mission. This approach is similar to a science team which communicates the desired target. The exact waypoints, however, were only defined and known by the mission operators until communicated to the robots.

Mission performance

The mission was performed between 10:30 am and 11:30 am local time (Mountain Standard Time) in Utah - all data presented in the following refers to the absolute starting time of 10:30 am. Figure 6.4 illustrates the robot poses throughout the mission. No ground truth data has been available. Hence, the plot reflects the positions assumed by each robot according to the distributed SLAM approach. The position assumption can change abruptly as the result of pose corrections, triggered by new sensor input. Figure 6.4 shows larger corrections for Coyote III, compared to SherpaTT. This behaviour can be attributed to the different sensor equipment and parametrisation in combination with the traverse through feature-poor areas: Coyote III used a maximum sensing distance of 15 m while SherpaTT used a maximum sensing distance of 40 m. Figure 6.5 illustrates the relative distances between the operating robots. Based on the robot's assumed poses, the maximum distance between both robots did not exceed 30 m. The robots were constantly operating in line of site to each other, so that a direct robot-to-robot communication was possible at all times during the mission. Figure 6.5 as well as several following illustrations visualise the sequentially performed mission: firstly, SherpaTT is commanded to its destination by using three intermediate waypoints. Subsequently, Coyote III starts from approximately 1400 s into the mission. Coyote III is directed towards a rendezvous location to meet SherpaTT. Plateau phases in Figure 6.5 point to phases where a robot does not change its pose. This can result from idle times resulting from preparation activities of the human operator's in the mission control centre, e.g., between 2500 s and 3000 s

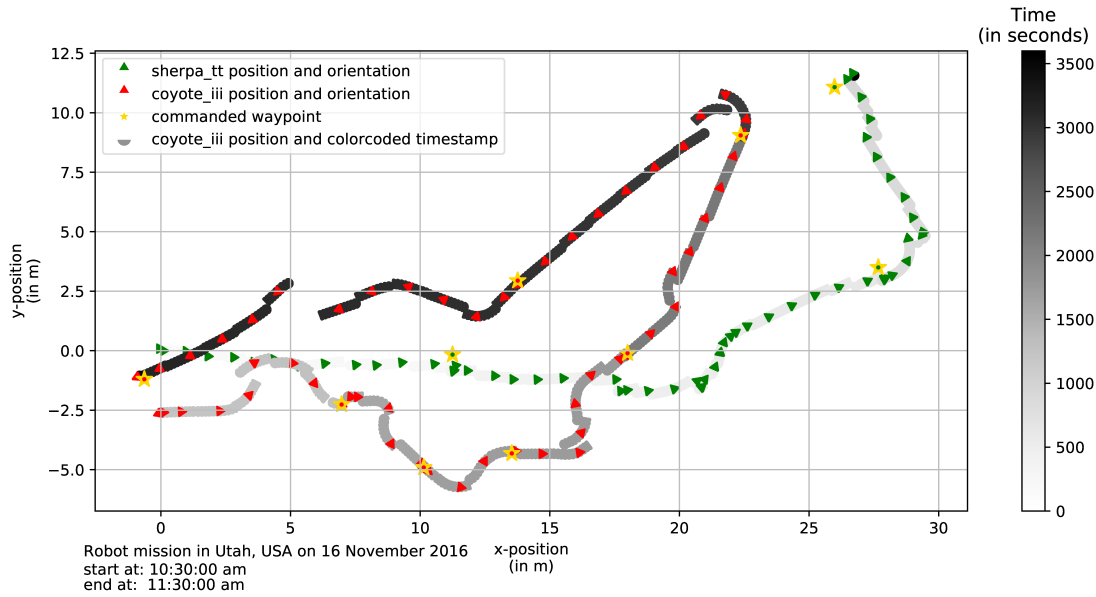


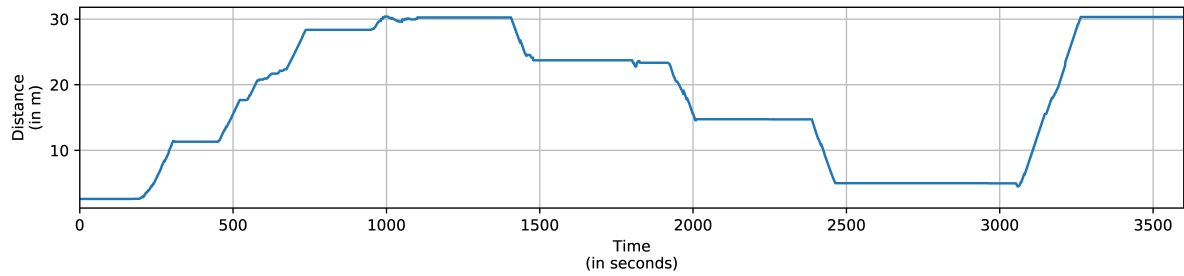
Figure 6.4: The assumed positions of the robots. Progression of time is encoded in greyscale. Approximate location coordinates: local control centre at (0,-35), Landmark 1 at (30,15), Landmark 2 at (20,-15).

SherpaTT's manipulator was controlled from the mission control centre. However, plateaus phases also indicate robot activities such as manipulation. This activity can also be seen by the corresponding higher communication rate of (arm) joint data in Figure 6.7.

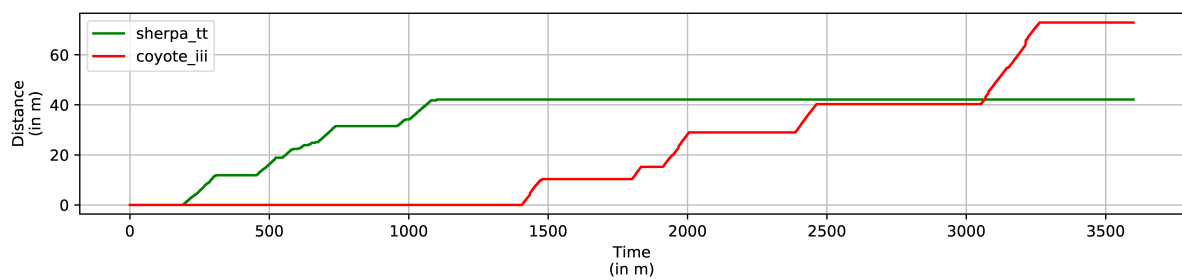
According to Figure 6.5 the combined travelled distance of both robots exceeds 100 m within one hour of operation. An even higher rate could have been achieved by a parallel execution of both systems. For comparison, the anticipated speed of the ExoMars Rover in full autonomy mode is 14 m/h according to Winter et al. (2015).

Communication between mission control and local control centre

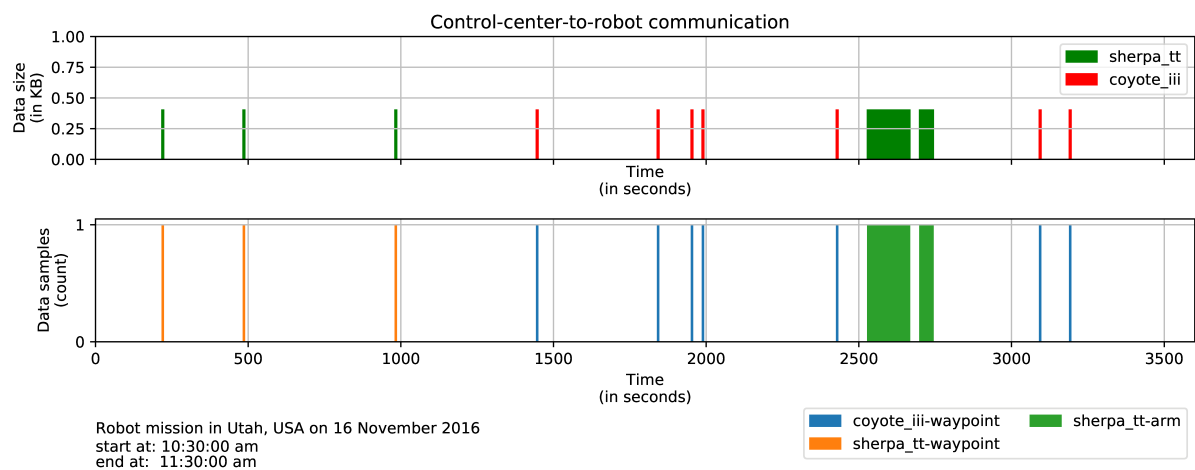
The local control centre proxies the data between the mission control centre and the team of robots. Figure 6.6 shows a low frequency of control commands to the robots in order to define waypoints. A high outgoing and incoming data rate can be observed between seconds 2500 and 3000, and this corresponds to the remote control of SherpaTT's manipulator. Figure 6.6 shows that control commands typically consume few data volume, in contrast to the transfer of images and maps as seen in Figure 6.7. For that reason the setup of the mission control station by Planthaber, Mallwitz, and E. A. Kirchner (2018) also allowed to control the telemetry data sending frequency. Both figures show again the sequential flow of the operation and the data transfer changes correspondingly - since it has been adapted by the operator in the mission control centre.



(a) Distance between SherpaTT and Coyote III.



(b) Travelled distance per robot.

Figure 6.5: Inter-robot and travelled distance for the analogue mission.**Figure 6.6:** Communication from mission control (Bremen) to local control centre (Utah).

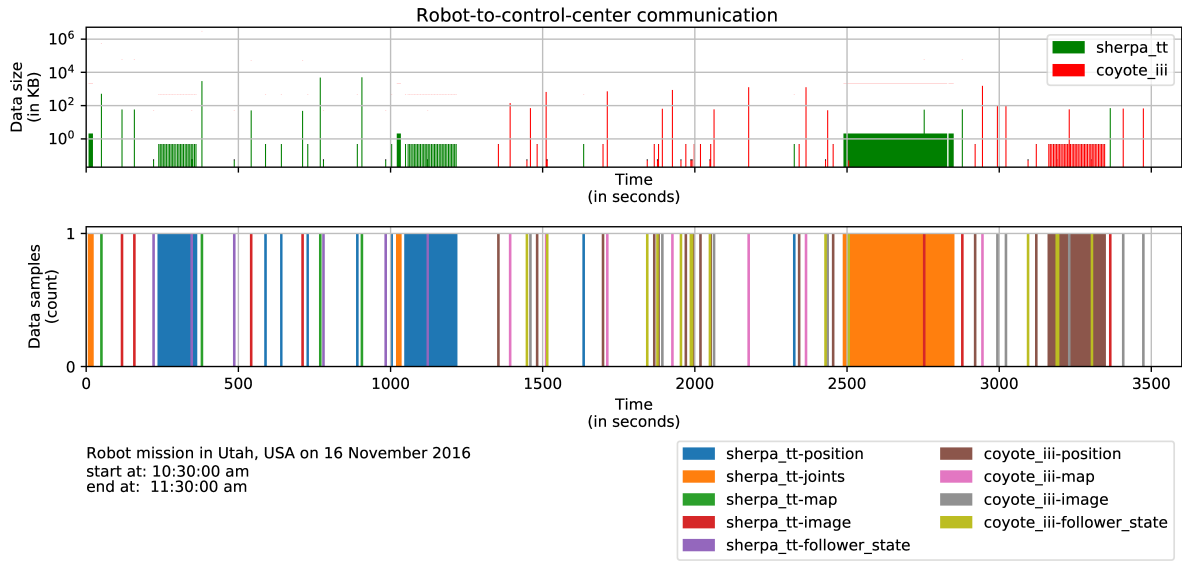


Figure 6.7: Communication from local control centre (Utah) to mission control (Bremen).

Communication between robots

The direct communication between SherpaTT and CoyoteIII servers mainly the distributed SLAM approach. This distributed SLAM approach has been developed by Sebastian Kasper-ski and it is based on previous work by Eich et al. (2014). The distributed SLAM serves as an example client application for the distributed communication architecture and acts as place-holder for any solution which enables the identification of robot poses relative to each other. The approach relies on graph-based SLAM approach where the so-called localised pointclouds are stored with spatial constraints. The localised pointclouds represent nodes in the graph and each edge corresponds to a spatial constraint. To avoid sharing of the complete map, localised pointclouds and spatial constraints are shared between robots. Localised pointclouds and spatial constraints are globally identifiable, and they can also be directly requested from any robot by referring to this id. Each localised pointcloud is a sub-sampled pointcloud of the robot's currently sensed environment. Localised pointclouds are sent in combination with spatial constraints, but they are only sent when a robot changes its position or when an update is explicitly requested, e.g., by a human operator or the robot's supervision. The exchange of the localised pointclouds is based on the FIPA-based distributed communication infrastructure. Figure 6.8 illustrates the sending of FIPA letters for each robot including the size of a letter and the corresponding multiplexed samples that are communicated. The sending of data is triggered by a robot's location change, so that Figure 6.8 can also be interpreted as an activity graph for both robots.

The overall communication volume for inter-site and inter-robot data is depicted in Figure 6.9. The total communication volume for inter-site and inter-robot data has the same magnitude, but the inter-robot communication channels have more remaining bandwidth. As already mentioned, the inter-site communication has already been actively managed in order to reduce the overall communication volume. The inter robot communication is not explicitly restricted, although several characteristics of the distributed SLAM approach - such as sending of data only

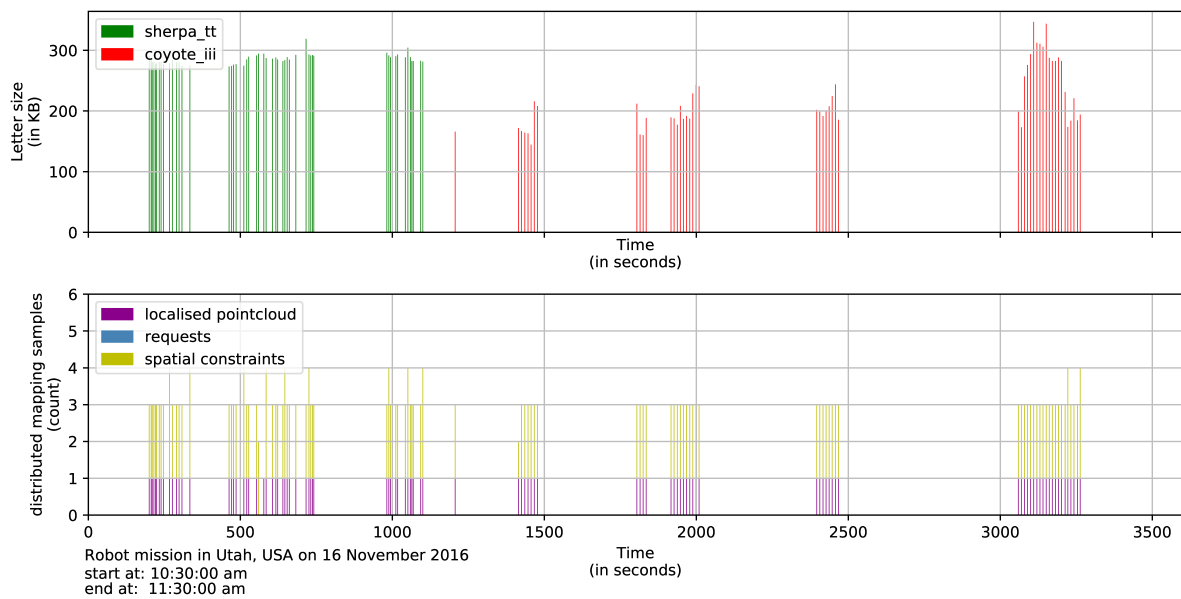


Figure 6.8: *Communication between the participating robots using the distributed communication infrastructure.*

when the robot moves - aim at a reduction of the communication volume.

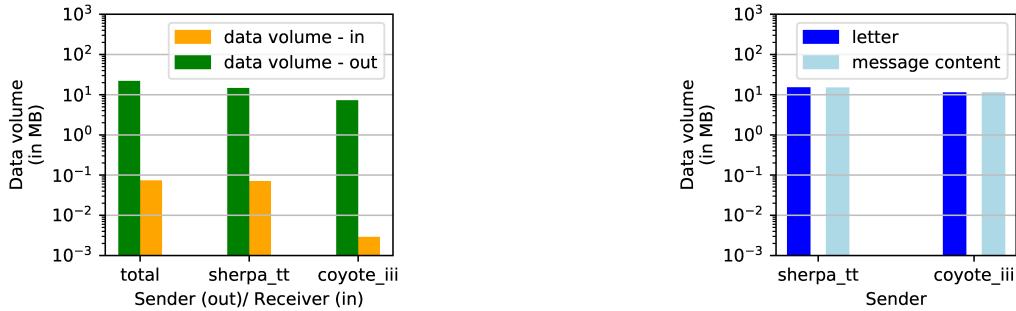
6.1.2 Validation of Subsystem Functionalities

During the field tests in Utah two major robot functionalities that are needed for a sample return mission have been tested: payload pickup and soil sampling. The payload pickup procedure has been developed by Sankaranarayanan Natarajan in cooperation with the author of this thesis based on the works by Roehr, Cordes, and F. Kirchner (2014), and the soil sampling procedure has been developed by Sankaranarayanan Natarajan and tested and incrementally improved with the author of this thesis (cf. (Brinkmann, Cordes, et al. 2018; Brinkmann, Roehr, et al. 2018)).

Payload Pickup

Payload pickup is an essential element for reconfiguration of the multi-robot team and it relies on a visual servoing approach. Figure 6.10 shows the general image data from the manipulator's camera during the process of a payload pickup. The payload pickup sequence failed under the presented conditions due to hard shadows of the EMI which prevented the detection of the inner two markers. The inner marker set is required for the final docking of the end effector to the payload due to the limited field of view of the camera in the EMI, and the field of view additionally narrows when the end effector approaches the EMI.

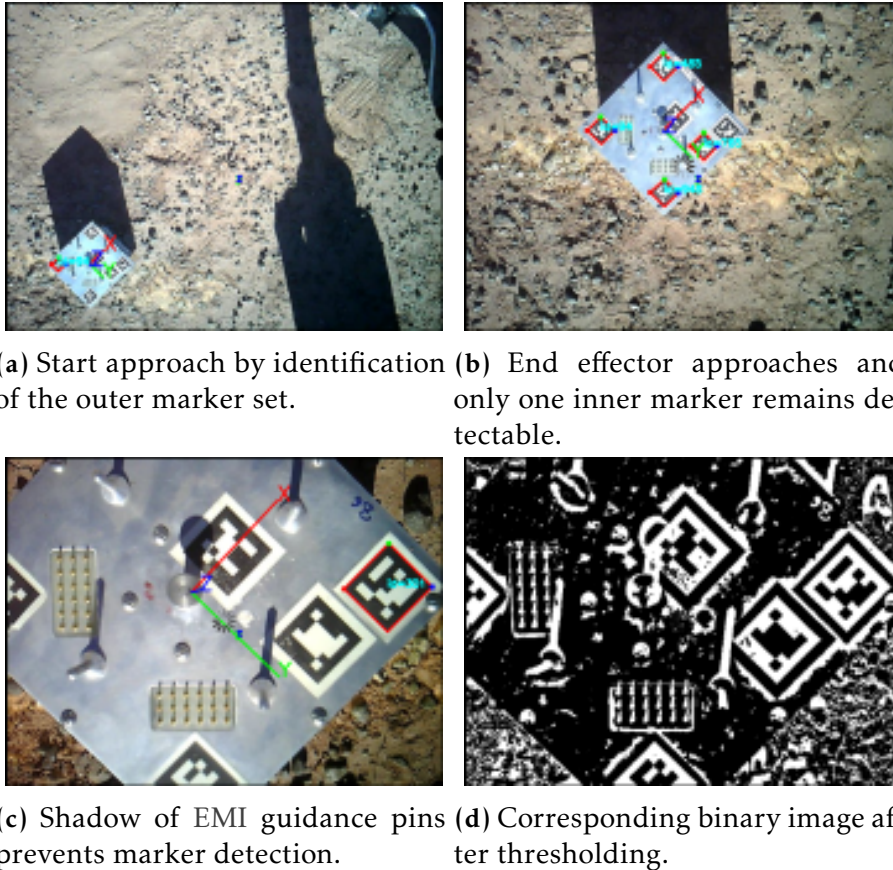
This outdoor experiment illustrated the lighting condition of a target environment and showed a point a failure as the result of augmenting the EMI with an additional marker set. Lighting conditions had been initially validated, although only for the initial (outer) marker set as illustrated in Figure 6.11. The following, intermediate addition of two markers intended to increase



(a) Data exchange between robots (local control centre) and mission control.

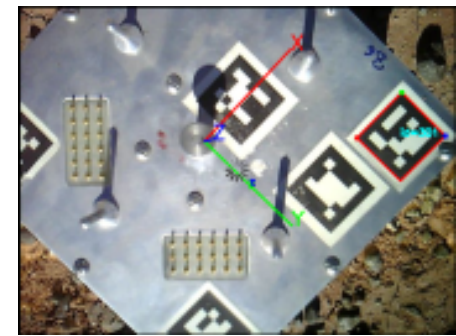
(b) Direct data exchange between robots.

Figure 6.9: Communication volumes for inter-site and inter-robot communication. The comparison between letter and actual message content size for the inter-robot communication shows the minimal protocol overhead in a real application.



(a) Start approach by identification of the outer marker set.

(b) End effector approaches and only one inner marker remains detectable.

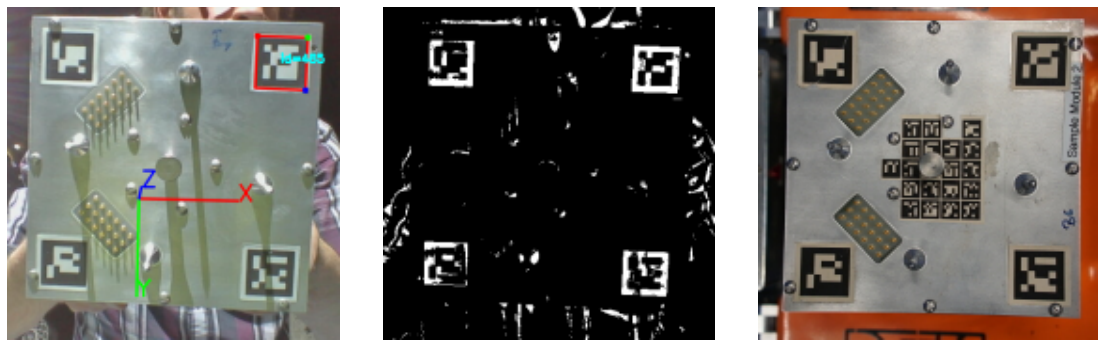


(c) Shadow of EMI guidance pins prevents marker detection.



(d) Corresponding binary image after thresholding.

Figure 6.10: Camera images augmented with the marker detection result during the approach of the end effector to pickup a payload item. Images (a) and (b) are taken during the same approach. Images (c) and (d) refer to an approach at later time of day, indicated by the shadows of the interface pins.



(a) The initial marker setup with only an outer marker set during generation of test data under extreme lighting conditions.

(b) The corresponding binary image after thresholding.

(c) Final marker setup using a highly redundant set of markers.

Figure 6.11: Initial and final marker setup for the EMI.

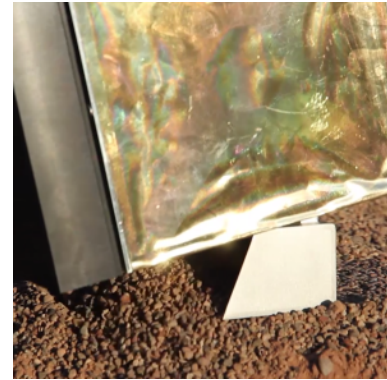
the precision of the overall approach, but at the same time introduced additional restrictions and effectively a point of failure. To increase the robustness of the visual servoing process the inner marker setup has been adapted as illustrated in Figure 6.11c. This final marker setup comes with an increased marker redundancy and thereby improves the detection of markers, so that the general 90° rotational invariance of the EMI can be maintained; the setup with two markers has limited this rotational invariance. An additional benefit of the redundant marker setup is also an improved precision of the detected pose. Figure 6.11a indicates the detected pose by a coordinate system draw into the camera image. The origin of this coordinate system lies ideally in the base of the central pin of the EMI, yet it can be seen that the detection of a single marker can be insufficient for a precise detection of this pose. Increasing the number of detected markers is therefore necessary to achieve a sufficient precision of the visual servoing approach. Section 6.2 presents the success rate of the adapted approach.

Soil Sampling

Soil sampling is an example of an actively designed superadditive functionality. Several specialised payload items have been implemented by Brinkmann, Cordes, et al. (2018) to illustrate the extensibility of reconfigurable multi-robot systems. The sampling functionality requires the mobile manipulation agent SherpaTT and a payload item which has been equipped with a soil sampling system. The so-called sampling module can be attached to SherpaTT's end effector and a procedure control the sampling process. Firstly, the sampling module is lowered to the ground until ground contact is established. Ground contact is identified through a force torque sensor in the manipulator, i.e., a certain threshold of force in z-direction has to be met. The identification of ground contact triggers a dragging of the sampling module towards its body centre in order to fill the shovel. After a predefined distance of dragging the shovel system is closed and the sampling module is lifted. Figure 6.12 shows the test setup in Utah. The implementation of the soil sampling application which has been tested in Utah showed that the sampling module has a too limited opening angle: the manipulator had to be tilted



(a) Sampling module is moved towards the ground.



(b) Sampling module is dragged over the ground.

Figure 6.12: Soil sampling in Utah using a dedicated soil sampling module.

slightly upwards in order to compensate for this limited opening angle in order to successfully sample soil. In effect, the sampling module could not be used for sampling in flat terrain and the design had to be revised. The revised version allows the manipulator's last link to remain parallel to the ground.

6.2 Autonomous mission in Bremen

Several experiments have been performed at the Robotics Innovation Center in Bremen (RH1) in order to achieve a fully autonomous operation of a reconfigurable multi-robot team and to identify the challenges and limitations of an implementation. This section illustrates firstly the experimental evaluation for pick and place actions which are required for uncooperative coalition structure changes. Secondly, it presents the evaluation of the performance of a fully autonomous sample return mission which involves uncooperative as well as cooperative coalition structure changes.

6.2.1 Baseline Mission

All experiments target the autonomous performance of an exemplary sample return mission. The respective mission is outlined in Figure 6.14 and it relies on the robots SherpaTT, Coyote III and a single sampling module. Initially, the single agent SherpaTT and a combined agent which is formed by Coyote III and a sampling module are located at l_0 and l_1 respectively. Coyote III is required at l_2 while in parallel SherpaTT has to perform soil sampling. After the soil sampling completed the sampling module shall return to l_1 . The mission design requires SherpaTT to remain at l_0 over the complete timeframe. This baseline mission turns into the mission specification shown in Figure 6.15, where the organisation model corresponds to the example model illustrated in Chapter 3. Locations are mapped to the following coordinates in the actual test setup: $l_1 = (0.0, 0.0, 0)$, $l_0 = (3, 0, 0)$, $l_2 = (-2, -2, 0)$. The correspondingly planned solution is illustrated in Figure 6.16.

The baseline mission illustrates a remaining gap between the mission planning and the exe-



(a) Revised sampling module.

(b) Repeating sampling procedure with unrevised sampling module, illustrating the need for an upwards tilted end effector.

(c) Sampling procedure with revised sampling module allows for the manipulator's last link to remain parallel to the ground.

Figure 6.13: Soil sampling as example for revising the design of a superadditive functionality.

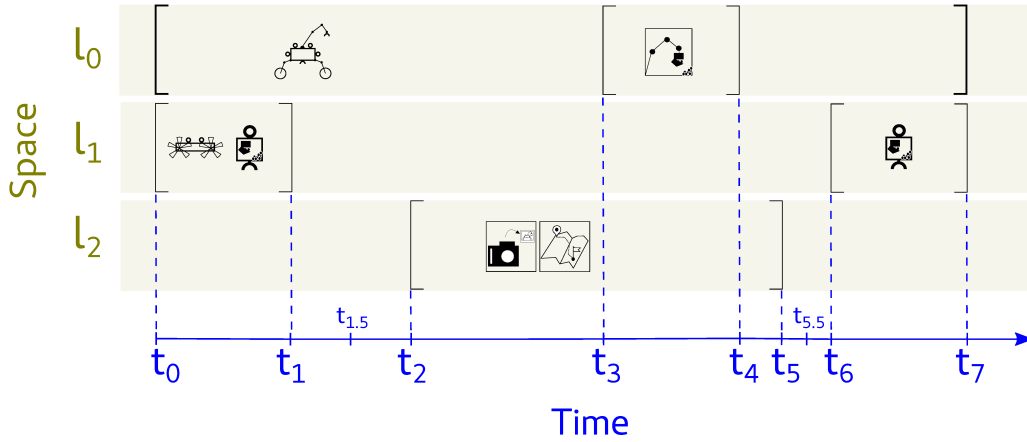


Figure 6.14: Baseline mission for a multi-robot sequence involving reconfiguration, soil sampling and sample return. The consideration of timepoints $t_{1.5}$ and $t_{5.5}$ allows Coyote III to deviate from a straight line path when transitioning between l_0 and l_1 .

$$\begin{aligned}
 \widehat{GA} &= \{(CoyoteIII, 1)(PayloadSoilSampler, 1)(Sherpa, 1)\} \\
 STR &= \{(\emptyset, \{(Sherpa, 1)\})@ (l_0, [t_0, t_7]), (\emptyset, \{(CoyoteIII, 1)(PayloadSoilSampler, 1)\})@ (l_1, [t_0, t_1]), \\
 &\quad (\{LocationImageProvider\}, \{\})@ (l_2, [t_2, t_5]), (\{SoilSamplingProvider\}, \{\})@ (l_0, [t_3, t_4]), \\
 &\quad (\emptyset, \{(PayloadSoilSampler, 1)\})@ (l_1, [t_6, t_7])\} \\
 X &= \{t_0 < t_1 < t_{1.5} < t_2 < \dots < t_7\} \\
 OM &= \dots
 \end{aligned}$$

Figure 6.15: Mission specification of the baseline mission.

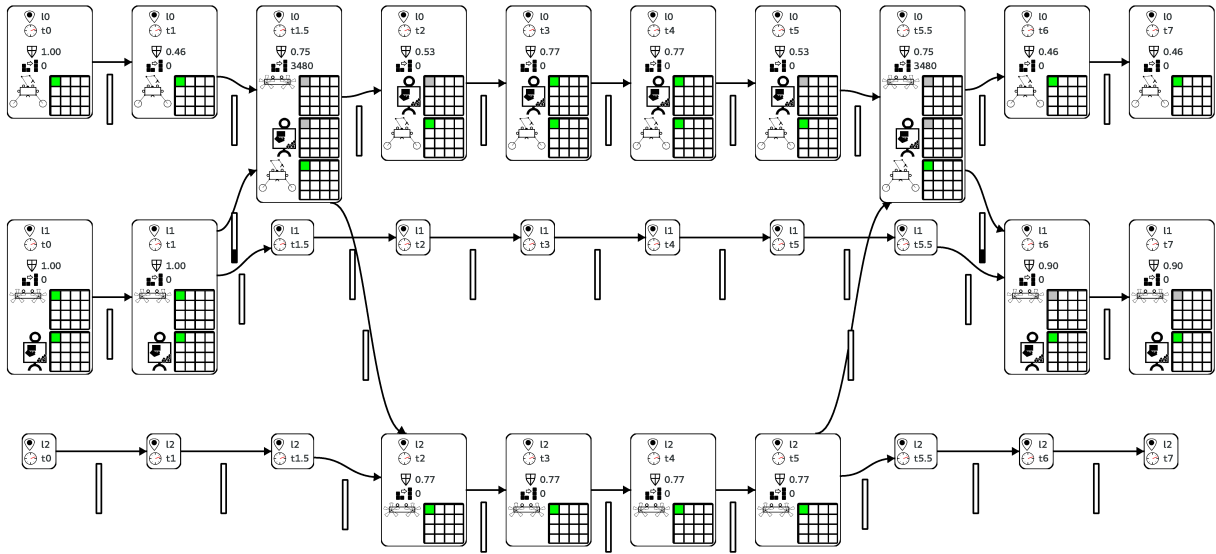


Figure 6.16: A solution computed by *TemPl* for the baseline mission.

cution layer. The baseline mission involves the core elements of a multi-robot sample return mission, but it can be further extended. For example, sending Coyote III off to a designated waypoint can be associated with an additional exploration activity. SherpaTT does change its position in this experiment. The evaluation still relies on a distributed mapping approach and Coyote III navigates to waypoints which are defined by poses relative to SherpaTT. Hence, the evaluated action subsequence remains generically applicable.

6.2.2 Experiments

In order to prepare the autonomous operation of the full mission several experiments have been performed. The corresponding experiments are listed in Table 6.1. The scope of the experiments reflects an incremental increase of complexity to finally achieve the full baseline mission.

Uncooperative Reconfiguration

Attaching SherpaTT's manipulator to a payload and placing a payload onto another male EMI, e.g., by attaching it to Coyote III, are essential elements to demonstrate the capability for uncooperative reconfiguration. The trials are listed in Table I.1 in Appendix I. All 32 trials involve payload pickup with SherpaTT's manipulator and a subset of 12 experiments also involves the placing of a payload onto Coyote III. Both, pickup and place, rely on the same docking procedure although a different parametrisation for the camera calibration has to be used - depending on the EMI camera which is used for the visual servoing approach. Camera mounting, camera calibration, marker placement, marker pose detection, and the manipulator itself influence the general accuracy of the approach. To reduce the number and impact of external error sources, (intermediate) target poses are defined as relative poses with respect to the identified marker poses. These target poses are identified through the following procedure:

- attach end effector to top EMI of payload and position payload on ground

Table 6.1: *Experiments with corresponding descriptions.*

Experiment	Description
D-0X	Pick up of a sampling module, human operator triggers the procedures
D-1X	Pick up of a sampling module and placing it back onto Coyote III, human operator trigger the procedures and defines waypoints
RS-VXX	Sequence of payload pick and placing it back onto Coyote III, involving Coyote III to autonomously approach SherpaTT from a known location and returning to this location, SherpaTT triggers all action of a predefined procedure
FM-IXX	Integration testing of the performance of the fully autonomous execution of the baseline mission, SherpaTT triggers all actions of a predefined procedure
FM-VXX	Testing of the fully autonomous execution of the baseline mission, SherpaTT triggers all actions of a predefined procedure

- detach end effector and move end effector in z-direction upwards for predefined distance, the detected marker position is denoted by *inner alignment pose*, and the moved distance the corresponding distance to command linear movement for blind docking from this pose
- move end effector in z-direction upwards for a predefined distance, the detection marker pose is denoted by *outer alignment pose* and the moved distance is again registered for blind linear movement

This teaching process results in two target poses, one for the outer and one for the inner alignment.

Pick and place are initiated by a search procedure of the manipulator. Although the position of the target might not be exactly known, the target is expected to be encountered in the workspace of the manipulator at SherpaTT's front. The search procedure first performs a spiral movement (cf. Figure 6.17) and continues to align to the outer marker set once a marker has been found. If the search procedure cannot find a target marker, the reconfiguration procedure is aborted. After alignment to the outer marker board, a predefined linear movement according to the previously mentioned end effector teaching procedure is performed. Subsequently, the alignment to the inner marker set is initiated. A successful inner alignment follows again a linear movement to attach the end effector to the EMI. Force torque values which act on the end effector are monitored and if a threshold is exceeded, the docking procedure is aborted and the manipulator returns linearly to a previous pose (defined by a minimum distance to the current pose). Figure 6.18 illustrates the entered task states of the component which controls the pickup of a sampling module (0 to 200 seconds) and the placing of a sampling module (500-800 seconds) for trial FM-V33. The general control approach is slow since the camera in the end effector and female EMIs produces images with an approximate frequency of 1 Hz; Figure 6.18 illustrates these low frequency transitions between marker detection and alignment. The lift payload activity corresponds to a linear movement of the SherpaTT's end effector in z direction. Depending upon the EMI's locking state the end effector might either move with or without an attached payload. The locking status is controlled by another component, so that the activities have to be synchronised by a high-level action. Figure 6.18 shows the significantly longer duration of the placement of the payload compared to the initial placement. For trial FM-V33 the long duration of the placement is due to an outstretched manipulator as shown in Figure 6.19. The wide opening angle of the manipulator's joint 3 results in a increased position

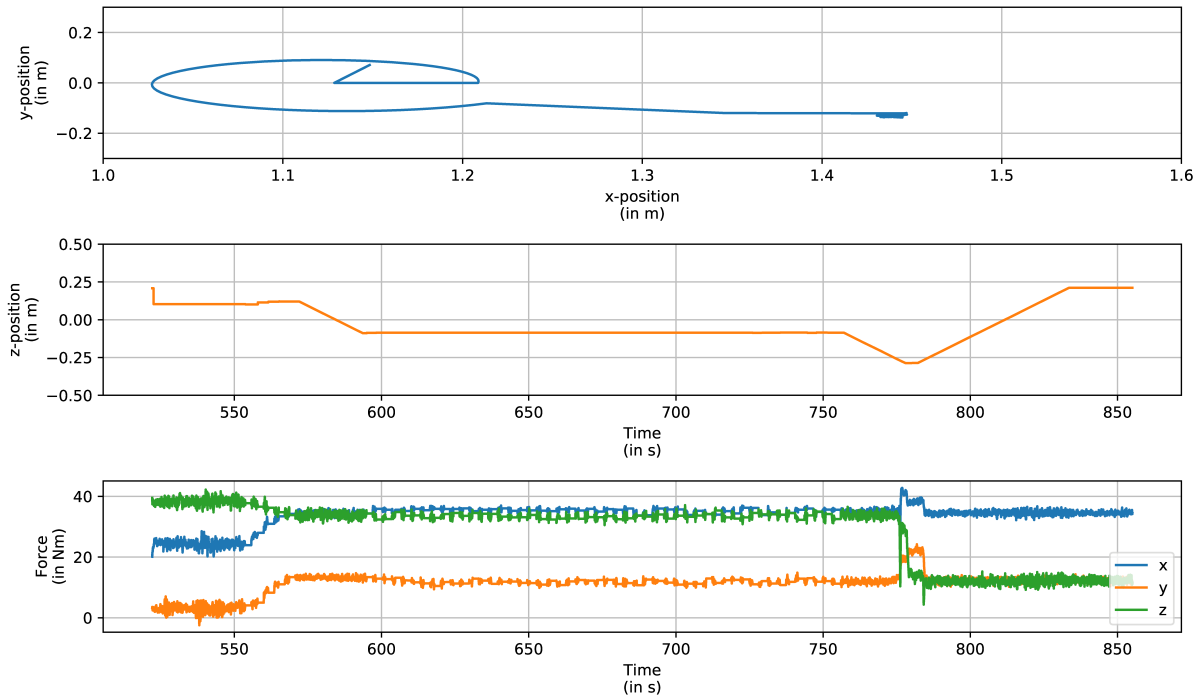


Figure 6.17: Position and force profile during the placing of a sampling module (FM-V33).

error, since play in the first joint has a greater impact in this situation compared to using the end effector closer to the manipulator's base. The combined performance of the pickup and place procedure is shown in Figure 6.20a. The procedures have a combined success rate of 0.84 based on 20 pickup and 12 place actions. The outer alignment is the fast and reliable part of the procedure. However, outer alignment also does not require a high precision and allows for position tolerance on all axes of 2 cm and for the orientation a 8° tolerance for the error of roll, pitch and yaw respectively. The outer alignment is only supposed to establish the start conditions for the precise inner alignment, which requires the inner alignment position to be met with position accuracy of 0.8 mm and allows a maximum orientation error of 0.5° on each axis. To achieve an end effector pose which remains within these tolerances, inner alignment can take up to several minutes until completion. The average duration for all types of alignment remains within a 40 s bound (cf. Figure 6.20b).

FM-VXX: An Autonomous Mission with a Reconfigurable Multi-Robot System

The mission illustrated in Figure 6.14 is the basis for a reference experiment to illustrate the feasibility for the autonomous operation of a reconfigurable multi-robot system. The mission involves SherpaTT, Coyote III and a sampling module and Coyote III in combination with a sampling payload approaches SherpaTT. SherpaTT extracts the sampling module from the composite system, and Coyote III retreats to its initial position. Coyote III moves to a target waypoint and waits for the composite agent consisting of SherpaTT and the sampling module to complete a soil sampling activity. Upon completion of this activity Coyote III is ordered back to take the sampling module and again retreat to the initial position.

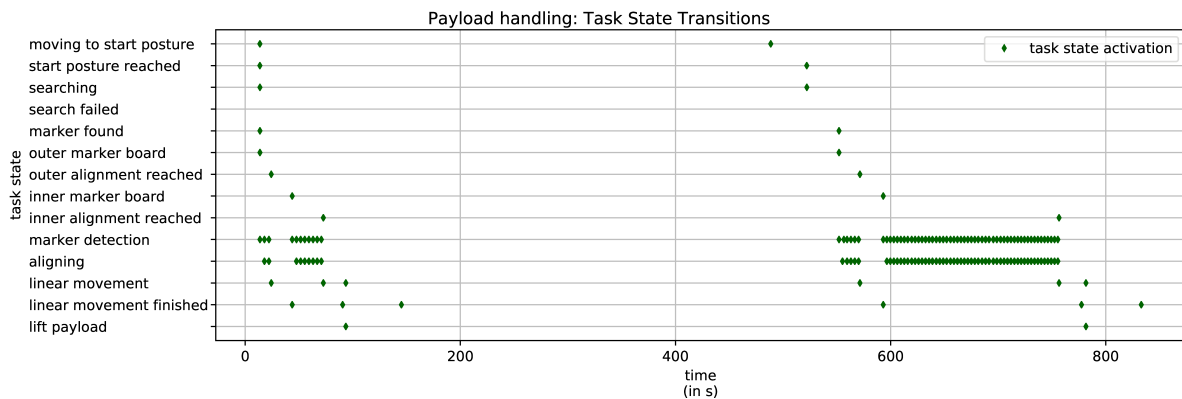
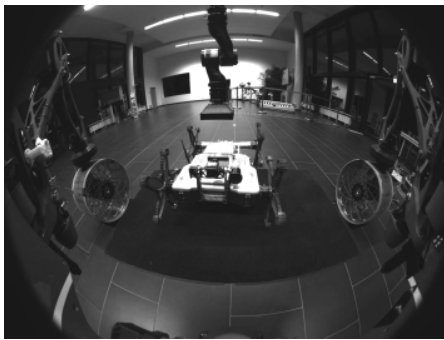
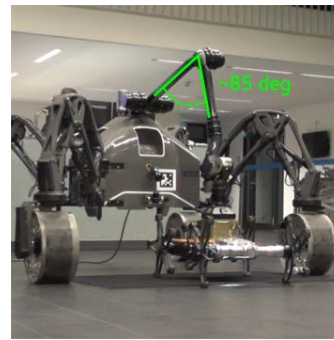


Figure 6.18: Task state transitions for payload pickup and payload placement during experiment FM-V33.



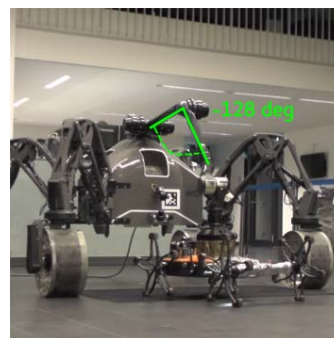
(a) Aligning for payload pickup in the workspace between SherpaTT's legs.



(b) During connecting to payload and lifting, manipulator operates with almost right angle.

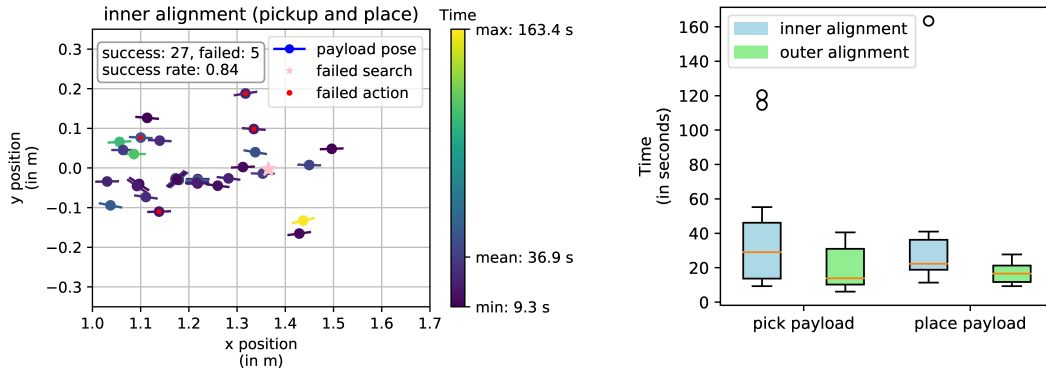


(c) Aligning the payload placement in the workspace between SherpaTT's legs.



(d) Alignment for payload placement with outstretched manipulator, leading to long alignment duration.

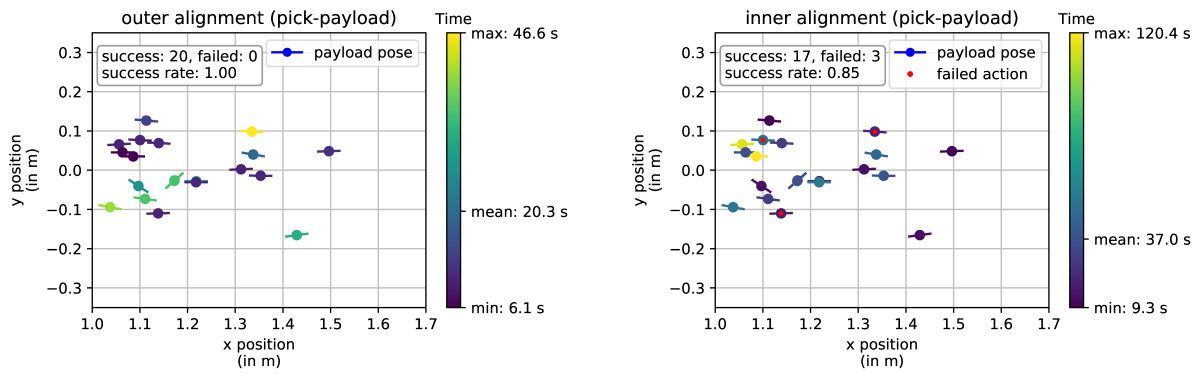
Figure 6.19: Exchange of a sampling module during experiment (FM-V33), pick and place from two perspectives.



(a) Inner alignment over all performed reconfiguration actions.

(b) Alignment duration of successful reconfiguration actions.

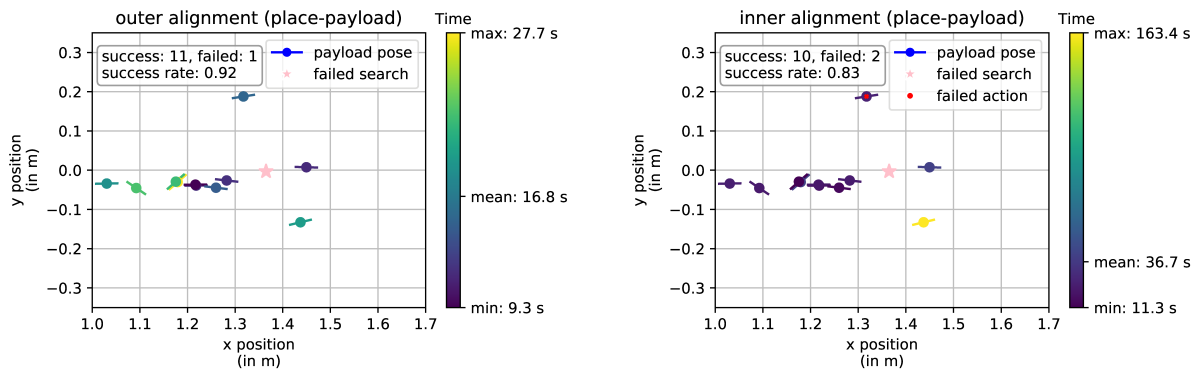
Figure 6.20: Alignment characteristics over all pick and place actions.



(a) Alignment to the outer marker set.

(b) Alignment to the inner marker set.

Figure 6.21: Alignment characteristics for picking a payload.



(a) Alignment to the outer marker set.

(b) Alignment to the inner marker set.

Figure 6.22: Alignment characteristics for placing a payload.

Table 6.2: *High level action sequence for the mission in experiment FM-VXX.*

#	Executor	Action name	Action description
1	SherpaTT	generate map	Trigger the initial map generation on SherpaTT
2	Coyote III	generate map	Trigger the initial map generation on Coyote III
3	Coyote III	move to target marker	Coyote III visually identifies the relative pose of a marker attached to SherpaTT and approaches this pose using a generated spline
4	SherpaTT	pickup payload	pickup of a payload is initiated, involving search for the top interface, attaching and extraction of the payload
5	Coyote III	move blind backward	Coyote III moves linearly 1.7 m straight backwards
6	Coyote III	move to relative target	Coyote III moves to a waypoint relative to SherpaTT, here to relative coordinates x: 5 m, y: 2 m, trajectory planning is performed on the distributed map
7	SherpaTT	pickup soil sample	activation of the payload sampling sequence
8	Coyote III	move to relative target	Coyote III moves back to its origin pose, commanded as waypoint relative to SherpaTT, here relative position x: 3 m y: 0 m
9	Coyote III	move to target marker	Coyote III identifies the relative pose of a marker attached to SherpaTT and approaches the pose
10	SherpaTT	place payload	SherpaTT places payload onto Coyote III
11	Coyote III	move blind backward	Coyote III moves linearly 1.7 m straight backwards

The FM-VXX experiment presents a reconfiguration sequence which can be used as a template for other multi-robot missions. The complete mission is controlled and monitored by SherpaTT, i.e., SherpaTT acts as master while Coyote III acts as a slave and performs commands which it receives from SherpaTT. The initial relative poses of Coyote III and SherpaTT are known to the robots so that a global reference frame exists. The global reference frame is essential for the use of the distributed SLAM. Both robots run distributed SLAM and like for the tests described in Section 6.1 localised pointclouds and corresponding spatial constraints are exchanged between the robots. The distributed SLAM version used for this experiment also exchanges footprint samples, i.e., the current pose of a robot plus the information about the current footprint size of the robot. The continuous exchange of footprint data allows to mask robots in received pointclouds, so that another robot is not perceived as a permanent obstacle in a computed map. Furthermore, a footprint might change during the mission for a robot like SherpaTT.

The performed mission consists of the high level action sequence listed in Table 6.2. Only the backup movement of Coyote III is defined using a predefined distance in order to guarantee that subsequent activities of SherpaTT and Coyote III can be safely executed. Otherwise, the mission approach tries to exploit the shared map and waypoints are defined as relative poses with respect to SherpaTT's (other rather the current commanding robot's) currently assumed pose. SherpaTT transforms these poses according to its current pose, so that Coyote III is still commanded to absolute coordinates.

In preparatory experiments FM-I1X and FM-I2X Coyote III returned to the origin and initiated a blind offset movement towards SherpaTT; this offset movement assumed a sufficiently precise pose of Coyote III at this stage. The experiments showed, however, that Coyote III did return with an orientation error to the origin. To allow for a more reliable approach to SherpaTT,

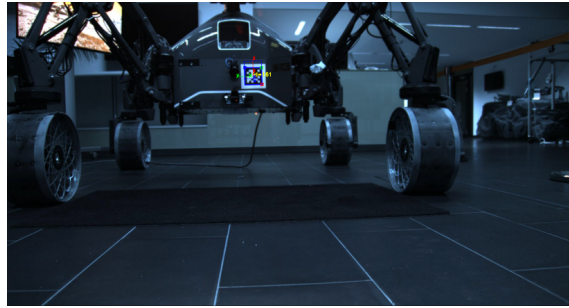


Figure 6.23: *Coyote III camera view which allows to detect the position of SherpaTT with the help of an artificial marker (FM-V33).*

Coyote III moves towards a target marker attached to SherpaTT in FM-VXX. If Coyote III is not able to detect the marker it moves incrementally forward - 0.25 m in straight line - assuming that this change of location improves the likelihood of detecting the marker. The approach is aborted after 4 iterations. Figure 6.23 illustrates Coyote III's successful detection of the marker attached to SherpaTT during trial FM-V33. The design of the approach assumes that the marker will be visible to Coyote III either directly after returning to the location requested by SherpaTT or after moving Coyote III incrementally forward. The final series of experiments showed that these assumptions did not hold, and that a robust approach requires an additional search behaviour for Coyote III to guarantee for a successful cooperative docking procedure.

Robot-to-Robot Communication Robot-to-robot communication is required for commanding robots and for exchanging data as part of the distributed mapping approach. SherpaTT acts as master and sends action commands to Coyote III. Since the activities of both systems have to be synchronised, the implemented communication protocol allows a robot to communicate status changes of each action back to the robot requesting an action. The effect is illustrated in Figure 6.24 by the robot specific action sample count. Coyote III shows a much higher action message volume, as a result of communicating the action status, while SherpaTT only sends action commands.

The exchange of footprint data between both robots leads to a continuous data stream between both systems. It can also be seen that the footprint sample count per message exchange lies at 25 samples and could be reduced to save bandwidth. Localised pointclouds are only exchanged when a system is active and Figure 6.24 illustrates the messaging activity, where the transfer of localised pointclouds corresponds to messages of multiple MB.

The general communication volume is visualised according to the used communication channel. The communication channel for action messages is named after the robot, while additional channels for direct subsystem communication come with an additional suffix. The suffix based naming schema simplifies the design of multicast groups for the distributed communication architecture. Sending a message to '*.sensors' directs the message to all receivers where the name matches this pattern. The overall communication data volume is approximately 47 MB for data sent from SherpaTT and 73 MB for data sent from Coyote III, leading to a data rate of 273 MB/h for Coyote III and 175 MB/h for SherpaTT. This data rate for inter-robot communication is significantly higher compared to the field test in Utah, where the data rate was approximately 15 MB/h for SherpaTT and 11.5 MB/h for Coyote III (cf. Section 6.1) This increased data volume is the result of the continuous exchange of footprint data (cf. Table 6.3).

Table 6.3: *Communication properties for the autonomous operation of an exemplary sample return mission (FM-V33).*

Sender	Data Volume (in MB)	Data Rate (in MB/h)	# of Msgs	Msg Frequency (in Hz)	Msg size (in Bytes)
Coyote III	0.060	0.22	64	0.066	940 ± 3.6
Coyote III Sensors	72.822	172.01	873	0.89	83416 ± 218060
SherpaTT	0.007	0.03	7	0.0072	984 ± 32
SherpaTT Sensors	46.673	268.38	924	0.95	50512 ± 18816

Mission Performance Figure 6.25 visualises the performance of three trials and additional image impressions are provided in Figure I.3 in Appendix I. Trial FM-V32 has been left out, since it was failed right after the payload pickup process. Trial FM-V31 failed to due a misalignment of Coyote III before initiating the placing of the sampling module. The implemented target alignment procedure could not compensate for the misalignment, since the target was not visible for Coyote III. Figure 6.25a shows the iterative forward movement of Coyote III leading to waypoints 5 to 8 - the approach is aborted after exceeding the permitted number of iteration.

The successful performance of a fully autonomous sample return mission sequence for FM-V33 is illustrated in Figure 6.25b. Despite an orientation error of Coyote III at waypoint 4, the placing of the sampling payload succeeds. As already mentioned, the duration of the alignment is significantly higher in this trial compared to the mean performance. Also visible is a spurious pose jump of Coyote III's assumed pose during the final backup movement. Since the robot is relying on the pose information in that situation, it has no further effect, but points to an issue of the distributed SLAM.

Figure 6.25c illustrates a mission sequence, which only succeeded with operator intervention. Two pose errors of Coyote III had to be compensated for by manually realigning the robot at waypoints 4 due to a too large orientation error, and at 9 due to overshooting - out of the manipulator's workspace. The overall mission sequence ran, however, continuously and was neither explicitly halted, nor interrupted. Coyote III automatically continued to approach the target after manual correction, and likewise did the search procedure of the manipulator identify the outer marker set after moving Coyote III back into the workspace.

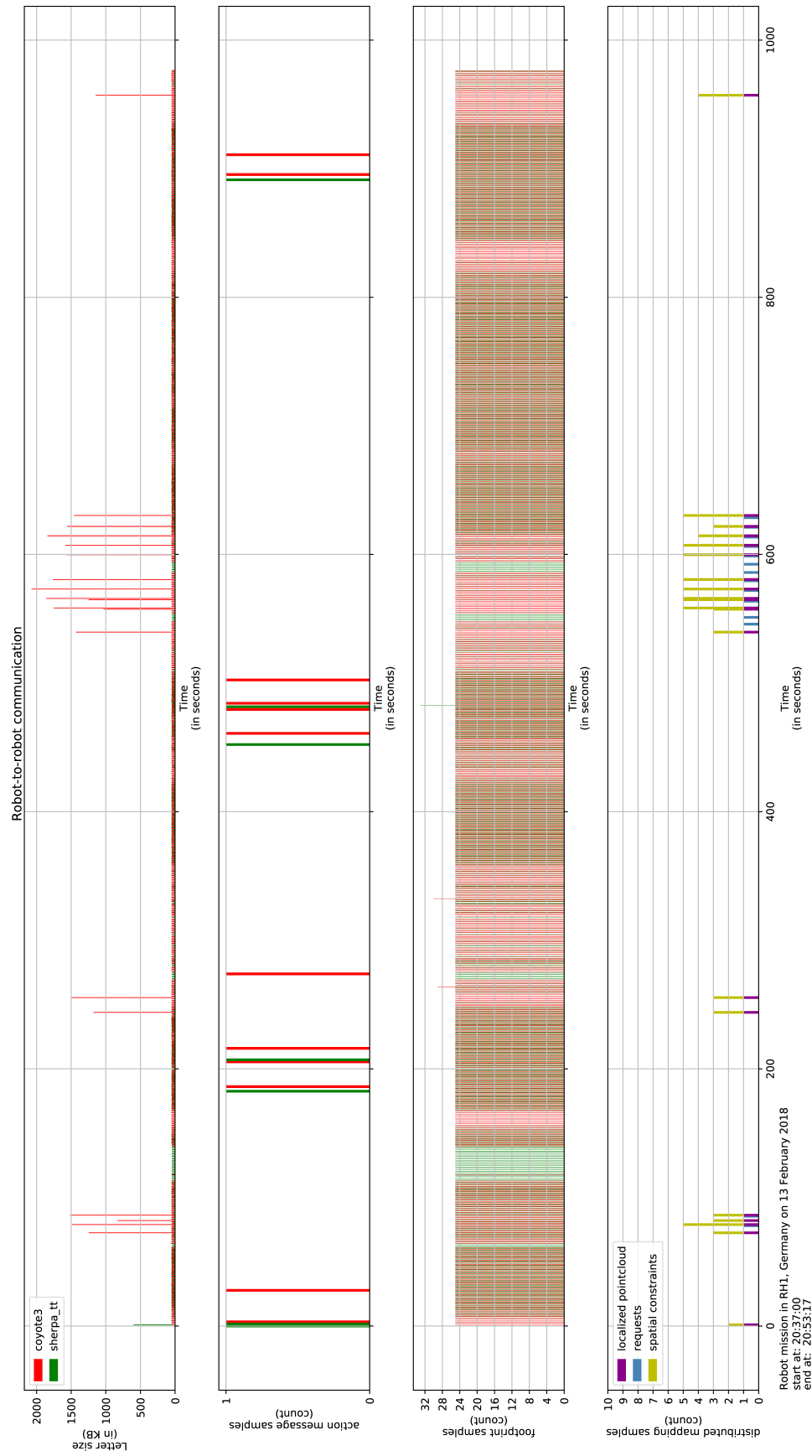


Figure 6.24: Robot-to-robot communication during a fully autonomous mission sequence (FM-V33).

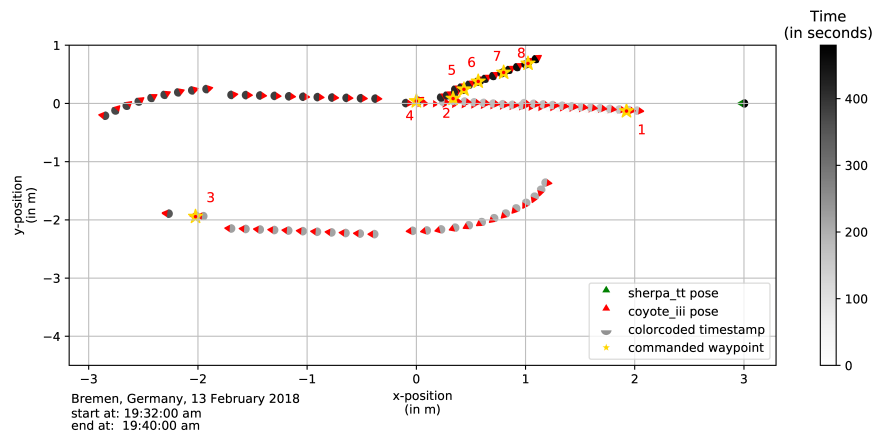
6.3 Discussion

This section discusses findings and lessons learnt from the field trials in Utah and the experiments at DFKI Bremen with respect to the autonomous operation of reconfigurable multi-robot systems.

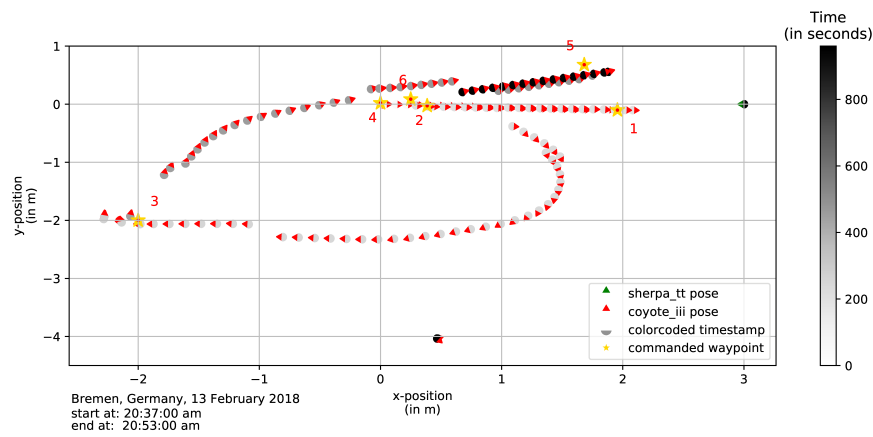
Operational Infrastructure The setup of an operational infrastructure is an essential precondition for any kind of robot operation. While a number of available robotic framework exist for the development of such infrastructure, the presented approach relies on the use of Rock and the implemented operational infrastructure particularly extends Rock's functionality with respect to multi-robot communication. The developed components can also extend other frameworks, so that the use of Rock serves only as a reference implementation. The experiments in Utah have verified the applicability of the communication architecture. Robot-to-robot communication was established using the distributed communication architecture and the connection between mission control centre and the local control centre was also based on core components of this architecture. The mission in Utah did not use a communication mesh, but the mesh has been successfully verified as part of the experiments in Bremen (RH1). In effect, two multi-robot missions have been evaluated and the communication could be characterised for a semi-autonomous and a fully autonomous mission. Some low-bandwidth data channels, e.g., such as a satellite link, require explicit data management, as it is the case for most space missions. Meanwhile, a robot-to-robot communication for an autonomous mission can exploit a high bandwidth, e.g., assuming an available bandwidth of 20 Mbit/s for a wireless link corresponding to data transfer rate of 8,78 GB/h. The observed consumption for two mobile agents in the sample return mission requires 442 MB/h, an approximately 5 percent of the anticipated available bandwidth. An optimisation of communication is required as soon as the number of agents increases or when message relaying is needed. Due to a relative small maintained inter-robot distance, none of the performed experiments has required the use of the relaying. The high-level coordination requires only a low data volume. In contrast, sub-system communication such as the distributed SLAM require an active management of the data volume, e.g., a reduction of sending robot footprints with lower frequency will be a first measure.

The usage of standardised messages for inter-robot communication facilitates the development of coordination schemes and has a low protocol overhead. Due to the generic message format additional protocols can be easily embedded, and communication between agents can be stateful. The synchronisation and coordination of robot actions in experiment FM-VXX is based on the use of a content language which allows to communicate the action status between agents. The implementation of this asynchronous remote procedure call is an example of the general flexibility of the multi-robot communication infrastructure. Overall, the usage of the operational infrastructure for two sample missions, proves the applicability of the distributed communication architecture to perform autonomous operation of fully distributed reconfigurable multi-agent system.

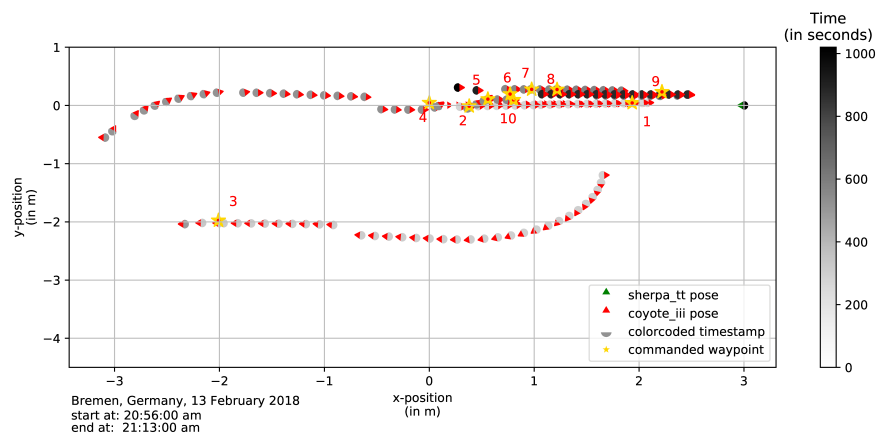
From Planning to Execution Chapter 4 has introduced a mission planning approach for reconfigurable multi-robot systems. The planner has not been embedded directly on the robots, but a baseline mission has been developed in order to identify the remaining gap between the mission planning and the execution layer. The selected baseline mission firstly illustrates an



(a) Trial FM-V31.



(b) Trial FM-V33.



(c) Trial FM-V34.

Figure 6.25: Robots' assumed poses during the sample mission - Coyote III initially at (0,0), SherpaTT remains at (3,0). Target waypoints are numbered according to the resulting sequence. Note that the orientation of Coyote III does not change, e.g., when moving from waypoint 1 to 2, since the robot is driving backwards (see Table 6.2).

additional characteristic of the planning approach: the explicit definition of qualitative timepoints constraints the solution space, e.g., a mobile agent cannot deviate from its path when it transitions between two locations and no intermediate timepoint exists. In the baseline mission Coyote III can only transport the sampling payload to SherpaTT between t_0 and t_1 , due to the existence of the intermediate timepoint $t_{1.5}$. This characteristic can be interpreted as a limitation of the current planning approach which results from the use of the temporally expanded network. However, additional preference constraints can also be derived from this observation, e.g., constraints which restrict or permit the deviation from the direct path for a transition between two locations.

The implementation of the planned baseline mission shows, that coalition structure changes vary in time and therefore vary in cost. The implemented autonomous coalition structure change shows only a payload exchange, but it illustrates the complexities of structure changes and their dependence on subsystems. For a cooperative approach two robots have to be operational so that a coalition structure change can be performed. However, robot activities can fail and failure handling routines are required for the cooperative approach, so that necessary preconditions for further robot actions can be (re)established. Additionally, coalition structure changes can fail and therefore have also to be accounted for as a risk. The implementation of the sample return mission shows, that a cooperative docking approach will benefit from additional redundancy. Such redundant design would estimate robot poses from the distributed SLAM as well as by visual means, and fuse this information for a fully cooperative docking approach, where each robot can monitor and support the activity of the other robot.

EMI: a Single Point of Failure The EMI is the key element of the reconfigurable multi-robot system. But the EMI is also a point of failure and its design is critical for many levels of operation. Hence, the design of the EMI has not only to meet physical requirements such as load, data and power throughout. The design must also consider physical and virtual guidance (visibility or detectability) to support automated docking procedures and account for context dependant influences of EMI features during such operations. The last aspect refers to the observation that the EMI's mechanical guidance pins have been implemented to support the docking process. Under a low positioned sun, they lead to long shadows on the interface which critically influence the visual detection of the interface pose. Therefore, any EMI design that enables a reconfigurable multi-robot system must be critically assessed and validated under conditions of the target environment and after defining acceptance tests. An additional single point of failure is the marker-based, pose-guided visual servoing approach which relies on the bottommost camera attached to the end effector. Hence, a payload item can only be exchanged as long as the bottommost camera is operational. Additional redundancy for the marker-based pose detection can be introduced by the previously mentioned pose estimation. The detection of a robot's pose in combination with either prior and ad-hoc exchanged information about the relative target interface pose, can serve as basis for an alternative docking approach. This information can also be part of an extended organisation model. As a benefit of reconfigurability, a failing end effector camera can also be compensated for by attaching a payload item.

The validation experiment uses an identical set of markers for all male EMIs. For scalability, however, each EMI has to be uniquely identifiable, e.g., by using an additional marker for identification. Likewise the organisation model has to be maintained in a distributed way, in order to allow a mapping between interfaces and the corresponding agents.

The EMI design can come with features which limit its applicability, e.g., as shown for the

operation with long shadows. However, a multi-robot system has an increased potential to actively control the operation environment. Lighting conditions can be actively influenced to avoid reflections and shadows, e.g., by either changing the docking approach and the position of each mobile agent or by using collaborative systems to shadow or illuminate designated areas.

Visual Servoing The current design of the visual servoing approach has limitations. Firstly, it depends on a teaching procedure. This procedure will not be necessary when either EMIs (including the augmentation with markers) can be manufactured with sufficient accuracy and precision, or when the marker detection is replaced with a more capable interface detection algorithm. The marker detection acts as a placeholder for any kind of interface and robot detection mechanism. Secondly, the current approach does not exploit the rotational invariance of the EMI design and uses only a single reference pose. But a full exploitation of the rotational invariance requires not only the support through the visual servoing approach. The organisation model has to be embedded into the operational architecture also, so that it reflects the current state of the organisation and more specifically the exact geometrical state of a composite agent.

Superadditive Functionality Superadditive functionalities of a reconfigurable system can be actively designed, e.g., as shown by the development of extension modules (cf.(Brinkmann, Cordes, et al. 2018; Sonsalla, Cordes, L. Christensen, Roehr, et al. 2017b)). As seen with the sampling module, a challenge for the design of the tool exchange lies in the identification and formalisation of usage constraints. The design of the sampling module has been performed without the explicit validation through a higher level sampling routine. So, while the design of reconfigurable (here: immobile) agents can be performed independently from other systems due to the EMI, the example shows that this can lead to severe usage constraints. Hence, a co-development process for those agents which enable a superadditive functionality should be favoured. Furthermore, known geometrical constraints could also be used to characterise agent functionalities further, so that a more sophisticated agent description has to be part of the organisation model. While the active design of superadditive functionality will be the primary focus for most developments, some superadditive functionality can also be discovered as the result of emergence. A detailed geometrical description in the organisation model can also be combined with evolutionary search in order to identify sleeping functionalities.

Generalisation of Coalition Structure Changes Coalition structure changes require detailed action planning for the involved robots. Reusable and parametrisable scene scripts for the cooperative approach of two robots can facilitate the execution of multi-robot missions. Since they form essential building blocks of a reconfigurable multi-robot mission any increase of robustness and speed of performing these scene scripts will lead to a higher success rate of reconfigurable multi-robot missions. A generic scene script for a docking approach between two agents a_0 and a_1 is provided with Algorithm 1. The algorithm suggests an increased level of active cooperation between the two approaching agents, so that inter-robot communication has to be intensified. The experiment FM-VXX was based on an initially known shared reference frame. A scalable approach builds on robots which are capable to establish a shared reference frame independent of their starting conditions, e.g., by identification and sharing of their relative poses. D. Fox et al. (1999) have shown that accounting for relative poses also improves the

localisation error compared to the use of a single robot for localisation and is therefore an available means to improve any multi-robot operation. The experiment FM-VXX does not negotiate a docking plan, and uses a predefined action sequence instead. Such sequence will be applicable to a limited set of situations, but additional local planning can increase the flexibility of the approach.

Algorithm 1: Pseudo code for a cooperative (two agent) approach.

Data: $A \leftarrow \{a_0, a_1\};$ // Agent pool

```

1  $a_0$  sends request for cooperation;
2 if  $a_1$  does not confirm cooperation request then
3   | abort
4  $a_0$  and  $a_1$  establish a shared reference frame;
5 repeat
6   |  $a_0$  and  $a_1$  exchange known robot poses;
7   |  $a_0$  and  $a_1$  select plan template, actively plan, or negotiate the docking plan;
8   |  $a_0$  and  $a_1$  parameterise and execute docking plan;
9 until  $a_0$  and  $a_1$  agree that target poses have been reached;
10  $a_0$  and  $a_1$  test connection;
11 if connection established then
12   |  $a_0$  triggers software reconfiguration to change to composite agent  $\{a_0, a_1\}$ ;
13 else
14   |  $a_0$  and  $a_1$  negotiate and initiate local repair routine;
```

Subsystem Reliability Experiment FM-V33 has demonstrated the single successful performance of a sample return sequence, and thereby shows the principal feasibility of such approach. However, the success rate is currently 25 percent. A resilient reconfiguration approach requires further improvements regarding error recovery. The reconfigurable multi-robot system used for FM-V33 has multiple subsystems which enable the operation of the multi-robot system. The experiment also stresses the importance of each subsystem for the full performance, or rather the dependence of the performance upon operative subsystems. As long as autonomous failure handling routines are unavailable, a human interaction can be requested as final backup. Trial FM-V34 demonstrates how the online correction of alignment error by a human can turn a failing mission into a success. The overall execution of the mission was not halted or interrupted in FM-V34 and Coyote III's pose errors have been manually corrected. After correction Coyote III was positioned within the required workspace(s) and fulfilled the preconditions for subsequent activities, e.g., allowing to align the end effector. Hence, although FM-V34 could not show a fully autonomous performance, it illustrated a feasible failure handling strategy: agents pause the cooperative approach upon failure and request human interaction, a human operator analyses the situation and corrects the last activity, agents continue with the cooperative approach.

Comparison with Previous Approaches The illustrated experiments are the achievements of operating a reconfigurable multi-robot system with respect to the successive development over different projects. In the following, the experiments are compared to the approaches and

results of the related projects LUNARES (Roehr, Cordes, F. Kirchner, and Ahrns 2010) and RIMRES (Roehr, Cordes, and F. Kirchner 2014).

In LUNARES a mobile, eight-legged scout robot was commanded from a wheeled rover to perform a mechanical docking procedure. The scout was equipped with a set of markers to enable the pose detection of the system from the wheeled rover. The scout itself was not able to verify its correct positioning, but had to enter the field of vision of the rover's camera as precondition of the approach. The final pose error remained relatively high with 9 mm in x-direction, 4 mm in y-direction, 12 mm in z-direction and a yaw error of 0.7° . To fulfil the preconditions for docking of a male and a female EMIs, we developed in RIMRES an approach between the six-legged scout CREX and the rover Sherpa. In this approach the master (Sherpa) does not move during the final approach, but only the slave (CREX). In RIMRES a higher precision and accuracy of the approach was achieved compared to LUNARES. However, the approach in RIMRES took significantly longer compared to the alignment procedure in the previous section, e.g., an alignment from a random yet visible position in the workspace took 161.72 ± 80.69 seconds.

The use of the manipulator for docking as shown in this thesis shows fast convergence and operates with a higher precision - which is likely the result of an optimisation of the marker layout and a better accuracy of the manipulator compared to the locomotion system of CREX. Still, all efforts presented in the previous section and in the referred publications are complementary. They show a set of feasible docking approaches which can be used by reconfigurable multi-robot systems.

6.4 Summary

This chapter illustrates two real world evaluations for the application of reconfigurable multi-robot systems. The field test in Utah verified a semi-autonomous operation approach and confirmed the applicability of the operational infrastructure. Furthermore, the communication characteristics for a fully distributed, semi-autonomously operating robotic teams are empirically analysed.

The experiments in Bremen reflect the incremental development of a fully autonomous sample return mission. The execution of the mission requires uncooperative and cooperative coalition structure changes. Furthermore, soil sampling is the result of activating a superadditive functionality. A evaluation shows a successful performance of an autonomous sample return mission. The overall set of experiments points, however, to a number of remaining practical challenges. The major challenge is the higher system complexity of any reconfigurable multi-robot system compared to a monolithic system. This thesis tackles a subset of identified single points of failures either by increasing larger tolerances, e.g., during the docking process, or by local failure handling routines. Future efforts, however, have to focus on the generalisation of failure handling routine to establish a robust mission execution approach.

7

Conclusion & Outlook

This thesis' focus is on reconfigurable multi-robot systems. Reconfigurability of a robotic system depends on a modular hardware design which enables the dynamic formation of distinct robot types, or rather so-called composite agents. Reconfigurable systems as dealt with in this thesis can adapt their hardware configuration to respond to changing environmental conditions and task requirements. Allowing robots to exploit this flexibility autonomously does not only lead to a better resource usage and reduced personnel for remote operation of these systems. It also offers the basis for new operation concepts concerning multi-agent teams. Hence, the results of this thesis are relevant for pure robot teams as well as mixed human-robot teams.

Section 7.1 summarises the findings and Section 7.2 the essential lessons learnt in this thesis. Section 7.3 provides an outlook and suggests related and complementary research activities as possible contributions to more autonomous and adaptive multi-robot systems of the future.

7.1 Thesis Summary

Increasing the ability of robotic systems to adapt is attractive for many application areas. Even more so, when robots can autonomously adapt. Therefore, enabling autonomous, context-dependant adaptivity is one of the general goals in the area of robotics research. Meanwhile, the degree of adaptation of robotic systems is most often limited to specialised domains and use cases, where single and monolithic robots dominate the research landscape. Achieving human-like adaptivity for robotic systems remains an unsolved challenge.

To work towards this goal and complementary to existing research, this thesis introduces an approach to autonomously exploit morphology changes in a reconfigurable multi-robot system. This thesis deals with a mixture of capable, but still modular systems. Capable reconfigurable heterogeneous robots and simple robots which are unable to physically interconnect can be

mixed in the suggested planning approach. Resources can be physically moved within a reconfigurable multi-robot system, so that a resource can be shifted to where it is needed the most. Hence, a continuous optimisation of a robotic system's properties - including functional as well as non-functional properties - is feasible. In effect, functional and non-functional properties can be actively modulated by changing the hardware structure. Therefore, according to the importance of activities, a multi-robot system can adapt its coalition structure to provide optimal support. This thesis has selected efficacy, efficiency, and safety as main optimisation objectives. Efficacy is a functional property while efficiency and safety are two interrelated, non-functional properties. They are selected with respect to the aspired application of reconfigurable multi-robot systems in space missions.

The main research objective of this thesis is the autonomous operation of reconfigurable multi-robot systems. This thesis has worked towards this objective by introducing a formalisation for dealing with a reconfigurable multi-robot system based on game and set theory. The organisation model MoreOrg implements this formalisation.

MoreOrg uses an ontological description of agents to enable modelling as well as reasoning. The ontological description is based on international standards to maintain an open and dynamically extensible system architecture. Furthermore, it allows scaling the approach to larger multi-robot systems in future works. The representation of atomic agents and their resource dependency structure is fundamental to the reasoning approach. The hierarchical resource decomposition scheme and a resource summation rule apply to atomic and composite agents. In combination with a definition of interface compatibility, an identification of structurally feasible and functionally suitable agents has been enabled. A mapping between an agent's functionality and an agent's resource structure is the basis for this identification.

Reasoning with reconfigurable multi-robot systems suffers from a combinatorial challenge. Therefore, MoreOrg introduces minimal agent types and the so-called functional saturation bound to avoid the handling of highly redundant agent types. In practice, MoreOrg significantly lowers the computational complexity when reasoning with composite agents, although the general worst case complexity remains.

MoreOrg is basis of the mission planner TemPl. A mission is defined as a mixture of functionality and resource requirements in combination with spatial and temporal synchronisation constraints. The developed planning approach accounts for activity synchronisation for a reconfigurable multi-robot systems. Thereby, it generalises requirements of the VRP with Trucks and Trailers. The mission planner does not perform general purpose action planning. Instead, it establishes a resource allocation and transition schedule, which is open to further enhancement and can serve as basis for detailed action planning. TemPl uses a constraint-based global search to generate a solution candidate. It validates and optimises this candidate solution with a local search. This combination allows for an effective search process for reconfigurable multi-robot systems.

This thesis presented the practical application and evaluation of a reconfigurable multi-robot system in two relevant application scenarios: one outdoor scenario to illustrate the general operability of a multi-robot system, and one indoor scenario to prove the autonomous reconfigurability. The specially developed infrastructure enables the full distribution of a multi-robot system and introduces the use of existing multi-agent standards to the robotics community. The evaluation highlights practical issues that are encountered by any application of reconfigurable multi-robot systems. Furthermore, the evaluation considers the isolated application of

uncooperative reconfiguration manoeuvres, as well as the application of cooperative reconfiguration in the context of an exemplary sample return mission.

This thesis outlines a successful strategy to operate reconfigurable multi-robot systems autonomously. The strategy comprises a feasible mission planning approach and a corresponding control approach evaluated with real robotic systems. Mission planning is using an exemplary quantification of safety. Thereby, it shows how safety or rather probability of survival can be considered for optimisation in multi-robot missions. In practice, improving safety in reconfigurable multi-robot systems depends on initially contradictory means. For example, reconfigurable multi-robot systems require additional software and standardised hardware interfaces to manage reconfiguration. Thereby, points of failures are added to the system. Hence, an increase of flexibility comes at the cost of increase of system complexity and a potential decrease in the quality of system performance. Industrial grade development processes can, however, increase the quality and reliability of components systematically. Furthermore, modularisation grants better maintainability. A serial production of EMIs and payload items can therefore mitigate the risk of failure. Nevertheless, the need for flexibility as basis for adaption has to be traded against an increase in system complexity compared to a set of monolithic robots.

In summary, this thesis outlines a unique approach to automating reconfigurable multi-robot systems by taking advantage of and bringing together research from organisation theory, knowledge engineering, operations research, planning, and robotics. This thesis brings theoretical elements together with the practical experience and issues arising when realising complex adaptive multi-robot systems.

7.2 Lessons Learned

The design and implementation of an organisation model has been the basis for automated planning for reconfigurable multi-robot systems. The approach has been based on an ontological description. Although the use of an ontology is not strictly necessary, an application of semantic technologies enables knowledge-based reasoning and serves as a standardised and extensible basis for the overall approach. Hence, it is a key element for an open extensible, scalable reconfigurable multi-robot system. In general, standardisation enables interoperation in multi-robot systems and is essential for reconfigurable systems.

A system designer has to make a careful choice between using an existing or implementing a new standard. Standards allow for extensibility, however, they come with an overhead as result of abstraction and might affect the performance of an application. A trade-off between performance and (re)usability has to be made. Furthermore, any standard requires a learning curve. Application specific developments are typically easier to handle by developers, since they are the originators. However, extensibility has to be thought also in terms of maintenance of such systems and in these cases established or commonly applied standards should be preferred over ad-hoc development solutions, despite a potential better runtime performances.

Exploiting modularisation is the key driver of using reconfigurable multi-robot systems. To exploit the flexibility of a reconfigurable multi-robot system standardisation should not be limited to hardware interface specifications, but include subcomponent structures and subsystem behaviour or rather functionality. At each level modularisation and typing of modules (software and hardware alike) should follow Liskov's substitution principle (Liskov and Wing

1994), so that same type modules can be freely exchanged while maintaining functionality. While modularisation increases the flexibility for operation, it demands at the same time more complex solutions to exploit this flexibility resulting in additional point of failures. For instance, the essential role of the EMI demands risk management to focus on this element with priority. Corresponding quality assurance processes have to be established to achieve a maximum level of reliability.

The availability of reliant individual modules is clearly important. Nevertheless, intelligent system composition and the active use of superadditive functionality offer a method to compensate for some deficiencies of individual modules. However, composition can lead to unforeseen negative effects and an exhaustive testing of agent coalitions might be practically infeasible. In practice, however, at least for relevant and prioritised agent coalitions superadditive functionality has to be validated and evaluated.

Reconfigurable multi-robot systems offer to exploit existing redundancies and can thereby increase the probability of success for robotic missions. The final evaluation of an autonomous operation for this thesis has only shown a low success rate. This illustrates the challenges that still have to be coped with until a robotic mission can really benefit from an application of reconfigurable systems. The main reason for the low success rate can be found in an insufficient tolerance against local failures or local deviations. Action planning uses preconditions, prevail conditions and post conditions to model the problem domain. Similar models should be used for the execution layer. Failure handling has to make sure that prevail conditions are maintained and pre and post conditions can be either established or that human operators (or other agents) are interactively requested to provide intermediate support. In addition, local failure handling routines and reconfiguration activities should be modelled as templates for reoccurring activity and cooperation patterns.

The development of the planning system TemPl has illustrated the combinatorial challenge at different levels when using reconfigurable multi-agent systems. Firstly, as part of the organisation model to initially identify the number of agent systems that are suited to perform a set of functionalities. Secondly, as part of planning for coalition structure changes to perform a robotic mission. Especially the careful incremental selection and optimisation of the embedded algorithms in the organisation model and the planning algorithm, starting from reducing the generation time for combinations with types to reducing solution symmetry, but particularly the algorithms to identify agent connectivity and functionality support in a coalition structure have led to the current planner's performance.

7.3 Outlook

This thesis illustrates one feasible approach for the autonomous operation of reconfigurable multi-robot systems. Clearly, multi-robot systems with a high-degree of interconnection and interdependence can directly benefit from the presented solutions. Although this thesis' goal is narrowed to robotic systems and a space application, the results of this thesis have a broader relevance. The Internet of Things, for example, shares many properties with reconfigurable multi-robot systems. It operates, however, close to swarm-based systems and combines physical as well as non-physical agents. It therefore offers further opportunities to exploit and study reconfigurability and resulting superaddition.

Many detailed challenges remain to exploit the full flexibility of reconfigurable systems. Therefore, this thesis hopes to initiate further research on the autonomous operation of reconfigurable multi-agent and multi-robot systems in general. Generally, the organisation model and planning approach in this thesis are based on several assumptions, similar to the deterministic classical planning. A removal of these assumptions should be targeted by future research activities. Furthermore, the following subsections suggest improvements of the existing approach and complementary research directions.

7.3.1 Organisation Modelling

The application of an organisation model can be improved and its relevance for (multi-)robot application increased in various ways. Firstly, the organisation modelling can be extended with a geometrical description of resources or rather atomic agents. Such a description should not be limited to a simple outer shape description. Much more required is the detailed description of coordinate transformations for actuators, sensors and EMIs and additionally, a specification of the resulting workspace of kinematic chains or the field of view of visual sensors. The availability of this information and accessibility for all members of a multi-robot systems permits rich interaction schemas. For instance, cooperative agents can request an image of a target location from a certain perspective - to get sensor data from the requesting agent's blind spots. Furthermore, external guidance for cooperative reconfiguration will be feasible if this data is available. Especially the development of a standardised approach is required to achieve a benefit.

The organisation model can also be augmented with an experience database. This experience database could be used to estimate the probability of survival of components and to classify the quality of functionalities - similar to the OMACS approach. Collecting experience of action activities can help to improve the identification of the robot which is best suited to provide a functionality. The collection of experience has to track not only the performance a single agent, but also that of coalitions and can thereby be able to estimate or learn possible negative effects. This thesis uses probability of survival to characterise mission safety. It should, however, be understood as a placeholder for more sophisticated safety characterisation. For instance time-dependant degradation functions will provide more realistic optimisation criteria to the planning system.

This thesis handles superaddition with an optimistic assumption towards joining two or more systems. In reality, negative effects might arise when agent coalitions are formed or as result of a failing subsystem. Geometrical constraints can restrict the functionality of a composite agent, block the field of view of sensors or exceed an admissible weight for the locomotion platform. Increased structural stress and wear of the system might be consequences of joining two agents. Further implications are a loss of the stability of mechanical structures or inaccuracies of calibrated sensors. The current organisation model has been prepared for negative effects with the min/max cardinality based modelling schema. However, it has not been evaluated in an applied approach. Hence, researching practical modelling approaches to embed negative effects is required.

7.3.2 Planning

The current planning approach starts by identifying the minimal resource requirements to solve a mission. Instead of planning directly with high redundancies, the usage of a minimal set of resources has been motivated by three observations: (a) maximum efficacy for a mission is reached as soon as a valid resource assignment has been found, (b) planning complexity rises exponentially with the number of resources involved, and (c) a minimal assignment can still be followed by an augmentation process to establish higher redundancy. The latter observation should trigger research into augmentation strategies for missions, so that all available resources are considered for strengthening the execution of a mission. Further approaches can still consider improving resource redundancies or alternative safety characteristics at an earlier planning stage.

The scalability of the current approach is limited due the use of the temporally-expanded network and application of linear programming for local optimisation. Hence, the planning approach requires further improvement, e.g., with the use of heuristic search techniques. Alternatively, the search process can be based on a VLNS strategy and destroy-repair techniques which are not applied on the actual transport network but only to the constraint-based mission specification. This could lead to a neighbourhood search at (meta-)constraint level. Note that the current mission representation already provides a foundation for such an approach by permitting to convert a solution into an equivalent constraint-based description using the (*all*) *equal* constraint (see Table 4.5).

Increasing the expressiveness of the mission specification by the introduction of additional (meta-)constraints will allow a fine-grained control of the mission design. The current mission specification does not account explicitly for transition constraints between two locations. Transition constraints can in some cases be modelled by the addition of at least two spatio-temporal constraints, relating to the start and the end of a transition. The explicit consideration of edge constraints, however, can combine resource requirements with temporally dependant transfer requirements, e.g., when a local transition is temporally limited.

A major motivation of developing an automated planning approach has been the safe(r) execution of multi-robot missions. The presented approach, however, only looks at a single solution without accounting for potential neighbouring solutions. Further research can evaluate the resilience of solutions by simulating agent outages and resulting mission repair cost. Failure handling in the form of replanning with less available resources could be used to identify performance impacts and further characterise the cost of alternative solutions.

The developed planning approach can further be part of meta-search to optimise missions. This search can evaluate different types and constellations of robotic agents and compare the mission characteristics. This might help to develop novel mission approaches.

7.3.3 Infrastructure

The existing infrastructure currently lacks a tight integration with the organisation model and the planning system. Future efforts could use a distributed database for the management of the organisation model, reflecting the distributed nature of reconfigurable systems. The current implementation already accounts for a modular description of a reconfigurable multi-robot team, but this information has to be accessible for and from each agent in the team. Hence,

a distributed database offers a basis for the dynamic, plug-and-play like extensibility of a reconfigurable multi-robot system. Note that the current infrastructure already provides the necessary communication infrastructure, although a content language has still to be defined.

7.3.4 Application of Real Reconfigurable Multi-Robot Systems

The design of general reconfiguration control patterns and algorithms, including dynamic failure handling routines still poses a significant engineering challenge. Fast, yet reliable coalition structure changes are required to improve the applicability of reconfigurable multi-robot systems. Therefore, strategies for increasing the robustness of control approaches have to be investigated. For example, the estimation of a relative pose between two robots could be supported by redundant measurements when multiple agents are available.

The planning and execution layers have to be coupled, so that a dynamic planning approach can be supported. Furthermore, any application of a plan leaves room for further online optimisation strategies. An example: two agents will be aware that they have to meet at a target location to exchange an attached atomic agent. A local online optimisation strategy suggests a better meeting point to save time and energy. Then both systems temporally and locally change their plan, while both agents guarantee that they maintain the preconditions of their next assignments.

Appendix **A**

Robots: Atomic Agents

A.1 Atomic agent properties

The following Tables A.1, A.2, and A.3 list the static properties of the atomic agents that have been operated and experimented with throughout the course of this thesis. Chapter 2 introduces the robotic systems and their application scope. Chapter 3 introduces the organisation model and for evaluation in this thesis atomic agent types with the following properties are considered:

Table A.1: *Properties of agent types.*

Property	BaseCamp	CREX	CoyoteIII
mass (kg)	3.00	27.00	15.00
supply voltage (V)	48.00	50.00	48.00
energy capacity (Ah)	0.00	2.4	7.0
nominal power consumption (W)	0.00	100.00	50.00
min acceleration (m/s)	0.00	0.00	0.00
max acceleration (m/s)	0.00	0.10	1.00
nominal acceleration (m/s)	0.00	0.00	0.00
min velocity (m/s)	0.00	0.00	0.00
min velocity (m/s)	0.00	0.30	0.30
nominal velocity (m/s)	0.00	0.10	0.20
transport consumption (units)	1	1	1
transport capacity (units)	0	2	4
mobility	\neg mobile	mobile	mobile

Table A.2: *Properties of agent types.*

Property	Payload	PayloadSoilSampler	PayloadCamera
mass (kg)	3.00	3.50	4.00
supply voltage (V)	48.00	48.00	48.00
energy capacity (Ah)	0.00	0.00	0.00
nominal power consumption (W)	25.00	30.00	50.00
min acceleration (m/s)	0.00	0.00	0.00
max acceleration (m/s)	0.00	0.00	0.00
nominal acceleration (m/s)	0.00	0.00	0.00
min velocity (m/s)	0.00	0.00	0.00
min velocity (m/s)	0.00	0.00	0.00
nominal velocity (m/s)	0.00	0.00	0.00
transport consumption (units)	1	1	1
transport capacity (units)	0	0	0
mobility	\neg mobile	\neg mobile	\neg mobile

Table A.3: *Properties of agent types.*

Property	PayloadBattery	Sherpa	SherpaTT
mass (kg)	5.00	160.00	160.00
supply voltage (V)	48.00	48.00	48.00
energy capacity (Ah)	2.4	10.00	10.00
nominal power consumption (W)	0.80	120.00	120.00
min acceleration (m/s)	0.00	0.00	0.00
max acceleration (m/s)	0.00	0.10	0.10
nominal acceleration (m/s)	0.00	0.00	0.00
min velocity (m/s)	0.00	0.00	0.00
min velocity (m/s)	0.00	1.00	1.00
nominal velocity (m/s)	0.00	0.50	0.50
transport consumption (units)	1	1	1
transport capacity (units)	0	10	10
mobility	\neg mobile	mobile	mobile

A.2 Classification of electromechanical interfaces

The existing classification approaches for modular reconfigurable systems consider only few available characteristics of hardware interface. However, for any generic and broader application, the design of electromechanical interfaces will be a key factor. Hence, a more sophisticated interface classification approach is required. Table A.4 suggests a set of properties for consideration for novel classification approaches.

Table A.4: *Suggested classification scheme for interfaces in reconfigurable multi-robot systems using the DFKI EMI, ATRON and iBOSS as illustration examples.*

Property	EMI	ATRON	iBOSS
embeddable	yes	no	yes
data connection	yes	yes	yes
power connection	yes	no	yes
thermal connection	no	no	yes
mechanical load	600 N	n/a	n/a
compatibility	gendered	gendered	androgynous
rotational invariance	90 deg	no	90 deg
redundancy	yes	no	yes
form factor	cuboid	ball	cylinder
dimensions (approx.)	surface: 0.15 m × 0.15 m depth 0.06 m	Ø 0.11 m	Ø 0.16 m depth: 0.254 m
weight	n/a	n/a	n/a
active degree of freedom	0	1	0
visual markers	yes	no	no
mechanical guidance	yes	no	no

Appendix **B**

Ontologies

The following listings show the set of ontologies that have been used in this thesis.

B.1 Base Ontology

Listing B.1: *Base Ontology.*

```
1 @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
2 @prefix : <http://www.rock-robotics.org/2014/01/om-schema#> .
3 @prefix owl: <http://www.w3.org/2002/07/owl#> .
4 @prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
5 @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
6
7 <http://www.rock-robotics.org/2014/01/om-base#>
8   a owl:Ontology .
9
10 <http://www.rock-robotics.org/2014/01/om-base#Agent>
11   a owl:Class ;
12   rdfs:subClassOf :Resource ;
13   owl:equivalentClass :Actor .
14
15 :Actor
16   a owl:Class, owl:NamedIndividual ;
17   rdfs:comment "An actor is a physical or logical entity that can act, i.e.
18     perform actions which have an effect" ;
19   rdfs:subClassOf :Resource .
20
21 :Capability
22   a owl:Class, owl:NamedIndividual ;
23   rdfs:comment "Capabilities represent 'soft' resources that need to be available
24     to provide services or perform actions" ;
25   rdfs:subClassOf :Functionality .
```

```

24
25 :ElectricalInterface
26     a owl:Class ;
27     rdfs:subClassOf :Interface .
28
29 :ElectroMechanicalInterface
30     a owl:Class ;
31     rdfs:subClassOf :ElectricalInterface, :MechanicalInterface .
32
33 :Functionality
34     a owl:Class ;
35     rdfs:subClassOf :Resource .
36
37 :Interface
38     a owl:Class, owl:NamedIndividual ;
39     rdfs:subClassOf :Resource .
40
41 :MechanicalInterface
42     a owl:Class ;
43     rdfs:subClassOf :Interface .
44
45 :PhysicalEntity
46     a owl:Class ;
47     rdfs:subClassOf :Resource .
48
49 :Resource
50     a owl:Class, owl:NamedIndividual .
51
52 :Service
53     a owl:Class, owl:NamedIndividual ;
54     rdfs:comment ""A service is an offer to potential consumers.
55
56 There is always only one instance of a service an actor can provide, e.g,
57 a StereoImageProvider service cannot depends upon two instances of ImageProvider but
58     need to associated directly with two camera"" ;
59     rdfs:subClassOf :Functionality .
60
61 :compatibleWith
62     a owl:ObjectProperty, owl:SymmetricProperty .
63
64 :dependsOn
65     a owl:ObjectProperty, owl:TransitiveProperty .
66
67 :energyCapacity
68     a owl:DatatypeProperty, owl:FunctionalProperty ;
69     rdfs:range xsd:double ;
70     rdfs:subPropertyOf :energyProperty .
71
72 :energyProperty
73     a owl:DatatypeProperty ;
74     rdfs:domain :Actor ;
75     rdfs:range xsd:double ;
76     rdfs:subPropertyOf :physicalProperty ;
77     owl:propertyDisjointWith :locomotionProperty, :safetyProperty .
78
79 :fulfills
80     a owl:ObjectProperty .
81
82 :has

```

```

82     a owl:ObjectProperty, owl:TransitiveProperty .
83
84 :hasTransportCapacity
85     a owl:ObjectProperty, owl:TransitiveProperty .
86
87 :locomotionProperty
88     a owl:DatatypeProperty ;
89     rdfs:domain :Actor ;
90     rdfs:range xsd:double ;
91     rdfs:subPropertyOf :physicalProperty ;
92     owl:propertyDisjointWith :safetyProperty .
93
94 :mass
95     a owl:DatatypeProperty, owl:FunctionalProperty ;
96     rdfs:range xsd:double ;
97     rdfs:subPropertyOf :physicalProperty .
98
99 :maxAcceleration
100     a owl:DatatypeProperty, owl:FunctionalProperty ;
101     rdfs:range xsd:double ;
102     rdfs:subPropertyOf :locomotionProperty .
103
104 :maxEnergyCapacity
105     a owl:DatatypeProperty, owl:FunctionalProperty ;
106     rdfs:range xsd:double ;
107     rdfs:subPropertyOf :energyProperty .
108
109 :maxHeight
110     a owl:DatatypeProperty, owl:FunctionalProperty ;
111     rdfs:range xsd:double ;
112     rdfs:subPropertyOf :physicalProperty .
113
114 :maxVelocity
115     a owl:DatatypeProperty, owl:FunctionalProperty ;
116     rdfs:range xsd:double ;
117     rdfs:subPropertyOf :locomotionProperty .
118
119 :maxWidth
120     a owl:DatatypeProperty, owl:FunctionalProperty ;
121     rdfs:range xsd:double ;
122     rdfs:subPropertyOf :physicalProperty .
123
124 :minAcceleration
125     a owl:DatatypeProperty, owl:FunctionalProperty ;
126     rdfs:range xsd:double ;
127     rdfs:subPropertyOf :locomotionProperty .
128
129 :minHeight
130     a owl:DatatypeProperty, owl:FunctionalProperty ;
131     rdfs:range xsd:double ;
132     rdfs:subPropertyOf :physicalProperty .
133
134 :minVelocity
135     a owl:DatatypeProperty, owl:FunctionalProperty ;
136     rdfs:range xsd:double ;
137     rdfs:subPropertyOf :locomotionProperty .
138
139 :minWidth
140     a owl:DatatypeProperty, owl:FunctionalProperty ;

```

```
141     rdfs:range xsd:double ;
142     rdfs:subPropertyOf :physicalProperty .
143
144 :modelledBy
145     a owl:ObjectProperty, owl:TransitiveProperty ;
146     rdfs:domain :Resource ;
147     owl:inverseOf :models .
148
149 :models
150     a owl:ObjectProperty ;
151     rdfs:range :Resource .
152
153 :nominalAcceleration
154     a owl:DatatypeProperty ;
155     rdfs:range xsd:double ;
156     rdfs:subPropertyOf :locomotionProperty .
157
158 :nominalHeight
159     a owl:DatatypeProperty, owl:FunctionalProperty ;
160     rdfs:range xsd:double ;
161     rdfs:subPropertyOf :physicalProperty .
162
163 :nominalPowerConsumption
164     a owl:DatatypeProperty, owl:FunctionalProperty ;
165     rdfs:range xsd:double ;
166     rdfs:subPropertyOf :energyProperty .
167
168 :nominalVelocity
169     a owl:DatatypeProperty, owl:FunctionalProperty ;
170     rdfs:range xsd:double ;
171     rdfs:subPropertyOf :locomotionProperty .
172
173 :nominalWidth
174     a owl:DatatypeProperty, owl:FunctionalProperty ;
175     rdfs:range xsd:double ;
176     rdfs:subPropertyOf :physicalProperty .
177
178 :physicalProperty
179     a owl:DatatypeProperty ;
180     rdfs:domain :Actor ;
181     rdfs:range xsd:double .
182
183 :probabilityOfFailure
184     a owl:DatatypeProperty, owl:FunctionalProperty ;
185     rdfs:range xsd:double ;
186     rdfs:subPropertyOf :safetyProperty .
187
188 :provides
189     a owl:ObjectProperty, owl:TransitiveProperty .
190
191 :safetyProperty
192     a owl:DatatypeProperty ;
193     rdfs:domain :Actor ;
194     rdfs:range xsd:double .
195
196 :supplyVoltage
197     a owl:DatatypeProperty, owl:FunctionalProperty ;
198     rdfs:range xsd:double ;
199     rdfs:subPropertyOf :energyProperty .
```

```

200
201 :transportCapacity
202     a owl:DatatypeProperty, owl:FunctionalProperty ;
203     rdfs:range xsd:double ;
204     rdfs:subPropertyOf :transportProperty .
205
206 :transportDemand
207     a owl:DatatypeProperty, owl:FunctionalProperty ;
208     rdfs:range xsd:double ;
209     rdfs:subPropertyOf :transportProperty .
210
211 :transportProperty
212     a owl:DatatypeProperty ;
213     rdfs:range xsd:double ;
214     rdfs:subPropertyOf :physicalProperty .
215
216 :uses
217     a owl:ObjectProperty, owl:TransitiveProperty .

```

B.2 Application-specific Base Ontology

Listing B.2: *Extended Base Ontology.*

```

1 @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
2 @prefix : <http://www.rock-robotics.org/2014/01/om-schema#> .
3 @prefix owl: <http://www.w3.org/2002/07/owl#> .
4 @prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
5 @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
6
7 <http://www.rock-robotics.org/2014/01/om-schema#>
8     a owl:Ontology ;
9     owl:imports <http://www.rock-robotics.org/2014/01/om-base#> .
10
11 :Camera
12     a owl:Class ;
13     rdfs:subClassOf :PhysicalEntity .
14
15 :EmiActive
16     :compatibleWith :EmiPassive ;
17     a :EmiActive, owl:Class, owl:NamedIndividual ;
18     rdfs:subClassOf :ElectroMechanicalInterface ;
19     owl:disjointWith :EmiPassive .
20
21 :EmiPassive
22     a :EmiPassive, owl:Class, owl:NamedIndividual ;
23     rdfs:subClassOf :ElectroMechanicalInterface .
24
25 :EmiPowerProvider
26     a owl:Class ;
27     rdfs:subClassOf :Service, [
28         a owl:Restriction ;
29         owl:minQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
30         owl:onClass :PowerSource ;
31         owl:onProperty :has
32     ], [
33         a owl:Restriction ;
34         owl:minQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
35         owl:onClass :ElectroMechanicalInterface ;

```

```

36         owl:onProperty :has
37     ] .
38
39 :ForceTorqueSensor
40     a owl:Class ;
41     rdfs:subClassOf :PhysicalEntity .
42
43 :ImageProvider
44     a owl:Class ;
45     rdfs:subClassOf :Service, [
46         a owl:Restriction ;
47         owl:minQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
48         owl:onClass :PowerSource ;
49         owl:onProperty :has
50     ], [
51         a owl:Restriction ;
52         owl:minQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
53         owl:onClass :Camera ;
54         owl:onProperty :has
55     ] .
56
57 :LaserScanner
58     a owl:Class ;
59     rdfs:subClassOf :PhysicalEntity .
60
61 :Link
62     a owl:Class ;
63     rdfs:subClassOf :Resource, [
64         a owl:Restriction ;
65         owl:onClass :Interface ;
66         owl:onProperty :has ;
67         owl:qualifiedCardinality "2"^^xsd:nonNegativeInteger
68     ] .
69
70 :Localization
71     a owl:Class ;
72     rdfs:subClassOf :Capability .
73
74 :Location
75     a owl:Class ;
76     rdfs:subClassOf :Resource .
77
78 :LocationImageProvider
79     a owl:Class ;
80     rdfs:subClassOf :Service, [
81         a owl:Restriction ;
82         owl:minQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
83         owl:onClass :MoveTo ;
84         owl:onProperty :has
85     ], [
86         a owl:Restriction ;
87         owl:minQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
88         owl:onClass :ImageProvider ;
89         owl:onProperty :has
90     ] .
91
92 :Locomotion
93     a owl:Class ;
94     rdfs:subClassOf :Capability .

```

```

95
96 :LogisticHubProvider
97     a owl:Class ;
98     rdfs:subClassOf :Service .
99
100 :Manipulation
101     a owl:Class ;
102     rdfs:subClassOf :Capability .
103
104 :ManipulationProvider
105     a owl:Class ;
106     rdfs:subClassOf :Service, [
107         a owl:Restriction ;
108         owl:minQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
109         owl:onClass :Manipulator ;
110         owl:onProperty :has
111     ], [
112         a owl:Restriction ;
113         owl:minQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
114         owl:onClass :Manipulation ;
115         owl:onProperty :has
116     ] .
117
118 :Manipulator
119     a owl:Class ;
120     rdfs:subClassOf :PhysicalEntity .
121
122 :Mapping
123     a owl:Class ;
124     rdfs:subClassOf :Capability .
125
126 :Mission
127     a owl:Class ;
128     rdfs:subClassOf :Resource .
129
130 :MoveTo
131     a owl:Class ;
132     rdfs:subClassOf :Capability, [
133         a owl:Restriction ;
134         owl:minQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
135         owl:onClass :Mapping ;
136         owl:onProperty :has
137     ], [
138         a owl:Restriction ;
139         owl:minQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
140         owl:onClass :Localization ;
141         owl:onProperty :has
142     ], [
143         a owl:Restriction ;
144         owl:minQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
145         owl:onClass :PowerSource ;
146         owl:onProperty :has
147     ], [
148         a owl:Restriction ;
149         owl:minQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
150         owl:onClass :Locomotion ;
151         owl:onProperty :has
152     ] .
153

```

```

154 :PayloadLogisticHub
155     a owl:Class ;
156     rdfs:subClassOf :Service, [
157         a owl:Restriction ;
158         owl:minQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
159         owl:onClass :EmiPassive ;
160         owl:onProperty :has
161     ], [
162         a owl:Restriction ;
163         owl:minQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
164         owl:onClass :LogisticHubProvider ;
165         owl:onProperty :has
166     ] .
167
168 :PowerSource
169     a owl:Class ;
170     rdfs:subClassOf :PhysicalEntity .
171
172 :Requirement
173     a owl:Class .
174
175 :SoilSampler
176     a owl:Class ;
177     rdfs:subClassOf :PhysicalEntity .
178
179 :SoilSamplingProvider
180     a owl:Class ;
181     rdfs:subClassOf :Service, [
182         a owl:Restriction ;
183         owl:minQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
184         owl:onClass :Manipulator ;
185         owl:onProperty :has
186     ], [
187         a owl:Restriction ;
188         owl:minQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
189         owl:onClass :PowerSource ;
190         owl:onProperty :has
191     ], [
192         a owl:Restriction ;
193         owl:minQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
194         owl:onClass :SoilSampler ;
195         owl:onProperty :has
196     ], [
197         a owl:Restriction ;
198         owl:minQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
199         owl:onClass :ForceTorqueSensor ;
200         owl:onProperty :has
201     ], [
202         a owl:Restriction ;
203         owl:minQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
204         owl:onClass :Manipulation ;
205         owl:onProperty :has
206     ] .
207
208 :StereoImageProvider
209     a owl:Class ;
210     rdfs:subClassOf :Service, [
211         a owl:Restriction ;
212         owl:minQualifiedCardinality "2"^^xsd:nonNegativeInteger ;

```



```

213         owl:onClass :Camera ;
214         owl:onProperty :has
215     ], [
216         a owl:Restriction ;
217         owl:minQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
218         owl:onClass :PowerSource ;
219         owl:onProperty :has
220     ] .
221
222 :TransportProvider
223     a owl:Class ;
224     rdfs:subClassOf :Service, [
225         a owl:Restriction ;
226         owl:minQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
227         owl:onClass :ElectroMechanicalInterface ;
228         owl:onProperty :has
229     ], [
230         a owl:Restriction ;
231         owl:minQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
232         owl:onClass :MoveTo ;
233         owl:onProperty :has
234     ] .

```

B.3 Robot Ontologies

Listing B.3: *Coyote III Ontology.*

```

1 @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
2 @prefix : <http://www.rock-robotics.org/2015/12/robots/CoyoteIII#> .
3 @prefix owl: <http://www.w3.org/2002/07/owl#> .
4 @prefix om-schema: <http://www.rock-robotics.org/2014/01/om-schema#> .
5 @prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
6 @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
7 @prefix om-base: <http://www.rock-robotics.org/2014/01/om-base#> .
8
9 om-schema:CoyoteIII
10     om-schema:energyCapacity 7.0 ;
11     om-schema:mass 15.0 ;
12     om-schema:maxAcceleration 1.0 ;
13     om-schema:maxEnergyCapacity 259200 ;
14     om-schema:maxHeight 0.3 ;
15     om-schema:maxVelocity 0.3 ;
16     om-schema:maxWidth 0.5 ;
17     om-schema:minAcceleration 0.0 ;
18     om-schema:minHeight 0.3 ;
19     om-schema:minVelocity 0.0 ;
20     om-schema:minWidth 0.5 ;
21     om-schema:nominalAcceleration 0.0 ;
22     om-schema:nominalHeight 0.3 ;
23     om-schema:nominalPowerConsumption 50.0 ;
24     om-schema:nominalVelocity 0.2 ;
25     om-schema:nominalWidth 0.5 ;
26     om-schema:probabilityOfFailure 0.5 ;
27     om-schema:supplyVoltage 48.0 ;
28     om-schema:transportCapacity "4"^^xsd:nonNegativeInteger ;
29     om-schema:transportDemand "1"^^xsd:nonNegativeInteger ;
30     a om-schema:CoyoteIII, owl:Class, owl:NamedIndividual ;
31     rdfs:subClassOf om-schema:Actor, [

```

```

32     a owl:Restriction ;
33     owl:maxQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
34     owl:onClass om-schema:Mapping ;
35     owl:onProperty om-schema:has
36 ], [
37     a owl:Restriction ;
38     owl:maxQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
39     owl:onClass om-schema:Camera ;
40     owl:onProperty om-schema:has
41 ], [
42     a owl:Restriction ;
43     owl:maxQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
44     owl:onClass om-schema:LaserScanner ;
45     owl:onProperty om-schema:has
46 ], [
47     a owl:Restriction ;
48     owl:maxQualifiedCardinality "2"^^xsd:nonNegativeInteger ;
49     owl:onClass om-schema:EmiPassive ;
50     owl:onProperty om-schema:has
51 ], [
52     a owl:Restriction ;
53     owl:maxQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
54     owl:onClass om-schema:Localization ;
55     owl:onProperty om-schema:has
56 ], [
57     a owl:Restriction ;
58     owl:maxQualifiedCardinality "4"^^xsd:nonNegativeInteger ;
59     owl:onClass om-schema:Actor ;
60     owl:onProperty om-schema:hasTransportCapacity
61 ], [
62     a owl:Restriction ;
63     owl:maxQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
64     owl:onClass om-schema:Power ;
65     owl:onProperty om-schema:has
66 ], [
67     a owl:Restriction ;
68     owl:maxQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
69     owl:onClass om-schema:Locomotion ;
70     owl:onProperty om-schema:has
71 ] .

```

Listing B.4: Base Camp Ontology.

```

1 @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
2 @prefix : <http://www.rock-robotics.org/2015/12/robots/BaseCamp#> .
3 @prefix owl: <http://www.w3.org/2002/07/owl#> .
4 @prefix om-schema: <http://www.rock-robotics.org/2014/01/om-schema#> .
5 @prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
6 @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
7 @prefix om-base: <http://www.rock-robotics.org/2014/01/om-base#> .
8
9 om-schema:BaseCamp
10   om-schema:energyCapacity 0.0 ;
11   om-schema:mass 3.0 ;
12   om-schema:maxHeight 0.3 ;
13   om-schema:maxWidth 0.5 ;
14   om-schema:minHeight 0.3 ;
15   om-schema:minWidth 0.5 ;
16   om-schema:nominalHeight 0.3 ;

```

```

17   om-schema:nominalPowerConsumption 0.0 ;
18   om-schema:nominalWidth 0.5 ;
19   om-schema:supplyVoltage 48.0 ;
20   om-schema:transportCapacity "0"^^xsd:nonNegativeInteger ;
21   om-schema:transportDemand "1"^^xsd:nonNegativeInteger ;
22   a om-schema:BaseCamp, owl:Class, owl:NamedIndividual ;
23   rdfs:subClassOf om-schema:Actor, [
24     a owl:Restriction ;
25     owl:maxQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
26     owl:onClass om-schema:LogisticHubProvider ;
27     owl:onProperty om-schema:has
28   ], [
29     a owl:Restriction ;
30     owl:maxQualifiedCardinality "6"^^xsd:nonNegativeInteger ;
31     owl:onClass om-schema:EmiPassive ;
32     owl:onProperty om-schema:has
33   ], [
34     a owl:Restriction ;
35     owl:maxQualifiedCardinality "0"^^xsd:nonNegativeInteger ;
36     owl:onClass om-schema:Actor ;
37     owl:onProperty om-schema:hasTransportCapacity
38   ] .

```

Listing B.5: CREX Ontology.

```

1  @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
2  @prefix : <http://www.rock-robotics.org/2015/12/robots/CREX#> .
3  @prefix owl: <http://www.w3.org/2002/07/owl#> .
4  @prefix om-schema: <http://www.rock-robotics.org/2014/01/om-schema#> .
5  @prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
6  @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
7  @prefix om-base: <http://www.rock-robotics.org/2014/01/om-base#> .
8
9  om-schema:CREX
10   om-schema:energyCapacity 2.4 ;
11   om-schema:mass 27.0 ;
12   om-schema:maxAcceleration 0.1 ;
13   om-schema:maxEnergyCapacity 2.4 ;
14   om-schema:maxHeight 4.0 ;
15   om-schema:maxVelocity 0.3 ;
16   om-schema:maxWidth 1.5 ;
17   om-schema:minAcceleration 0.0 ;
18   om-schema:minHeight 0.25 ;
19   om-schema:minVelocity 0.0 ;
20   om-schema:minWidth 1.0 ;
21   om-schema:nominalAcceleration 0.0 ;
22   om-schema:nominalHeight 0.5 ;
23   om-schema:nominalPowerConsumption 100.0 ;
24   om-schema:nominalVelocity 0.1 ;
25   om-schema:nominalWidth 1.0 ;
26   om-schema:probabilityOfFailure 0.5 ;
27   om-schema:supplyVoltage 48.0 ;
28   om-schema:transportCapacity "2"^^xsd:nonNegativeInteger ;
29   om-schema:transportDemand "1"^^xsd:nonNegativeInteger ;
30   a om-schema:CREX, owl:Class, owl:NamedIndividual ;
31   rdfs:subClassOf om-schema:Actor, [
32     a owl:Restriction ;
33     owl:maxQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
34     owl:onClass om-schema:EmiPassive ;

```

```

35     owl:onProperty om-schema:has
36 ], [
37     a owl:Restriction ;
38     owl:maxQualifiedCardinality "2"^^xsd:nonNegativeInteger ;
39     owl:onClass om-schema:Actor ;
40     owl:onProperty om-schema:hasTransportCapacity
41 ], [
42     a owl:Restriction ;
43     owl:maxQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
44     owl:onClass om-schema:Mapping ;
45     owl:onProperty om-schema:has
46 ], [
47     a owl:Restriction ;
48     owl:maxQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
49     owl:onClass om-schema:Camera ;
50     owl:onProperty om-schema:has
51 ], [
52     a owl:Restriction ;
53     owl:maxQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
54     owl:onClass om-schema:Localization ;
55     owl:onProperty om-schema:has
56 ], [
57     a owl:Restriction ;
58     owl:maxQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
59     owl:onClass om-schema:Power ;
60     owl:onProperty om-schema:has
61 ], [
62     a owl:Restriction ;
63     owl:maxQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
64     owl:onClass om-schema:LaserScanner ;
65     owl:onProperty om-schema:has
66 ], [
67     a owl:Restriction ;
68     owl:maxQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
69     owl:onClass om-schema:Locomotion ;
70     owl:onProperty om-schema:has
71 ] .

```

Listing B.6: Payload Ontology.

```

1  @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
2  @prefix : <http://www.rock-robotics.org/2015/12/robots/Payload#> .
3  @prefix owl: <http://www.w3.org/2002/07/owl#> .
4  @prefix om-schema: <http://www.rock-robotics.org/2014/01/om-schema#> .
5  @prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
6  @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
7  @prefix robots: <http://www.rock-robotics.org/2015/12/robots#> .
8  @prefix om-base: <http://www.rock-robotics.org/2014/01/om-base#> .
9
10 om-schema:Payload
11     om-schema:energyCapacity 0.0 ;
12     om-schema:mass 3.0 ;
13     om-schema:maxEnergyCapacity 0.0 ;
14     om-schema:maxHeight 0.15 ;
15     om-schema:maxWidth 0.15 ;
16     om-schema:minHeight 0.15 ;
17     om-schema:minWidth 0.15 ;
18     om-schema:nominalHeight 0.15 ;
19     om-schema:nominalPowerConsumption 25.0 ;

```

```

20    om-schema:nominalWidth 0.15 ;
21    om-schema:supplyVoltage 48.0 ;
22    om-schema:transportCapacity "0"^^xsd:nonNegativeInteger ;
23    om-schema:transportDemand "1"^^xsd:nonNegativeInteger ;
24    a om-schema:Payload, owl:Class, owl:NamedIndividual ;
25    rdfs:subClassOf om-schema:Actor, [
26        a owl:Restriction ;
27        owl:maxQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
28        owl:onClass om-schema:EmiPassive ;
29        owl:onProperty om-schema:has
30    ], [
31        a owl:Restriction ;
32        owl:maxQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
33        owl:onClass om-schema:EmiActive ;
34        owl:onProperty om-schema:has
35    ], [
36        a owl:Restriction ;
37        owl:maxQualifiedCardinality "0"^^xsd:nonNegativeInteger ;
38        owl:onClass om-schema:Actor ;
39        owl:onProperty om-schema:hasTransportCapacity
40    ] .
41
42    om-schema:PayloadBattery
43        om-schema:energyCapacity 2.4 ;
44        om-schema:mass 5.0 ;
45        om-schema:maxEnergyCapacity 2.4 ;
46        om-schema:maxHeight 0.15 ;
47        om-schema:maxWidth 0.15 ;
48        om-schema:minHeight 0.15 ;
49        om-schema:minWidth 0.15 ;
50        om-schema:nominalHeight 0.15 ;
51        om-schema:nominalPowerConsumption 0.0 ;
52        om-schema:nominalWidth 0.15 ;
53        om-schema:supplyVoltage 48.0 ;
54        om-schema:transportCapacity "0"^^xsd:nonNegativeInteger ;
55        om-schema:transportDemand "1"^^xsd:nonNegativeInteger ;
56        a om-schema:PayloadBattery, owl:Class, owl:NamedIndividual ;
57        rdfs:subClassOf om-schema:Payload, [
58            a owl:Restriction ;
59            owl:maxQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
60            owl:onClass om-schema:Power ;
61            owl:onProperty om-schema:has
62        ] .
63
64    om-schema:PayloadCamera
65        om-schema:energyCapacity 0.0 ;
66        om-schema:mass 4.0 ;
67        om-schema:maxEnergyCapacity 0.0 ;
68        om-schema:maxHeight 0.15 ;
69        om-schema:maxWidth 0.15 ;
70        om-schema:minHeight 0.15 ;
71        om-schema:minWidth 0.15 ;
72        om-schema:nominalHeight 0.15 ;
73        om-schema:nominalPowerConsumption 50.0 ;
74        om-schema:nominalWidth 0.15 ;
75        om-schema:supplyVoltage 48.0 ;
76        om-schema:transportCapacity "0"^^xsd:nonNegativeInteger ;
77        om-schema:transportDemand "1"^^xsd:nonNegativeInteger ;
78        a om-schema:PayloadCamera, owl:Class, owl:NamedIndividual ;

```

```

79     rdfs:subClassOf om-schema:Payload, [
80         a owl:Restriction ;
81         owl:maxQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
82         owl:onClass om-schema:Camera ;
83         owl:onProperty om-schema:has
84     ] .
85
86 om-schema:PayloadSoilSampler
87     om-schema:energyCapacity 0.0 ;
88     om-schema:mass 3.5 ;
89     om-schema:maxEnergyCapacity 0.0 ;
90     om-schema:maxHeight 0.15 ;
91     om-schema:maxWidth 0.15 ;
92     om-schema:minHeight 0.15 ;
93     om-schema:minWidth 0.15 ;
94     om-schema:nominalHeight 0.15 ;
95     om-schema:nominalPowerConsumption 30.0 ;
96     om-schema:nominalWidth 0.15 ;
97     om-schema:supplyVoltage 48.0 ;
98     om-schema:transportCapacity "0"^^xsd:nonNegativeInteger ;
99     om-schema:transportDemand "1"^^xsd:nonNegativeInteger ;
100    a om-schema:PayloadSoilSampler, owl:Class, owl:NamedIndividual ;
101    rdfs:subClassOf om-schema:Payload, [
102        a owl:Restriction ;
103        owl:maxQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
104        owl:onClass om-schema:SoilSampler ;
105        owl:onProperty om-schema:has
106    ] .

```

Listing B.7: Sherpa Ontology.

```

1  @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
2  @prefix : <http://www.rock-robotics.org/2015/12/robots/Sherpa#> .
3  @prefix owl: <http://www.w3.org/2002/07/owl#> .
4  @prefix om-schema: <http://www.rock-robotics.org/2014/01/om-schema#> .
5  @prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
6  @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
7  @prefix om-base: <http://www.rock-robotics.org/2014/01/om-base#> .
8
9  om-schema:Sherpa
10     om-schema:energyCapacity 10.0 ;
11     om-schema:mass 160.0 ;
12     om-schema:maxAcceleration 0.1 ;
13     om-schema:maxEnergyCapacity 10.0 ;
14     om-schema:maxHeight 2.5 ;
15     om-schema:maxVelocity 1.0 ;
16     om-schema:maxWidth 2.5 ;
17     om-schema:minAcceleration -0.1 ;
18     om-schema:minHeight 1.5 ;
19     om-schema:minVelocity 0.0 ;
20     om-schema:minWidth 2.0 ;
21     om-schema:nominalAcceleration 0.0 ;
22     om-schema:nominalHeight 2.0 ;
23     om-schema:nominalPowerConsumption 120 ;
24     om-schema:nominalVelocity 0.5 ;
25     om-schema:nominalWidth 10, 2.0 ;
26     om-schema:probabilityOfFailure 0.5 ;
27     om-schema:supplyVoltage 48.0 ;
28     om-schema:transportCapacity 10.0 ;

```

```

29 om-schema:transportDemand 1.0 ;
30 a om-schema:Sherpa, owl:Class, owl:NamedIndividual ;
31 rdfs:subClassOf om-schema:Actor, [
32   a owl:Restriction ;
33   owl:maxQualifiedCardinality "2"^^xsd:nonNegativeInteger ;
34   owl:onClass om-schema:EmiActive ;
35   owl:onProperty om-schema:has
36 ], [
37   a owl:Restriction ;
38   owl:maxQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
39   owl:onClass om-schema:Manipulation ;
40   owl:onProperty om-schema:has
41 ], [
42   a owl:Restriction ;
43   owl:maxQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
44   owl:onClass om-schema:Power ;
45   owl:onProperty om-schema:has
46 ], [
47   a owl:Restriction ;
48   owl:maxQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
49   owl:onClass om-schema:Manipulator ;
50   owl:onProperty om-schema:has
51 ], [
52   a owl:Restriction ;
53   owl:maxQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
54   owl:onClass om-schema:Localization ;
55   owl:onProperty om-schema:has
56 ], [
57   a owl:Restriction ;
58   owl:maxQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
59   owl:onClass om-schema:ForceTorqueSensor ;
60   owl:onProperty om-schema:has
61 ], [
62   a owl:Restriction ;
63   owl:maxQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
64   owl:onClass om-schema:Mapping ;
65   owl:onProperty om-schema:has
66 ], [
67   a owl:Restriction ;
68   owl:maxQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
69   owl:onClass om-schema:Locomotion ;
70   owl:onProperty om-schema:has
71 ], [
72   a owl:Restriction ;
73   owl:maxQualifiedCardinality "2"^^xsd:nonNegativeInteger ;
74   owl:onClass om-schema:Camera ;
75   owl:onProperty om-schema:has
76 ], [
77   a owl:Restriction ;
78   owl:maxQualifiedCardinality "4"^^xsd:nonNegativeInteger ;
79   owl:onClass om-schema:EmiPassive ;
80   owl:onProperty om-schema:has
81 ], [
82   a owl:Restriction ;
83   owl:maxQualifiedCardinality "10"^^xsd:nonNegativeInteger ;
84   owl:onClass om-schema:Actor ;
85   owl:onProperty om-schema:hasTransportCapacity
86 ] .

```

B.4 Project Ontology

Listing B.8: *Project Specific Ontology.*

```

1 @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
2 @prefix : <http://www.rock-robotics.org/2015/12/projects/TransTerrA#> .
3 @prefix owl: <http://www.w3.org/2002/07/owl#> .
4 @prefix om-schema: <http://www.rock-robotics.org/2014/01/om-schema#> .
5 @prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
6 @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
7 @prefix om-base: <http://www.rock-robotics.org/2014/01/om-base#> .
8
9 om-schema:BaseCamp
10     rdfs:subClassOf :BulkyAgent .
11
12 om-schema:CREX
13     rdfs:subClassOf :BulkyAgent, [
14         a owl:Restriction ;
15         owl:maxQualifiedCardinality "0"^^xsd:nonNegativeInteger ;
16         owl:onClass :BulkyAgent ;
17         owl:onProperty om-schema:hasTransportCapacity
18     ] .
19
20 om-schema:CoyoteIII
21     rdfs:subClassOf :BulkyAgent, [
22         a owl:Restriction ;
23         owl:maxQualifiedCardinality "0"^^xsd:nonNegativeInteger ;
24         owl:onClass :BulkyAgent ;
25         owl:onProperty om-schema:hasTransportCapacity
26     ] .
27
28 om-schema:Sherpa
29     rdfs:subClassOf [
30         a owl:Restriction ;
31         owl:maxQualifiedCardinality "2"^^xsd:nonNegativeInteger ;
32         owl:onClass :BulkyAgent ;
33         owl:onProperty om-schema:hasTransportCapacity
34     ] .
35
36 <http://www.rock-robotics.org/2015/12/projects/TransTerrA>
37     a owl:Ontology ;
38     owl:imports <http://www.rock-robotics.org/2014/01/om-base#>, <http://www.rock-robotics.org/2014/01/om-schema#>, <http://www.rock-robotics.org/2015/12/robots/BaseCamp>, <http://www.rock-robotics.org/2015/12/robots/CREX>, <http://www.rock-robotics.org/2015/12/robots/CoyoteIII>, <http://www.rock-robotics.org/2015/12/robots/Payload>, <http://www.rock-robotics.org/2015/12/robots/Sherpa> .
39
40 :BulkyAgent
41     a owl:Class ;
42     rdfs:subClassOf om-schema:Actor .

```

B.5 Interface Test Ontology

Listing B.9: *Ontology to illustrate generic interface modelling.*

```

1 @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
2 @prefix : <http://www.rock-robotics.org/2017/10/om-lego#> .

```



```

3 @prefix owl: <http://www.w3.org/2002/07/owl#> .
4 @prefix om-schema: <http://www.rock-robotics.org/2014/01/om-schema#> .
5 @prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
6 @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
7
8 <http://www.rock-robotics.org/2017/10/om-lego#>
9   a owl:Ontology ;
10   owl:imports <http://www.rock-robotics.org/2014/01/om-base#> .
11
12 :AdapterBlock
13   a owl:Class ;
14   rdfs:subClassOf om-schema:Actor, [
15     a owl:Restriction ;
16     owl:maxQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
17     owl:onClass :FlatInterface ;
18     owl:onProperty om-schema:has
19   ], [
20     a owl:Restriction ;
21     owl:maxQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
22     owl:onClass :LowerInterface ;
23     owl:onProperty om-schema:has
24   ] .
25
26 :BlueBlock
27   a owl:Class ;
28   rdfs:subClassOf :StandardBlock .
29
30 :Flag
31   a owl:Class ;
32   rdfs:subClassOf om-schema:Actor, [
33     a owl:Restriction ;
34     owl:maxQualifiedCardinality "1"^^xsd:nonNegativeInteger ;
35     owl:onClass :FlatInterface ;
36     owl:onProperty om-schema:has
37   ] .
38
39 :FlatInterface
40   om-schema:compatibleWith :FlatInterface ;
41   a :FlatInterface, owl:Class, owl:NamedIndividual ;
42   rdfs:subClassOf om-schema:MechanicalInterface .
43
44 :GreenBlock
45   a owl:Class ;
46   rdfs:subClassOf :StandardBlock .
47
48 :LegoBlock
49   a owl:Class ;
50   rdfs:subClassOf om-schema:Actor .
51
52 :LowerInterface
53   om-schema:compatibleWith :UpperInterface ;
54   a :LowerInterface, owl:Class, owl:NamedIndividual ;
55   rdfs:subClassOf om-schema:MechanicalInterface .
56
57 :RedBlock
58   a owl:Class ;
59   rdfs:subClassOf :StandardBlock .
60
61 :StandardBlock

```

```
62     a owl:Class ;
63     rdfs:subClassOf :LegoBlock, [
64         a owl:Restriction ;
65         owl:maxQualifiedCardinality "2"^^xsd:nonNegativeInteger ;
66         owl:onClass :LowerInterface ;
67         owl:onProperty om-schema:has
68     ], [
69         a owl:Restriction ;
70         owl:maxQualifiedCardinality "2"^^xsd:nonNegativeInteger ;
71         owl:onClass :UpperInterface ;
72         owl:onProperty om-schema:has
73     ] .
74
75 :UpperInterface
76     om-schema:compatibleWith :LowerInterface ;
77     a :UpperInterface, owl:Class, owl:NamedIndividual ;
78     rdfs:subClassOf om-schema:MechanicalInterface .
79
80 :YellowBlock
81     a owl:Class ;
82     rdfs:subClassOf :StandardBlock .
```

Appendix **C**

Combinatorics

C.1 Generation of composite agent combinations

The generative approach for finding the feasible number of agent combinations is based on Algorithm 2 and Algorithm 3 (from (Roehr and Hartanto 2015)). Algorithm 2 can be used to generate a set of feasible link combinations, but considers a maximum coalition size. Each link combination is then equivalent to a composite agent consisting of a maximum number of atomic agents.

C.2 Generation of limited combinations

An algorithm to compute combinations with variable repetitions per element type.

Algorithm 2: Compute valid combinations of links (interface connections) up to maximum composition size n , a combination directly maps to a composite agent.

Data: $A \leftarrow \{\text{All atomic agents}\}$, $n \geq 0$, $I \leftarrow \emptyset$, $\mathcal{L} \leftarrow \emptyset$

Result: Set of link combinations

```

1 Function computeLinkCombinations( $A, n$ ) is
2   forall  $a \in A$  do
3      $I \leftarrow I \cup \text{allPhysicalInterfacesOf}(a)$ ;
4    $\mathcal{L} \leftarrow \text{validCombinationsOfSize}(I, 2)$ ;
5    $l_{\max} \leftarrow \min(|A| - 1, n - 1)$ ;
6    $\mathcal{C} \leftarrow \text{combinationsUpToSize}(\mathcal{L}, l_{\max})$ ;
7    $\mathcal{LC} \leftarrow \emptyset$ ;
8   forall  $C \in \mathcal{C}$  do
9     forall  $L_c \in C$  do
10       $A_c \leftarrow \text{involvedAgents}(L_c)$ ;
11      if  $|A_c| - 1 = |L_c|$  then
12        if  $\text{maxInterfaceUse}(L_c) = 1$  then
13           $\mathcal{LC} \leftarrow \mathcal{LC} \cup \{L_c\}$ ;
14 return  $\mathcal{LC}$ 

```

Algorithm 3: Custom inference for functionalities (services and capabilities) of atomic and composite agents.

Data: $A \leftarrow \{\text{All atomic agents}\}$, $S \leftarrow \{\text{All services}\}$, $C \leftarrow \{\text{All capabilities}\}$, $n =$
maximum link count

```

1 Function inferAgentsAndFunctionalities( $A, C, S, n$ ) is
2    $\mathcal{AC} \leftarrow \emptyset$ ;
3    $\mathcal{LC} \leftarrow \text{computeLinkCombinations}(A, n)$ ;
4   forall  $L \in \mathcal{LC}$  do
5      $A_c \leftarrow \text{createCompositeAgent}(L)$ ;
6      $S_c \leftarrow \text{inferAvailableServices}(A_c, S)$ ;
7      $C_c \leftarrow \text{inferAvailableCapabilities}(A_c, C)$ ;
8      $\mathcal{AC} \leftarrow \mathcal{AC} \cup \{(A_c, S_c, C_c)\}$ ;
9 return  $\mathcal{AC}$ ;

```

Algorithm 4: The *limited combination* algorithm: generation of all feasible combinations with limited element repetitions.

Data: S : array of unique elements, $\mu(i_s)$: number of repetitions of the element at index i_s in the result C , where i_s is the index of s in S

Result: C : set of combinations with repeated elements

```

1 begin
2   Let  $CUR[1 \dots |S|]$  and  $UB[1 \dots |S|]$  be new arrays;
3    $C \leftarrow \emptyset$ ;
4   for  $i \leftarrow 1$  to  $|S|$  do
5      $CUR[i] \leftarrow 0$ ;
6      $UB[i] \leftarrow \mu(i)$ ;
7   while true do
8      $hasNext \leftarrow \text{increment}(CUR, UB)$ ;
9     if  $hasNext$  then
10       $C \leftarrow C \cup \text{instantiate}(CUR)$ ;
11    else
12      return  $C$ ;
13 Function  $\text{instantiate}(CUR: \text{integer array})$  is
14    $CMB \leftarrow \emptyset$ ;
15   for  $i \leftarrow 1$  to  $|CUR|$  by 1 do
16     for  $c \leftarrow 1$  to  $CUR[i]$  by 1 do
17        $CMB \leftarrow CMB \cup \{i\}$ ;
18   return  $CMB$ ;
19 Function  $\text{increment}(CUR: \text{integer array}, UB: \text{integer array})$  is
20   for  $p \leftarrow 1$  to  $|S|$  by 1 do
21     if  $CUR[p] < UB[p]$  then
22        $CUR[p] \leftarrow CUR[p] + 1$ ;
23     if  $p = 0$  then
24       return true;
25     for  $lp \leftarrow p$  to 1 by  $-1$  do
26        $CUR[lp] \leftarrow 0$ ;
27   else
28     if  $p = |S|$  then
29       return false;

```

Appendix D

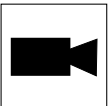

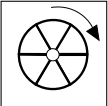
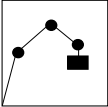
Pictograms




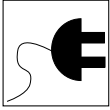
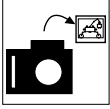
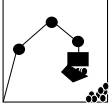
A set of pictograms has been developed to facilitate the understanding and illustration of robotic missions. For clarification the following section describes the meaning of these pictogram and points to necessary details.

D.1 Functions

Functions are illustrated in Table D.1.

Table D.1: *Pictograms for functions.*


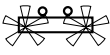
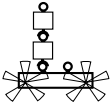
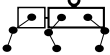
Pictogram	Name	Description
	imaging	Capability of taking images of the environment
	localisation	Determining the current position with respect to a global coordinate system
	locomotion	Capability of moving across terrain
	manipulation	Capability of manipulating objects

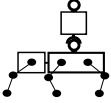
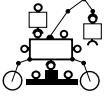
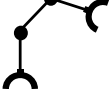


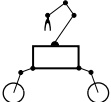





	mapping	Capability of mapping an environment
	moveto	Capability of moving to a location given as position in a global coordinate system
	power	Availability of an internal power source, e.g., a battery
	power provider	Ability to provide access to the internal power source, e.g. a battery
	image provider	Taking images and providing access to images for others
	soil-sampling	Taking soil samples

D.2 Robotic systems

Table D.2 lists pictograms which illustrate atomic and composite agents used in this thesis (cf. Chapter 2).

Table D.2: *Pictograms for robotic systems.*

Pictogram	Name	Description
	Base Camp	A Base Camp is a platform hosting communication infrastructure elements and five EmiPassive. Its purpose is to provide a logistic hub to temporarily store payloads in a logistic chain.
	Coyote III	Coyote III is a mobile exploration robot comprising two EmiPassive
	Coyote III extended	An agent coalition consisting of one Coyote III and two payload items
	CREX	The six legged robot CREX, comprises a single EmiPassive

	CREX extended	An agent coalition consisting of one CREX extended with a payload item
	SherpaTT extended	SherpaTT extended with two payload items and a base camp
	Manipulator	SIMA modular manipulator with two EmiActive
	Female (Active) Connector	Female EMI (EmiActive)
	Male (Passive) Connector	Male EMI (EmiPassive)
	Simple SherpaTT	A SherpaTT variant without EMIs for reconfiguration
	Simple Robot	A simple mobile robot without EMIs
	Payload	Standard payload with an EmiActive and an EmiPassive
	Seismograph	A payload hosting a seismograph module
	Soil sampling module	The soil sampling module represents a combined system of a shovel mechanism and soil sample container
	Power module	A payload which contains a battery to power itself and attached agents

Appendix E

Router configuration

The distributed communication infrastructure requires a special setup of the OM2P mesh routers. Therefore, each router has been flashed with the open-source router operating system openWRT OpenWRT/LEDE Community 2018, Version ChaosCalmer 15.05. The usage of the routing software B.A.T.M.A.N. Open-mesh 2012 has been verified using openWRT's default selection version 2016.2 and the more recent 2017.4. The latter, however, required manual patching of the installation procedure to work with the respective openWRT version. For reproduction of results the setup configuration of the OM2P Open Mesh Inc. 2018 routers are listed in the following.

E.1 Configuration files

Example configuration files for an access point with IP 10.250.247.133/24 and no encryption in use.

Listing E.1: */etc/config/network*

```
1 config interface 'loopback'
2     option ifname 'lo'
3     option proto 'static'
4     option ipaddr '127.0.0.1'
5     option netmask '255.0.0.0'
6
7 config globals 'globals'
8     option ula_prefix 'fd9f:4cf3:6d90::/48'
9
10 config interface 'lan'
11     option ifname 'eth0 bat0'
12     option force_link '1'
13     option type 'bridge'
```

```
14      option proto 'static'
15      option netmask '255.255.255.0'
16      option ip6assign '60'
17      option ipaddr '10.250.247.133'
18
19 config interface 'wan'
20     option ifname 'eth1'
21     option proto 'dhcp'
22
23 config interface 'wan6'
24     option ifname 'eth1'
25     option proto 'dhcpv6'
26
27 config interface 'mesh0'
28     option ifname 'adhoc0'
29     option proto 'batadv'
30     option mtu '1532'
31     option mesh 'bat0'
```

Listing E.2: */etc/config/batman-adv*

```
1 config mesh 'bat0'
2     option interfaces 'mesh0'
3     option distributed_arp_table '0'
4     option bridge_loop_avoidance '1'
```

Listing E.3: */etc/config/wireless*

```
1 config wifi-device 'radio0'
2     option type 'mac80211'
3     option hwmode '11g'
4     option path 'platform/ar933x_wmac'
5     option htmode 'HT40+'
6     option channel '1'
7     option disabled '0'
8     option phy 'phy0'
9
10 config wifi-iface
11     option device 'radio0'
12     option ifname 'adhoc0'
13     option encryption 'none'
14     option network 'mesh0'
15     option mode 'adhoc'
16     option bssid '02:CA:FE:CA:CA:40'
17     option ssid 'robot-mesh'
18     option mcast_rate '11000'
```

Appendix **F**

PDDL: Domain and Problem Examples

The following section provides an example of a PDDL-based problem encoding for a reconfigurable multi-robot system. It has been solved with LAMA (Richter and Westphal 2010).

F.1 Domain definition

```
1 ; BEGIN domain definition
2 (define (domain om)
3   (:requirements
4     :typing
5     :strips
6     :equality
7     :conditional-effects
8     :action-costs
9   ); end requirements
10  (:types
11    Actor
12    ActorModel
13    Capability
14    CapabilityModel
15    CompositeActor
16    CompositeActorModel
17    Interface
18    InterfaceModel
19    Location
20    Mission
21    Requirement
22    Resource
23    ResourceModel
24    Service
25    ServiceModel
26    Class
```

```

27     Thing
28 ); end types
29 (:constants
30     PayloadCamera-instance-0+Sherpa-instance-0[1]
31     PayloadCamera-instance-0+Sherpa-instance-0[2]
32     PayloadCamera-instance-0+Sherpa-instance-0[3]
33     PayloadCamera-instance-0+Sherpa-instance-0[4]
34     CREX-instance-0+PayloadCamera-instance-0[1]
35     PayloadCamera-instance-0+Sherpa-instance-0[5]
36     CREX-instance-0+Sherpa-instance-0[1]
37     PayloadCamera-instance-0+Sherpa-instance-0[6]
38     CREX-instance-0+Sherpa-instance-0[2]
39     CREX-instance-0+PayloadCamera-instance-0+Sherpa-instance-0[1]
40     CREX-instance-0+PayloadCamera-instance-0+Sherpa-instance-0[2]
41     CREX-instance-0+PayloadCamera-instance-0+Sherpa-instance-0[3]
42     CREX-instance-0+PayloadCamera-instance-0+Sherpa-instance-0[4]
43     CREX-instance-0+PayloadCamera-instance-0+Sherpa-instance-0[5]
44     CREX-instance-0+PayloadCamera-instance-0+Sherpa-instance-0[6]
45     CREX-instance-0+PayloadCamera-instance-0+Sherpa-instance-0[7]
46     CREX-instance-0+PayloadCamera-instance-0+Sherpa-instance-0[8]
47     CREX-instance-0+PayloadCamera-instance-0+Sherpa-instance-0[9]
48     CREX-instance-0+PayloadCamera-instance-0+Sherpa-instance-0[10]
49     CREX-instance-0+PayloadCamera-instance-0+Sherpa-instance-0[11]
50     CREX-instance-0+PayloadCamera-instance-0+Sherpa-instance-0[12]
51     Sherpa-instance-0
52     PayloadCamera-instance-0
53     CREX-instance-0
54     - Actor
55     EmiPowerProvider
56     ImageProvider
57     StereoImageProvider
58     LocationImageProvider
59     - Service
60 ); end constants
61 (:predicates
62     ( at ?x - Actor ?l - Location )
63     ( operative ?x - Actor )
64     ( embodies ?x - Actor ?y - Actor )
65     ( mobile ?x - Actor )
66     ( provides ?x - Actor ?s - Service )
67 ); end predicates
68 (:functions
69     ( distance ?start - Location ?end - Location )
70     ( total-cost )
71 )
72 (:action move
73     :parameters ( ?obj - Actor ?m - Location ?l - Location )
74     :precondition (and (at ?obj ?m) (not (= ?m ?l)) (mobile ?obj) (operative ?
75         obj))
76     :effect (and (at ?obj ?l) (not (at ?obj ?m)))
77 )
78 (:action physical_merge
79     :parameters ( ?z - Actor ?l - Location )
80     :precondition (and (forall (?a - Actor) (or (and (embodies ?z ?a) (operative
81         ?a) (at ?a ?l)) (not (embodies ?z ?a)))) (not (atomic ?z)) (not (
82         operative ?z)))
83     :effect (and (forall (?a - Actor) (when (embodies ?z ?a) (and (not (at ?a ?l
84         )) (not (operative ?a)))))) (operative ?z) (at ?z ?l))
85 )

```

```

82      (:action physical_split
83        :parameters ( ?z - Actor ?l - Location )
84        :precondition (and (operative ?z) (at ?z ?l))
85        :effect (and (not (operative ?z)) (not (at ?z ?l)) (forall (?a - Actor) (
            when (embodies ?z ?a) (and (operative ?a) (at ?a ?l)))))
86      )
87 ; END domain definition
88 )

```

F.2 Problem definition

```

1  ; BEGIN problem definition
2  (define (problem om-partial)
3    (:domain om)
4    (:objects
5      s0 - Location
6      c0 - Location
7      p0 - Location
8      m0 - Location
9    )
10   (:init
11     (embodies PayloadCamera-instance-0+Sherpa-instance-0[1] Sherpa-instance-0)
12     (embodies PayloadCamera-instance-0+Sherpa-instance-0[1] PayloadCamera-
        instance-0)
13     (mobile PayloadCamera-instance-0+Sherpa-instance-0[1])
14     (provides PayloadCamera-instance-0+Sherpa-instance-0[1] EmiPowerProvider)
15     (provides PayloadCamera-instance-0+Sherpa-instance-0[1] ImageProvider)
16     (provides PayloadCamera-instance-0+Sherpa-instance-0[1] StereoImageProvider)
17     (provides PayloadCamera-instance-0+Sherpa-instance-0[1]
        LocationImageProvider)
18     (embodies PayloadCamera-instance-0+Sherpa-instance-0[2] Sherpa-instance-0)
19     (embodies PayloadCamera-instance-0+Sherpa-instance-0[2] PayloadCamera-
        instance-0)
20     ....
21     (provides PayloadCamera-instance-0+Sherpa-instance-0[6] EmiPowerProvider)
22     (provides PayloadCamera-instance-0+Sherpa-instance-0[6] ImageProvider)
23     (provides PayloadCamera-instance-0+Sherpa-instance-0[6] StereoImageProvider)
24     (provides PayloadCamera-instance-0+Sherpa-instance-0[6]
        LocationImageProvider)
25     (embodies CREX-instance-0+Sherpa-instance-0[2] Sherpa-instance-0)
26     (embodies CREX-instance-0+Sherpa-instance-0[2] CREX-instance-0)
27     (mobile CREX-instance-0+Sherpa-instance-0[2])
28     (provides CREX-instance-0+Sherpa-instance-0[2] EmiPowerProvider)
29     (provides CREX-instance-0+Sherpa-instance-0[2] ImageProvider)
30     (provides CREX-instance-0+Sherpa-instance-0[2] StereoImageProvider)
31     (provides CREX-instance-0+Sherpa-instance-0[2] LocationImageProvider)
32     (embodies CREX-instance-0+PayloadCamera-instance-0+Sherpa-instance-0[1]
        Sherpa-instance-0)
33     (embodies CREX-instance-0+PayloadCamera-instance-0+Sherpa-instance-0[1]
        PayloadCamera-instance-0)
34     (embodies CREX-instance-0+PayloadCamera-instance-0+Sherpa-instance-0[1] CREX
        -instance-0)
35     (mobile CREX-instance-0+PayloadCamera-instance-0+Sherpa-instance-0[1])
36     ...
37     (provides CREX-instance-0+PayloadCamera-instance-0+Sherpa-instance-0[12]
        EmiPowerProvider)
38     (provides CREX-instance-0+PayloadCamera-instance-0+Sherpa-instance-0[12]
        ImageProvider)

```

```

39      (provides CREX-instance-0+PayloadCamera-instance-0+Sherpa-instance-0[12]
        StereoImageProvider)
40      (provides CREX-instance-0+PayloadCamera-instance-0+Sherpa-instance-0[12]
        LocationImageProvider)
41      (mobile Sherpa-instance-0)
42      (provides Sherpa-instance-0 EmiPowerProvider)
43      (provides Sherpa-instance-0 ImageProvider)
44      (provides Sherpa-instance-0 StereoImageProvider)
45      (provides Sherpa-instance-0 LocationImageProvider)
46      (mobile CREX-instance-0)
47      (provides CREX-instance-0 ImageProvider)
48      (provides CREX-instance-0 LocationImageProvider)
49      (atomic Sherpa-instance-0)
50      (atomic PayloadCamera-instance-0)
51      (atomic CREX-instance-0)
52      (at Sherpa-instance-0 s0)
53      (at CREX-instance-0 c0)
54      (at PayloadCamera-instance-0 p0)
55      (operative Sherpa-instance-0)
56      (operative CREX-instance-0)
57      (operative PayloadCamera-instance-0)
58      )
59      (:goal (exists (?a - Actor) (and (provides ?a LocationImageProvider) (at ?a m0))
        ))
60 ; END problem definition
61 )

```

F.3 Solution: an action plan

```

1 0: (move sherpa-instance-0 s0 p0) [1]
2 0: (move crex-instance-0 c0 p0) [1]
3 1: (physical_merge crex-instance-0_payloadcamera-instance-0_sherpa-instance-0__12 p0
    ) [1]
4 2: (move crex-instance-0_payloadcamera-instance-0_sherpa-instance-0__12 p0 s0) [1]

```


Appendix G

TemPl: Representations and Tools

G.1 XML-based representation for the Mission Specification

The following section describes the XML-based format developed to encode a mission specification. Table G.1 briefly explains the tags in this description.

Table G.1: *Description of XML Tags in the mission specification.*

Tag	Description
mission	root node
organisation_model	reference to the organisation model, which shall be used for planning this mission
resources	describe the set of maximum available atomic agents
resource	a resource description constraining, model and cardinality information
model	reference to a resource model (functionality or atomic agent class) in the ontology
minCardinality	minimum cardinality of a resource model
maxCardinality	maximum cardinality of a resource model
constants	location constants described by id and either an combination of x,y,z coordinate, or latitude/longitude with reference to given object radius (earth or moon)
requirements	set of identifiable requirements corresponding to stqes
requirement	tuple of spatial, temporal and resource requirement to represent a stqe
spatial-requirement	location reference
temporal-requirement	interval definition for the stqe
resource-requirement	model requirements (atomic agents and functionalities) with min/max cardinality constraints
constraints	collection of temporal constraints and model constraints
temporal-constraints	qualitative constraints: greaterThan, lessThan, equals, combined with quantitative constraints: min-duration, max-duration
model constraints	min-equal, max-equal, min-distinct, max-distinct

```

1  <!-- <!xml version="1.0"?> -->
2  <mission xmlns="http://www.rock-robotics.org/missions#"
3      xmlns:om="http://www.rock-robotics.org/2014/01/om-schema#"
4      xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
5  >
6  <name>Immobile and mobile system deployment</name>
7  <description>
8      Immobile system needs a transport, but the mobile system is not dedicated to
9      transport the
10     device
11 </description>
12 <organization_model>http://www.rock-robotics.org/2015/12/projects/TransTerra</
13     organization_model>
14 <resources>
15     <resource>
16         <model>http://www.rock-robotics.org/2014/01/om-schema#Sherpa</model>
17         <maxCardinality>2</maxCardinality>
18     </resource>
19     <resource>
20         <model>http://www.rock-robotics.org/2014/01/om-schema#Payload</model>
21         <maxCardinality>3</maxCardinality>
22     </resource>
23 </resources>
24 <constants>
25     <location>
26         <id>lander</id>
27         <radius>moon</radius>
28         <latitude>-83.82009</latitude>
29         <longitude>87.53932</longitude>
30     </location>
31     <location>
32         <id>base1</id>
33         <radius>moon</radius>
34         <latitude>-84.1812</latitude>
35         <longitude>87.60494</longitude>
36     </location>
37     <location>
38         <id>base2</id>
39         <radius>moon</radius>
40         <latitude>-83.58491</latitude>
41         <longitude>85.98319</longitude>
42     </location>
43 </constants>
44 <requirements>
45     <requirement id='0'>
46         <spatial-requirement>
47             <location>
48                 <id>lander</id>
49             </location>
50         </spatial-requirement>
51         <temporal-requirement>
52             <from>t0</from>
53             <to>t1</to>
54         </temporal-requirement>
55         <resource-requirement>
56             <resource>
57                 <model>http://www.rock-robotics.org/2014/01/om-schema#Sherpa</model>
58                 <minCardinality>2</minCardinality>
59                 <maxCardinality>2</maxCardinality>

```

```

58         </resource>
59     <resource>
60         <model>http://www.rock-robotics.org/2014/01/om-schema#Payload</model>
61         <minCardinality>3</minCardinality>
62         <maxCardinality>3</maxCardinality>
63     </resource>
64 </resource-requirement>
65 </requirement>
66 <requirement id='1'>
67     <spatial-requirement>
68         <location>
69             <id>base1</id>
70         </location>
71     </spatial-requirement>
72     <temporal-requirement>
73         <from>t2</from>
74         <to>t3</to>
75     </temporal-requirement>
76 </resource-requirement>
77 <resource>
78     <model>http://www.rock-robotics.org/2014/01/om-schema#Payload</model>
79     <minCardinality>1</minCardinality>
80 </resource>
81 </resource-requirement>
82 </requirement>
83 <requirement id='2'>
84     <spatial-requirement>
85         <location>
86             <id>base2</id>
87         </location>
88     </spatial-requirement>
89     <temporal-requirement>
90         <from>t2</from>
91         <to>t5</to>
92     </temporal-requirement>
93 </resource-requirement>
94 <resource>
95     <model>http://www.rock-robotics.org/2014/01/om-schema#Sherpa</model>
96     <minCardinality>1</minCardinality>
97 </resource>
98 <resource>
99     <model>http://www.rock-robotics.org/2014/01/om-schema#Payload</model>
100     <minCardinality>1</minCardinality>
101 </resource>
102 </resource-requirement>
103 </requirement>
104 <requirement id='3'>
105     <spatial-requirement>
106         <location>
107             <id>base2</id>
108         </location>
109     </spatial-requirement>
110     <temporal-requirement>
111         <from>t4</from>
112         <to>t5</to>
113     </temporal-requirement>

```

```

114         <resource-requirement>
115             <resource>
116                 <model>http://www.rock-robotics.org/2014/01/om-schema#Payload</model>
117                 <minCardinality>1</minCardinality>
118             </resource>
119         </resource-requirement>
120     </requirement>
121 </requirements>
122 <constraints>
123     <temporal-constraints>
124         <greaterThan lval="t2" rval="t1" />
125         <greaterThan lval="t3" rval="t2" />
126         <greaterThan lval="t4" rval="t3" />
127         <greaterThan lval="t5" rval="t4" />
128     </temporal-constraints>
129     <model-constraints>
130         <min-property value="1">
131             <model>http://www.rock-robotics.org/2014/01/om-schema#TransportProvider
132             </model>
133             <requirements>1</requirements>
134             <property>http://www.rock-robotics.org/2014/01/om-schema#
135             transportCapacity</property>
136         </min-property>
137         <min-equal value="1">
138             <model>http://www.rock-robotics.org/2014/01/om-schema#Sherpa</model>
139             <requirements>2,3</requirements>
140         </min-equal>
141     </model-constraints>
142 </constraints>
143 </mission>

```

G.2 Multi-commodity min cost flow

Flow optimisation is based on the inclusion of linear programming. After the creation of the planning problem by the planner it can be exported into a standard format and solved by an available linear program solver. The following listing shows the flow optimisation formulation for the simple mission example illustrated in Section 4.4.

```

1  \* Problem: multicommodity_min_cost_flow *\
2
3  Minimize
4  obj: + x2 + x3 + x4 + x6 + x7 + x8 + x9 + x10 + x12 + x13 + x14 + x15
5  + x16 + x17 + x18 + x19 + x20 + x21
6
7  Subject To
8  y1: + x1 - ~r_1 = 0
9  y2: + x2 - ~r_2 = 0
10 y3: + x3 - ~r_3 = 0
11 y4: + x4 - ~r_4 = 0
12 y5: + x5 - ~r_5 = 0
13 y6: + x6 - ~r_6 = 0
14 y7: + x7 - ~r_7 = 0
15 y8: + x8 - ~r_8 = 0
16 y9: + x9 - ~r_9 = 0
17 y10: + x10 - ~r_10 = 0
18 y11: + x11 - ~r_11 = 0
19 y12: + x12 - ~r_12 = 0
20 y13: + x13 - ~r_13 = 0
21 y14: + x14 - ~r_14 = 0
22 y15: + x15 - ~r_15 = 0
23 y16: + x16 - ~r_16 = 0
24 y17: + x17 - ~r_17 = 0
25 y18: + x18 - ~r_18 = 0
26 y19: + x19 - ~r_19 = 0

```

```

27  y20: + x20 - ~r_20 = 0
28  y21: + x21 - ~r_21 = 0
29  y22: + x22 - ~r_22 = 0
30  y23: + x23 - ~r_23 = 0
31  y24: + x24 - ~r_24 = 0
32  y25: + x1 - x2 = 0
33  y26: + x2 >= 0
34  y27: + x1 >= 0
35  y28: + x2 - x8 = 0
36  y29: + x8 >= 0
37  y30: + x2 >= 0
38  y31: + x3 - x4 = 0
39  y32: + x4 >= 1
40  y33: + x3 >= 1
41  y34: + x4 - x21 = 0
42  y35: + x21 >= 0
43  y36: + x4 >= 0
44  y37: + x5 - x6 = 0
45  y38: + x6 >= 0
46  y39: + x5 >= 0
47  y40: + x6 - x7 - x9 = 0
48  y41: + x7 + x9 >= 0
49  y42: + x6 >= 0
50  y43: - x3 + x7 + x8 - x14 = 0
51  y44: + x3 + x14 >= 1
52  y45: + x7 + x8 >= 1
53  y46: + x9 - x15 = 0
54  y47: + x15 >= 0
55  y48: + x9 >= 0
56  y49: + x10 - x20 = 0
57  y50: + x20 >= 0
58  y51: + x10 >= 0
59  y52: - x1 - x5 - x11 = -1
60  y53: + x11 - x12 = 0
61  y54: + x12 >= 0
62  y55: + x11 >= 0
63  y56: + x12 - x13 = 0
64  y57: + x13 >= 0
65  y58: + x12 >= 0
66  y59: + x13 - x16 = 0
67  y60: + x16 >= 0
68  y61: + x13 >= 0
69  y62: - x10 + x14 + x15 - x17 = 0
70  y63: + x10 + x17 >= 0
71  y64: + x14 + x15 >= 0
72  y65: + x16 - x18 = 0
73  y66: + x18 >= 0
74  y67: + x16 >= 0
75  y68: + x17 + x18 - x19 = 0
76  y69: + x19 >= 1
77  y70: + x17 + x18 >= 1
78  y71: + x19 - x22 = 0
79  y72: + x22 >= 1
80  y73: + x19 >= 1
81  y74: + x20 - x23 = 0
82  y75: + x23 >= 0
83  y76: + x20 >= 0
84  y77: + x21 - x24 = 0
85  y78: + x24 >= 0

86  y79: + x21 >= 0
87  y80: + x22 + x23 + x24 = 1
88
89  Bounds
90  0 <= ~r_1 <= 4294967295
91  0 <= ~r_2 <= 4294967295
92  0 <= ~r_3 <= 4294967295
93  0 <= ~r_4 <= 4294967295
94  0 <= ~r_5 <= 4294967295
95  0 <= ~r_6 <= 4294967295
96  0 <= ~r_7 <= 10
97  0 <= ~r_8 <= 4294967295
98  0 <= ~r_9 <= 4294967295
99  0 <= ~r_10 <= 4294967295
100 0 <= ~r_11 <= 4294967295
101 0 <= ~r_12 <= 4294967295
102 0 <= ~r_13 <= 4294967295
103 0 <= ~r_14 <= 10
104 0 <= ~r_15 <= 4294967295
105 0 <= ~r_16 <= 4294967295
106 0 <= ~r_17 <= 10
107 0 <= ~r_18 <= 4294967295
108 0 <= ~r_19 <= 4294967295
109 0 <= ~r_20 <= 4294967295
110 0 <= ~r_21 <= 4294967295
111 0 <= ~r_22 <= 4294967295
112 0 <= ~r_23 <= 4294967295
113 0 <= ~r_24 <= 4294967295
114 0 <= x1 <= 4294967295
115 0 <= x2 <= 4294967295
116 0 <= x3 <= 4294967295
117 0 <= x4 <= 4294967295
118 0 <= x5 <= 4294967295
119 0 <= x6 <= 4294967295
120 0 <= x7 <= 10
121 0 <= x8 <= 4294967295
122 0 <= x9 <= 4294967295
123 0 <= x10 <= 4294967295
124 0 <= x11 <= 4294967295
125 0 <= x12 <= 4294967295
126 0 <= x13 <= 4294967295
127 0 <= x14 <= 10
128 0 <= x15 <= 4294967295
129 0 <= x16 <= 4294967295
130 0 <= x17 <= 10
131 0 <= x18 <= 4294967295
132 0 <= x19 <= 4294967295
133 0 <= x20 <= 4294967295
134 0 <= x21 <= 4294967295
135 0 <= x22 <= 1
136 x23 = 0
137 x24 = 0
138
139 Generals
140 x1
141 x2
142 x3
143 x4
144 x5

```

145	x6	156	x17
146	x7	157	x18
147	x8	158	x19
148	x9	159	x20
149	x10	160	x21
150	x11	161	x22
151	x12	162	x23
152	x13	163	x24
153	x14	164	
154	x15	165	End
155	x16		

G.3 A graph-based plan visualisation

This thesis introduces a representation schema for multi-robot missions which allows a compact visualisation (see Chapter 4). Figure G.1 is repeated here for further explanation of the illustration scheme.

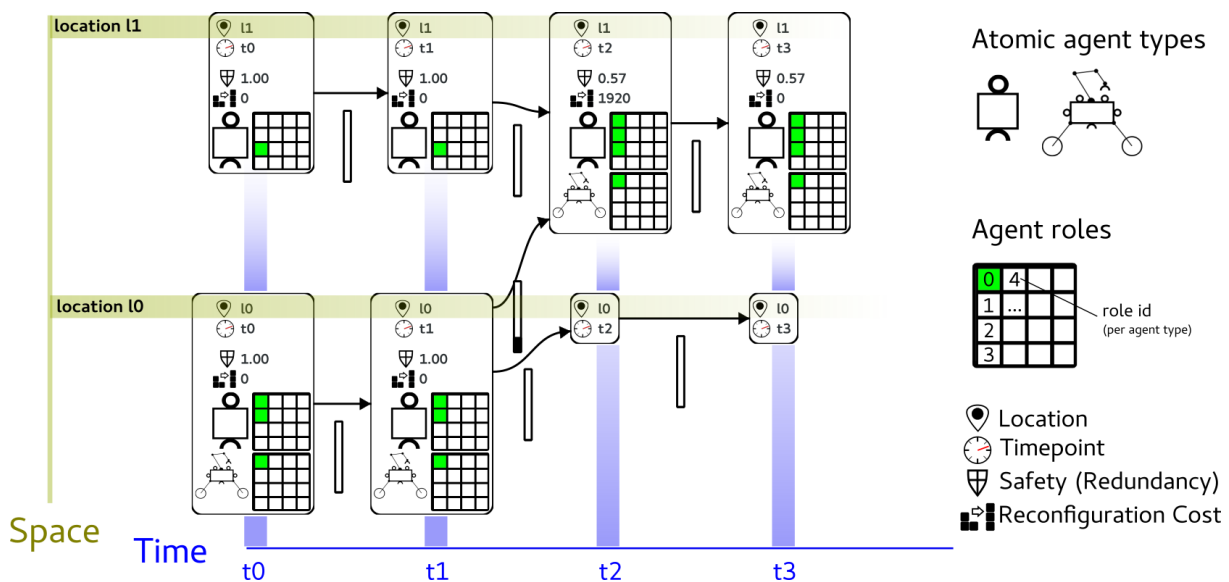


Figure G.1: A mission solution representation.

Each vertex in the network represents a space-time tuple which relates space, time and agent type assignments. Each agent type is represented by a corresponding pictogram. The number of used agent roles can be derived from the coloured squares next to the pictogram. Each small square is related to one role/agent instance - the position of the square in the grid is unique for every instance. To identify the route of an agent instance the markers have to be followed through the graph. Colour-coding of squares indicates whether a role fulfils a direct requirement (green) or whether it is present without having an explicit requirement to be there. A red square indicates an unfulfilled requirement. Safety attributes and reconfiguration cost for each vertex are depicted with each vertex. These attributes are based on the heuristics discussed in Chapter 3. Weighted edges indicate the consumed capacity of the transitioning agents using fill bars. The transition from (l_0, t_1) to (l_1, t_1) , for instance, shows that the payload items consumed a quarter of the rover's total transport capacity.

G.4 Comparing temporally expanded network sizes

The temporally expanded network according to Definition 4.3 permits vertices to connect only to a vertex related to the subsequent timepoint. The order of the time-expanded network is $|V| = |T||L|$ and each vertex has at most $|L|$ outgoing edges. Furthermore, the time-expanded network has $|L|$ vertices associated with the last timepoint. These vertices have no outgoing edges. Hence, the set of edges which needs to be considered is $|E| = |L|(|V| - |L|)$.

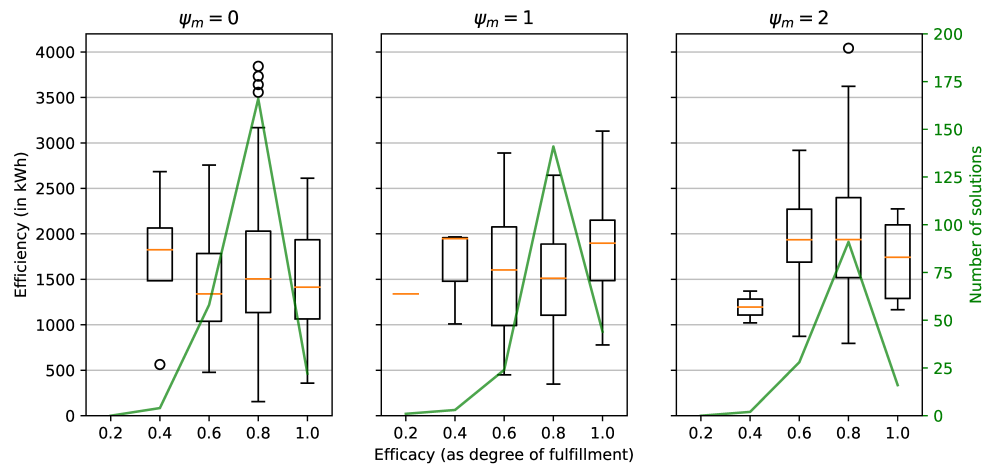
An exhaustive approach can consider each vertex to be connected to any vertex which is associated with any later timepoint. Let $|E_x|$ be the size of this exhaustive version of network, while $|E|$ is the size of the network according to the given definition. Comparing the size of the networks:

$$\begin{aligned}
 |E_x| &\geq |E| \\
 \sum_{i=1}^{|T|} |L|[(|T| - i)|L|] &\geq |L|(|V| - |L|) \\
 |L|^2 \sum_{i=1}^{|T|} (|T| - i) &\geq |L|^2(|T| - 1) \\
 |L|^2(|T|^2 - \sum_{i=1}^{|T|} i) &\geq |L|^2(|T| - 1) \\
 |T|^2 - \sum_{i=1}^{|T|} i &\geq |T| - 1 \\
 |T|^2 - |T| \frac{|T| + 1}{2} &\geq |T| - 1 \\
 |T|^2 - \frac{|T|^2}{2} - \frac{|T|}{2} &\geq |T| - 1 \\
 \frac{|T|^2}{2} - \frac{|T|}{2} &\geq |T| - 1
 \end{aligned}$$

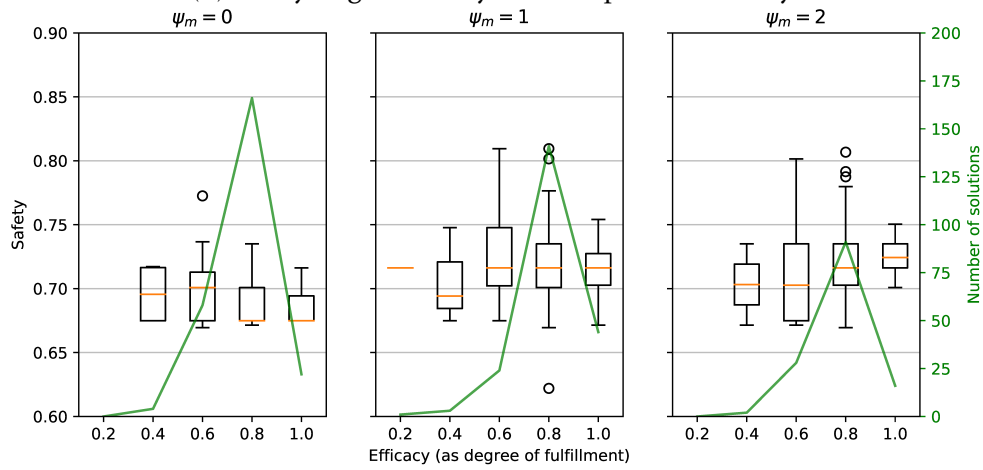
For larger $|T|$ the dominating terms on each side shows, that the exhaustive version of the network has a quadratic relationship between number of timepoints and edges. In contrast, the restricted definition leads to a linear relationship between number of timepoints and edges.

G.5 Details on the space mission example

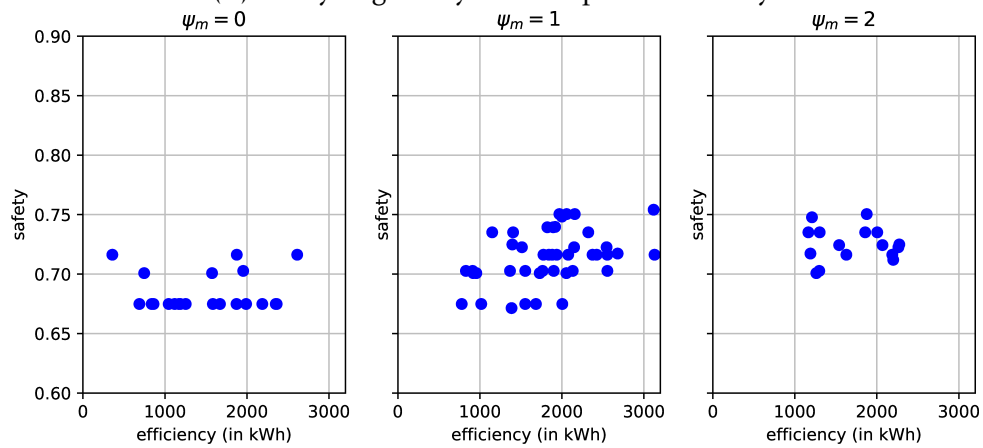
Figure G.2 details the solution landscape for the exemplary space mission in Chapter 4.



(a) Analysing efficiency with respect to efficacy.



(b) Analysing safety with respect to efficacy.



(c) Efficiency with respect to safety, where efficacy = 1.0.

Figure G.2: Analysing the solution landscape of the space mission in Section 4.4.3 with respect to the optimisation objectives.

Appendix H

Operational Infrastructure

H.1 Interaction protocol examples

The FIPA ACL Library implementation (Roehr 2010) provides a set of standard interaction protocol definitions including: brokering, contractNet, dutchAuction, inform, iteratedContractNet and others. Interaction protocols are represented with XML Statechart. A specification file does not contain the full information about an interaction protocol. Default states of the FIPA specification are automatically added to the protocol as part of the provided implementation. Listing H.1 lists the specification without default states. To compare, Listing H.2 includes the default states.

Listing H.1: *The interaction protocol definition.*

```
1 <scxml version="1.0" initial="1">
2   <state id="1">
3     <transition performative="inform" from="initiator" to="B" target="2"/>
4   </state>
5   <state id="2" final="yes">
6   </state>
7 </scxml>
```

Listing H.2: *The interaction protocol definition including default states.*

```
1 <scxml version="1.0" initial="1">
2 <state id="1">
3   <transition performative="inform" from="initiator" to="B" target="2" />
4   <transition performative="not-understood" from="initiator" to="B" target="
    __internal_state:not_understood__" />
5   <transition performative="not-understood" from="B" to="initiator" target="
    __internal_state:not_understood__" />
6   <transition performative="cancel" from="initiator" to="B" target="__internal_state
    :conversation_cancelling__" />
```

```

7   <transition performative="cancel" from="B" to="initiator" target="__internal_state
   :conversation_cancelling__" />
8   <transition performative="failure" from=".*" to="__self__" target="__
   __internal_state:general_failure__" />
9   <state id="2" final="yes" />
10  <state id="__internal_state:conversation_cancel_failure__" final="yes" />
11  <state id="__internal_state:conversation_cancel_success__" final="yes" />
12  <state id="__internal_state:conversation_cancelling__">
13    <transition performative="inform" from=".*" to=".*" target="__internal_state:
   conversation_cancel_success__" />
14    <transition performative="failure" from=".*" to=".*" target="__internal_state:
   conversation_cancel_failure__" />
15  <state id="__internal_state:general_failure__" final="yes" />
16  <state id="__internal_state:not_understood__" final="yes" />
17 </scxml>

```

H.1.1 Distributed locking

Listing H.3: *Interaction protocol to handle the resource probe stage.*

```

1 <scxml version="1.0" initial="1">
2 <state id="1">
3   <!-- request a response -->
4   <transition performative="request" from="initiator" to="B" target="2"/>
5 </state>
6 <state id="2">
7   <transition performative="confirm" from="B" to="initiator" target="3"/>
8 </state>
9 <state id="3" final="yes"/>
10 </scxml>

```

Listing H.4: *Interaction protocol to handle the resource discovery stage.*

```

1 <scxml version="1.0" initial="1">
2 <state id="1">
3   <transition performative="query-if" from="initiator" to="B" target="2"/>
4   <transition performative="inform" from="B" to=".*" target="2"/>
5 </state>
6 <state id="2">
7   <transition performative="inform" from="B" to=".*" target="2"/>
8 </state>
9 <state id="3" final="yes"/>
10 </scxml>

```

Listing H.5: *Interaction protocol to handle the Ricart-Agrawala protocol for locking.*

```

1 <scxml version="1.0" initial="1">
2 <state id="1">
3   <!-- request the lock by broadcasting-->
4   <transition performative="request" from="initiator" to="all" target="2"/>
5 </state>
6 <state id="2">
7   <transition performative="agree" from="all" to="initiator" target="3"/>
8 </state>
9 <state id="3">
10   <transition performative="agree" from="all" to="initiator" target="3"/>
11   <transition performative="confirm" from="initiator" to="owner" target="4" />

```

```
12 </state>
13 <state id="4">
14     <!-- the client provides the information that it released the resource -->
15     <transition performative="disconfirm" from="initiator" to="owner" target="5"
        />
16 </state>
17 <state id="5" final="1" >
18     <transition performative="agree" from="initiator" to=".*" target="5"/>
19 </state>
20 </scxml>
```


Appendix I

Field testing

I.1 Field trial Utah

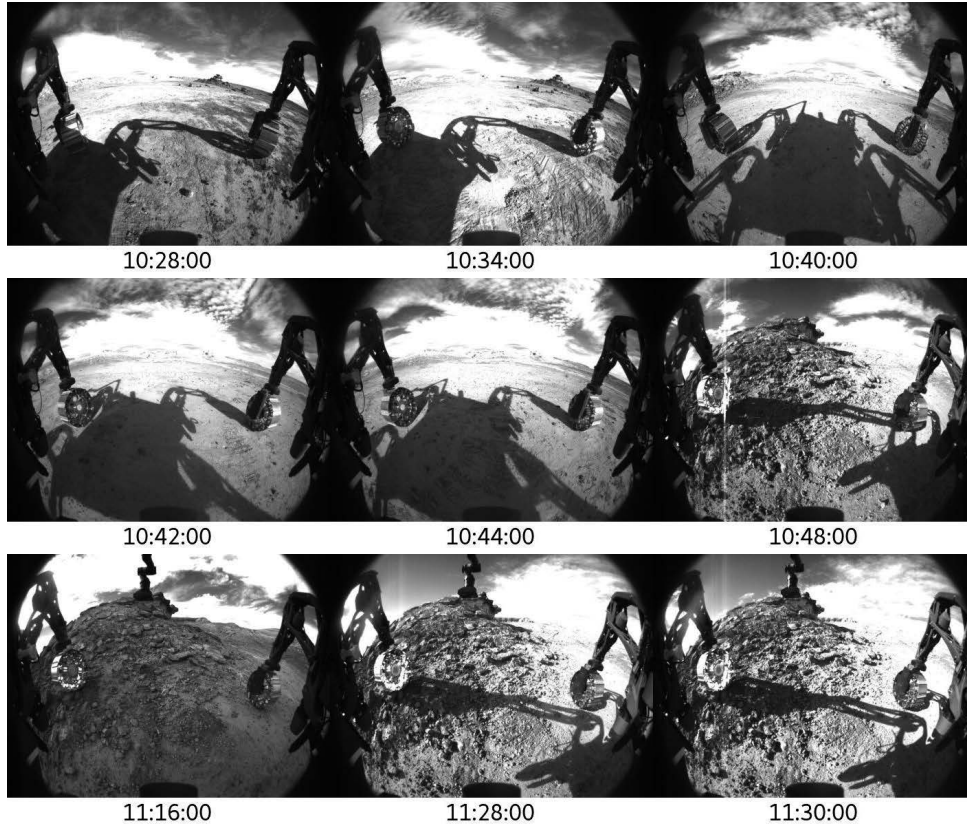


Figure I.1: Key frames to visualise SherpaTT's path for the analogue mission.

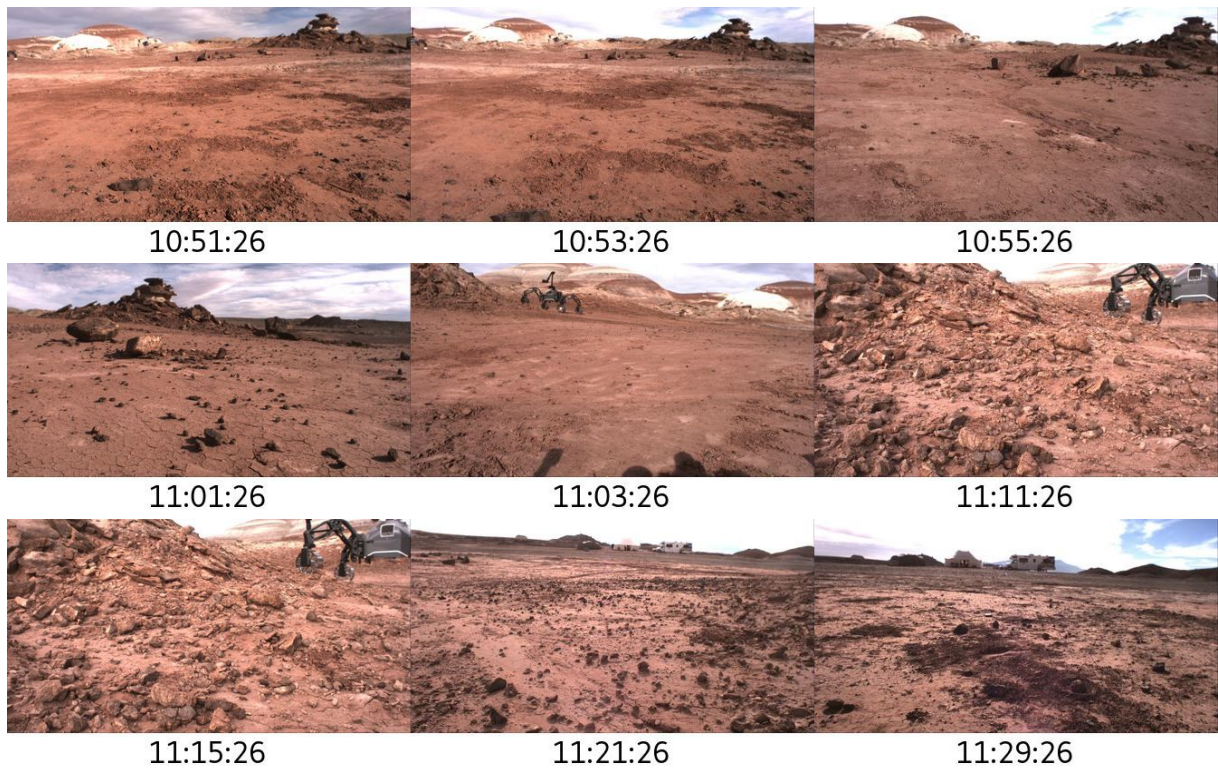


Figure I.2: Key frames to visualise Coyote III's path for the analogue mission.

I.2 Autonomous Mission in Bremen (RH1)

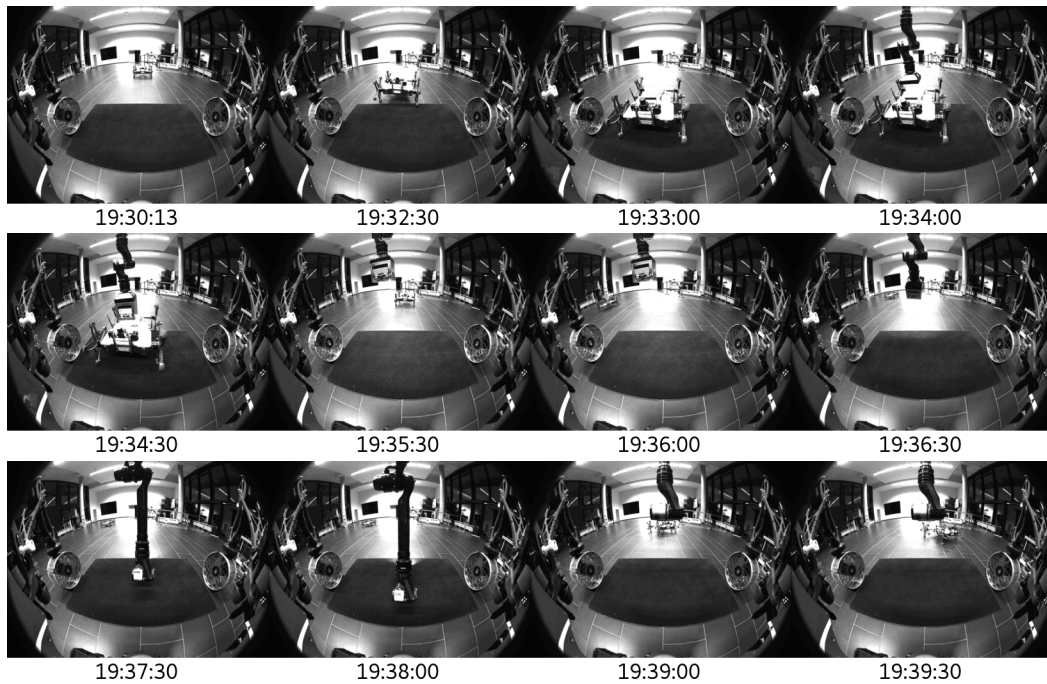


Figure I.3: Key frames to visualise experiment trial FM-V31.

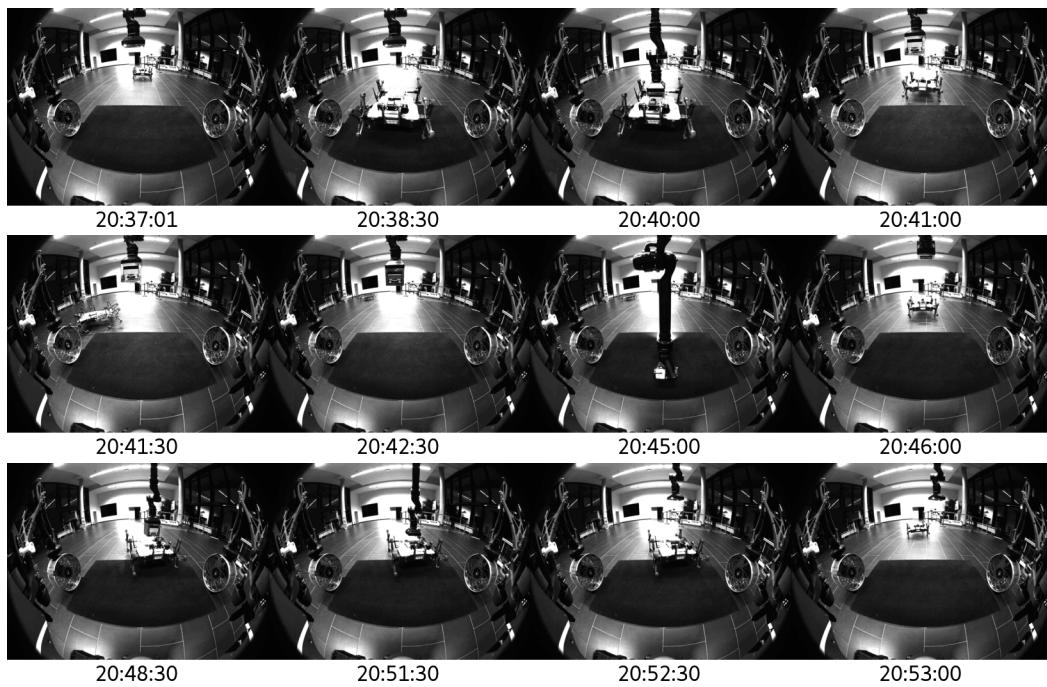


Figure I.4: Key frames to visualise experiment trial FM-V33.

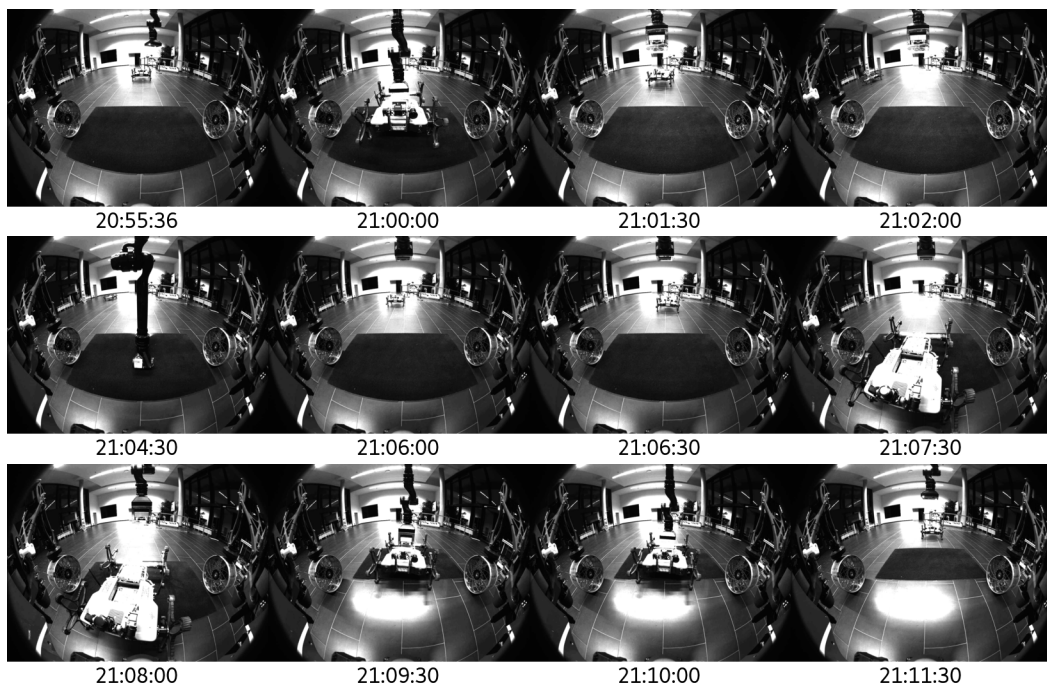


Figure I.5: Key frames to visualise experiment trial FM-V34.

Table I.1: *Trials to incrementally validate autonomous coalition structure changes.*

Trial	Start time	Description
D-01	2017-08-16 17:02	Pickup of a single payload in an outdoor environment
D-02	2017-08-16 17:23	Pickup of a single payload from Coyote III in an outdoor environment in the context of a manually controlled multi-robot mission
D-11	2017-08-30 14:15	Pickup of a single payload from Coyote III in an indoor environment and place back, too low force threshold led to an abort of the placing procedure
D-12	2017-08-30 14:39	Pickup of a single payload from Coyote III in an indoor environment and place back, Coyote III parallel to SherpaTT
D-13	2017-08-30 15:02	Pickup of a single payload from Coyote III in an indoor environment and place back, Coyote III with +45° rotation with respect to SherpaTT
D-14	2017-08-30 15:15	Pickup of a single payload from Coyote III in an indoor environment and place back, Coyote III with +45° rotation with respect to SherpaTT
D-15	2017-08-30 15:25	Pickup of a single payload from Coyote III in an indoor environment and place back, Coyote III with -45° rotation
RS-V11	2018-01-26 18:06	payload pickup and deployment
RS-V21	2018-02-07 17:00	simple pickup docking success, backup of coyote success, sampling starting, Coyote III returns to position with large error, so that visual servoing for placing payload fails
RS-V22	2018-02-07 17:17	first payload pickup successful, but too high forces lead to quick backup of manipulator leading to abort of the sequence
RS-V23	2018-02-07 17:25	first payload pickup successful, manipulator does not move high enough, so that Coyote III cannot backup, manual abort
RS-V24	2018-02-07 17:34	fully successful performance of pickup and backup with Coyote III
FM-I11	2018-02-09 18:08	pickup successful
FM-I12	2018-02-09 18:31	pickup successful, Coyote III moves to destination, placing of module without success
FM-I13	2018-02-09 19:23	pickup successful
FM-I14	2018-02-09 20:02	pickup successful, Coyote III moves to destination
FM-I21	2018-02-12 20:28	pickup successful
FM-I22	2018-02-12 20:48	pickup failed due to imprecision
FM-I23	2018-02-12 20:58	pickup failed due to imprecision
FM-V31	2018-02-13 19:32	successful payload placement, Coyote III pose too inaccurate, placing of sampling module not possible
FM-V32	2018-02-13 20:20	pickup successful, Coyote III approach to waypoint failed
FM-V33	2018-02-13 20:37	complete successful run, Coyote III approaches successfully, final placing takes very long due to an outstretched manipulator arm
FM-V34	2018-02-13 20:56	completion required human intervention to correct for Coyote III misalignments, long time for initial camera convergence, Coyote III pose misaligned after return - out of envelope, pose fixed manually, Coyote III moved too far in, repositioned - placing payload succeeded

Terminology

Capability

The ability (of a robot) to perform a particular activity. For a robot that will correspond to a particular ability to act or sense.

Cardinality

“The number of elements in a set or other grouping, as a property of that grouping.” (Oxford University Press 2018)

Coalition structure

A set of agents can form different combinations of non-overlapping coalitions. A combination of coalitions is called a coalition structure. Example: a set of agents $\{a, b, c\}$, can have one the following coalition structures: $\{\{a\}, \{b\}, \{c\}\}, \{\{a, b\}, \{c\}\}, \{\{a, c\}, \{b\}\}, \{\{a\}, \{b, c\}\},$ or $\{\{a, b, c\}\}.$

Cooperative agents

A set of agents that support each other while aiming to achieve individual and global goals.

Efficacy

“The ability to produce a desired or intended result. “ (Oxford University Press 2018)

Efficiency

The quantified resource usage of a task describes the efficiency for that task. In this thesis, efficiency is equivalent to energy efficiency or rather power consumption.

Functionality

“The quality of being suited to serve a purpose well; practicality.” (Oxford University Press 2018). Used as parent concept for capability and service in the organisation model MoreOrg.

Ontology

In knowledge engineering, an ontology is a network-based representation of knowledge. It encodes information about the world, based on a set of classes, instances, properties thereof and relations between them.

Reconfigurable Systems

A system that is capable of adapting its hardware, in terms of composition and parametrisation of subsystems, and its corresponding software configuration.

Redundancy

“The inclusion of extra components which are not strictly necessary to functioning, in case of failure in other components.” (regarding the meaning in engineering) (Oxford University Press 2018)

Safety

In this thesis context, a robot’s safety is understood as a robot’s probability of survival. The probability of survival is used as optimisation criteria for mission planning.

Service

An activity that can be performed by an agent to serve another agent.

Strategic Flexibility

The strategic use of the flexibility of a system, here robots, in an either proactive or reactive way, combined with an either offensive or defensive intention according to Evans (1991).

Superaddition

Joining two components leads to a greater utility than the sum of each individual component’s utility.

Acronyms

ABox	assertional box
ACC	Agent Communication Channel
ACL	Agent Communication Language
ADL	Action Description Language
apaka	automated packaging for autoproj
ARM	Application Reference Model
ARP	Address Resolution Protocol
CCSDS	Consultative Committee for Space Data Systems
CG	Causal graph
CORBA	Common Open Request Broker Architecture
CPL	CRAM Plan Language
CRAM	Cognitive Robot Abstract Machine
CSP	constraint-satisfaction problem
CVRP	Capacitated VRP
DARP	Dial-a-Ride Problem
DDS	Data Distribution Service
DGPS	Differential Global Positioning System
DL	Description Logic
DNS	Domain Name System
DoF	Degree of Freedom
DSD	Distributed Service Directory Service
DSN	Deep Space Network
DTG	Domain Transition Graph

ECSS	European Collaboration for Space Standardization
EMI	electromechanical interface
FIPA	Foundation for Intelligent Physical Agents
FOL	First Order Language
FRM	Functional Reference Model
Gecode	Generic constraint programming framework
GQR	Generic Qualitative Reasoner
HTN	Hierarchical Task Network
HWMP	Hybrid Wireless Mesh Protocol
IA	interval algebra
iBLOCK	intelligent Building Block
iBOSS	intelligent Building Block for On-Orbit Satellite Servicing
IP	Internet Protocol
ISEG	International Space Exploration Coordination Group
iSSI	intelligent Space System Interface
JADE	Java Agent DEvelopment Framework
KIF	Knowledge Integration Framework
LDSB	Lightweight Dynamic Symmetry Breaking
LP	Linear Program
MDHFVRPTW	Multi-depot heterogeneous fleet VRP with time windows
MMCF	multi-commodity min-cost flow problem
MOISE+	Model of Organisation for multiI-agent SystEms
MoreOrg	Model for Reconfigurable Multi-Robot Organisations
MTS	message-transport service
OMACS	Organisation Model for Adaptive Computational Systems
OMNI	Organisational Model for Normative Institutions
ORA	Ontologies for Robotics and Automation
ORM	Operations Reference Model

OSI	Open System Interconnect
OWL	Web Ontology Language
OWL 2	Web Ontology Language 2
PA	point algebra
PDDL	Planning-Domain Definition Language
RASDS	Reference Architecture for Space Data Systems
RBox	role box
RDF	Resource Description Framework
Rock	Robot Construction Kit
ROS	Robot Operating System
SDS	Service Directory Service
SIMA	Symmetrical Interface Manipulator
SLAM	simultaneous localization and mapping
STN	simple temporal network
stqe	spatio-temporally qualified expression
str	spatio-temporal requirement
SWRL	Semantic Web Rule Language
TBox	terminological box
TCN	temporal constraint network
TCP	Transmission Control Protocol
TemPl	Temporal Planning for Reconfigurable Multi-Robot Systems
TSP	Travelling Salesman Problem
UDP	User Datagram Protocol
UDT	UDP-based Data Transfer Protocol
VLNS	Very Large Neighbourhood Search
VRP	Vehicle Routing Problem
VRPMSs	VRP with multiple synchronization constraints
VRPPD	VRP with Pick-up and Delivery
VRPTT	VRP with Trailers and Transshipments

VRPTW	VRP with Time Windows
W3C	World Wide Web Consortium

List of Figures

1.1	Main chapters of this thesis with respective references to authored publications.	7
2.1	Components of the robotic team in LUNARES.	19
2.2	Male and female electromechanical interfaces (EMIs) in the project RIMRES.	20
2.3	Interfaces to connect the mobile systems.	21
2.4	Docking variants of the mobile systems CREX and Sherpa in the project RIMRES.	21
2.5	Mobile and immobile agents in the project TransTerra.	24
2.6	Variants of electromechanical interfaces in the project TransTerra.	24
2.7	Schematic description of an incremental design of planetary space missions using reconfigurable multi-robot systems.	26
2.8	An available set of atomic agents and a subset of composite agents that can be formed by combining different atomic agents.	29
3.1	Combinatorial explosion for an example scenario with an upper link bound of 3 (left), and the corresponding composite agent types for link space and agent space (right) (both y-axis have the same logarithmic scale).	46
3.2	Upper bound in agent space: exact and accumulated composite agent types.	46
3.3	Comparison of performance between Twiddle and the generative limited combination approach as developed in this thesis. The number of element types is fixed to five, while the number of instances per element type homogeneously varies. Y-axis with logarithmic scale.	46
3.4	Characterisation of the robotic organisation at its different decomposition levels.	48
3.5	General architecture of the organisation model.	49
3.6	Class hierarchy excerpt of the base ontology.	52
3.7	Property hierarchies of the base ontology.	53
3.8	Organisation model excerpt of a DL-based description of an atomic agent concept <i>ARobot</i> . The example associates a functionality named <i>LocationImageProvider</i> with the agent concept, reflecting the ability to take images at different locations.	53
3.9	Knowledge base example to account for interface compatibility.	55
3.10	Filtering suitable agents out of the set of available agents requires the application of bounds and filtering stages.	56
3.11	One feasible link structure (out of many) for a composite agent after solving the assignment problem. Edges are annotated with the interface corresponding to the source vertex. Agent models and interfaces are related to the reference system described in Section 2.2.	58

3.12	The performance of feasibility checking for a composite agent depends upon the selection strategy for variable assignment and the available set of interfaces. Note that different y-axis ranges are used between the upper and lower plots to achieve a more detailed visualisation.	60
3.13	Example application of the functional saturation bound.	65
3.14	The formulas depicted on the right hand side serve as examples for property inference which is based a property value summation.	68
3.15	Schematic of a system composition consisting of three resource types: a,b,c, where the ratio from required to available is for a 1:3, for b 1:1, and for c 2:8. . .	73
4.1	Operations as part of the domain definition, for a set of atomic agent A and location variables l, l_s, l_t (based on Figure 4 in (Roehr and F. Kirchner 2016)). . .	87
4.2	A temporal constraint describes the lower and upper bound for the time distance between two timepoints, here a transition between timepoint t_0 and t_1 requires at least 10 and at most 30 (time units).	89
4.3	Comparison of the consistency checking for a temporal constraint network $N = (V, E)$ with a predefined number of timepoints $ V $, and constraints between every two timepoints so that $ E = 0.5 \cdot V $	90
4.4	Timeline-based illustration of an example mission. Agents are assigned in space and time, while the need for functionality is indicated by pictograms above the top timeline.	91
4.5	A mission specification example consisting of three spatio-temporal requirements: $(\emptyset, \bar{C}_0)@(l_0, [t_0, t_1])$ and $(\emptyset, \bar{C}_1)@(l_1, [t_0, t_1])$ representing the initial state and $(\mathcal{F}_0, \{\hat{a}_0, 3\})@(l_1, [t_2, t_3])$, where l_0, l_1 are location variables and t_0, \dots, t_3 are timepoint variables	94
4.6	A mission solution representation: each vertex in the network represents a space-time tuple describing the agent type assignments. Fillbars on the capacitated edges indicate the total consumed capacity of the transitioning agent. Colour-coded boxes represent unique agent roles: green for fulfilled requirements, grey for an agent role presence without requirement.	95
4.7	Schematic view illustrating the high-level components of TemPl's algorithm to generate and process solution candidates.	99
4.8	Example for a time expanded network (based on Figure 6 in (Roehr and F. Kirchner 2016)): a legal path (solid green line) and an illegal path (dotted red line). .	100
4.9	Matrix-based representation of agent type cardinality constraints. Columns and rows and annotated with the corresponding meaning of the variable indices. . .	102
4.10	Internal CSP model to deal with agent role constraints. Columns relating to the same agent type are interchangeable. Constraints are applied: (a) to items with the same column index to account for unary resource constraints and mutually exclusive spatio-temporal requirements (red), (b) to items with the same row index , and same agent type to set a lower agent type cardinality bound to fulfil each requirement (blue), and (c) to set an upper agent type cardinality bound for mutually exclusive spatio-temporal requirements (green).	103
4.11	Example of a matrix based interpretation of a timeline corresponding to a time-expanded network which is constructed for four locations and four timepoints. Superscript values (in grey) indicate the CSP variable index, and value assignments (in blue) are the index of the variable corresponding to the next space-time point in the timeline.	106

4.12	Flawed solution with the missed fulfilment of a requirement (red), while some requirements are fulfilled (green). All assignment of resources without requirement are marked in grey.	112
4.13	Solution for a simple mission with $\psi_m = 2$	112
4.14	No suitable coalition structure can be found for a transition, leading to an edge flaw (red).	114
4.15	Possible locations with science goal defined in the project TransTerra for a lunar robotic exploration mission (extracted from Workpackage Document E2100.1). .	115
4.16	Solution landscape compared for different bound settings. Black star-shaped markers identify the current local best solutions, where the cost function is parametrised with $\alpha = -1.0$, $\beta = 100.0$, and $\epsilon = 10.0$	116
4.17	Performance and LP problem size comparison with respect to ψ_m for the example space mission.	117
4.18	Identified solution for the exemplary space mission after 20 min of search ($\psi_m = 2, \psi_{-m} = 0$)	122
5.1	The basic structure of the FRM as described by Putz and Elfving (1991, p. 94). .	129
5.2	The FIPA-based messaging infrastructure for a multi-robot system (adapted from (Roehr and Herfert 2016)).	132
5.3	The nominal message flow. Note that for each agent there might be multiple components which act as sender or receiver.	135
5.4	Error reporting back to a message's sender is automatically triggered by an MTS, when the message delivery fails.	135
5.5	Channel-based communication with MTS.	136
5.6	Component network structure to support a decentralised publish-subscribe based communication for sensor data inter-robot sensor-data/telemetry exchange. . . .	137
5.7	Net message and envelope overhead for a message with the content field containing 1 byte and 99 bytes in the remaining message fields as listed in Table 5.1 (adapted by permission from Springer Customer Service Centre GmbH: Springer Nature (Roehr and Herfert 2016) © Springer International Publishing Switzerland 2016).	138
5.8	Encoding and decoding performances. The colour coding indicates the performance of encoding in dark green and decoding in light green (adapted from (Roehr and Herfert 2016)).	139
5.9	Wireless communication network setups which have been used for comparison of a centralised vs. a mesh-based communication setup.	141
5.10	Comparison between a central AP-based solution and a mesh-based solution implemented with OM2P routers using the operation system openWRT with three communicating agents.	141
5.11	Statemachine for performing an action on a robot	143
5.12	Three separate interaction protocols that are part of the (extended) Ricart-Agrawala's algorithm; A: probing of inter-dependant agents, B: discovery of the resource owner (owner information is sent as broadcast), C: the message flow when performing the actual locking algorithm. Failure handling is embedded into default transitions (which are not depicted) and S denotes the conversation initiator's role and R the recipient's role (reprinted by permission from Springer Customer Service Centre GmbH: Springer Nature (Roehr and Herfert 2016) © Springer International Publishing Switzerland 2016).	144

5.13 Interdependencies of bundles: hierarchical function design is achieved by modularising functionalities. Component network templates and generically implemented actions are collected in so-called bundles. This approach leads to standardisation and facilitates the reuse of component network templates.	145
6.1 Left hand side: general overview over the testing site in Utah with the respective main landmarks and approximate distances (Source: manually annotated Imagery ©2018 Google, Map data ©2018 Google), right hand side: view onto the site from additional perspectives.	153
6.2 The general mission control setup, including a local control centre which redirects the data streams between mission control and robots.	153
6.3 Reuse and setup of the demultiplexing and multiplexing infrastructure components for the inter-site communication communication.	154
6.4 The assumed positions of the robots. Progression of time is encoded in greyscale. Approximate location coordinates: local control centre at (0,-35), Landmark 1 at (30,15), Landmark 2 at (20,-15).	155
6.5 Inter-robot and travelled distance for the analogue mission.	156
6.6 Communication from mission control (Bremen) to local control centre (Utah). .	156
6.7 Communication from local control centre (Utah) to mission control (Bremen). .	157
6.8 Communication between the participating robots using the distributed communication infrastructure.	158
6.9 Communication volumes for inter-site and inter-robot communication. The comparison between letter and actual message content size for the inter-robot communication shows the minimal protocol overhead in a real application.	159
6.10 Camera images augmented with the marker detection result during the approach of the end effector to pickup a payload item. Images (a) and (b) are taken during the same approach. Images (c) and (d) refer to an approach at later time of day, indicated by the shadows of the interface pins.	159
6.11 Initial and final marker setup for the EMI.	160
6.12 Soil sampling in Utah using a dedicated soil sampling module.	161
6.13 Soil sampling as example for revising the design of a superadditive functionality.	162
6.14 Baseline mission for a multi-robot sequence involving reconfiguration, soil sampling and sample return. The consideration of timepoints $t_{1.5}$ and $t_{5.5}$ allows Coyote III to deviate from a straight line path when transitioning between l_0 and l_1	162
6.15 Mission specification of the baseline mission.	162
6.16 A solution computed by TemPl for the baseline mission.	163
6.17 Position and force profile during the placing of a sampling module (FM-V33). .	165
6.18 Task state transitions for payload pickup and payload placement during experiment FM-V33.	166
6.19 Exchange of a sampling module during experiment (FM-V33), pick and place from two perspectives.	166
6.20 Alignment characteristics over all pick and place actions.	167
6.21 Alignment characteristics for picking a payload.	167
6.22 Alignment characteristics for placing a payload.	167
6.23 Coyote III camera view which allows to detect the position of SherpaTT with the help of an artificial marker (FM-V33).	169

6.24	Robot-to-robot communication during a fully autonomous mission sequence (FM-V33).	171
6.25	Robots' assumed poses during the sample mission - Coyote III initially at (0,0), SherpaTT remains at (3,0). Target waypoints are numbered according to the resulting sequence. Note that the orientation of Coyote III does not change, e.g., when moving from waypoint 1 to 2, since the robot is driving backwards (see Table 6.2).	173
G.1	A mission solution representation.	228
G.2	Analysing the solution landscape of the space mission in Section 4.4.3 with respect to the optimisation objectives.	230
I.1	Key frames to visualise SherpaTT's path for the analogue mission.	235
I.2	Key frames to visualise Coyote III's path for the analogue mission.	236
I.3	Key frames to visualise experiment trial FM-V31.	236
I.4	Key frames to visualise experiment trial FM-V33.	237
I.5	Key frames to visualise experiment trial FM-V34.	237

List of Tables

2.1	Dimensions of change (adapted with permission from (Roehr, Cordes, and F. Kirchner 2014) ©2013 Wiley Periodicals, Inc.).	10
2.2	Characterisation of manoeuvres according to Evans (1991).	11
2.3	Distinguishing properties of robotic system types: positive attributes are marked in green, negative in red, neutral in yellow, and in purple a mix of positive and negative attributes.	14
2.4	Autonomy levels for nominal mission operation according to the ECSS (European Cooperation for Space Standardization 2005).	27
2.5	Autonomy levels for failure handling according to the ECSS (European Cooperation for Space Standardization 2005).	28
3.1	Language features of the DL <i>SHOIQ</i> (Krötzsch, Simancik, and Ian Horrocks 2012).	41
3.2	Notation for ontology description based on (Krötzsch, Simancik, and Ian Horrocks 2012).	42
3.3	Setup of the multi-robot scenario.	45
3.4	Requirements for the given multi-robot scenario.	45
3.5	Class descriptions.	51
3.6	A heterogeneous reconfigurable multi-robot team.	64
3.7	Atomic agent type properties.	69
3.8	Resources of the agent type <i>SherpaTT</i> which are relevant to provision the functionality <code>LocationImageProvider</code>	74
4.1	Intra-route constraints in VRPs as described by Toth and Vigo (2014).	84
4.2	Inter-route constraints in VRPs as described by Toth and Vigo (2014) and Drexler (2012, 2013).	84
4.3	Predicates as part of the domain definition.	86
4.4	Temporal constraints for a mission $\mathcal{M} = \langle \widehat{A}, STR, \mathcal{X}, \mathcal{OM}, T, L \rangle$	96
4.5	Model constraints, where $S \subseteq STR$ and $A_s^{\hat{a}}$ represents the general agent type requirement of $s \in S$	96
4.6	Functionality and property constraints.	97
4.7	Heuristics for cost computation on the solution \mathcal{M}_s for a mission \mathcal{M}	98
4.8	Search statistics for a simple mission with minimal and extended agent assignment bounds.	113
4.9	Solution properties for a simple mission with minimal and extended agent assignment bounds.	113

5.1	Structure of a message according to the FIPA Standards (Foundation for Intelligent Physical Agents 2002b).	127
5.2	The essential network technologies involved in the communication layer, sorted according to the seven layers of the OSI model (ISO/IEC 1996).	133
5.3	FIPA representations.	134
5.4	Extract of a content language.	142
5.5	Operation modi for reconfiguration.	146
5.6	Action sequence for an uncooperative reconfiguration.	147
5.7	Action sequence for a cooperative join.	148
6.1	Experiments with corresponding descriptions.	164
6.2	High level action sequence for the mission in experiment FM-VXX.	168
6.3	Communication properties for the autonomous operation of an exemplary sample return mission (FM-V33).	170
A.1	Properties of agent types.	187
A.2	Properties of agent types.	188
A.3	Properties of agent types.	188
A.4	Suggested classification scheme for interfaces in reconfigurable multi-robot systems using the DFKI EMI, ATRON and iBOSS as illustration examples.	189
D.1	Pictograms for functions.	213
D.2	Pictograms for robotic systems.	214
G.1	Description of XML Tags in the mission specification.	223
I.1	Trials to incrementally validate autonomous coalition structure changes.	238

Bibliography

- Achterberg, Tobias et al. (2008). “Constraint Integer Programming: A New Approach to Integrate CP and MIP”. In: *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*. Ed. by Laurent Perron and Michael A. Trick. Vol. 5015 LNCS. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 6–20. ISBN: 354068154X. DOI: 10.1007/978-3-540-68155-7_4.
- Ahuja, Ravindra K., Thomas L. Magnanti, and James B. Orlin (1993). *Network Flows: Theory, Algorithms, and Applications*. Michigan, US: Prentice Hall, p. 846.
- Alexander, R D (Nov. 1974). “The Evolution of Social Behavior”. In: *Annual Review of Ecology and Systematics* 5.1, pp. 325–383. ISSN: 0066-4162. DOI: 10.1146/annurev.es.05.110174.001545.
- Allen, James F (1990). “Maintaining knowledge about temporal intervals”. In: *Readings in qualitative reasoning about physical systems* 26.11, pp. 361–372. ISSN: 00010782. DOI: 10.1145/182.358434.
- Aoki, Takeshi, Yuki Murayama, and Shigeo Hirose (May 2011). “Mechanical design of three-wheeled lunar rover ‘Tri-Star IV’”. In: *2011 IEEE International Conference on Robotics and Automation*. IEEE, pp. 2198–2203. ISBN: 978-1-61284-386-5. DOI: 10.1109/ICRA.2011.5980180.
- Baca, José et al. (July 2014). “ModRED: Hardware design and reconfiguration planning for a high dexterity modular self-reconfigurable robot for extra-terrestrial exploration”. In: *Robotics and Autonomous Systems* 62.7, pp. 1002–1015. ISSN: 09218890. DOI: 10.1016/j.robot.2013.08.008.
- Bäckström, Christer and Bernhard Nebel (1995). “Complexity Results for SAS+ Planning”. In: *Computational Intelligence* 11.4, pp. 625–655. ISSN: 0824-7935. DOI: 10.1111/j.1467-8640.1995.tb00052.x.
- Bader, Franz et al., eds. (2007). *The Description Logic Handbook*. 2nd ed. Cambridge, UK: Cambridge University Press. ISBN: 978-0521-87625-4.
- Bajracharya, Max, Mark W. Maimone, and Daniel Helmick (2008). “Autonomy for Mars Rovers: Past, present, and future”. In: *Computer* 41.12, pp. 44–50. ISSN: 00189162. DOI: 10.1109/MC.2008.479.
- Bartlett, Paul, David Wettergreen, and William (Red) L Whittaker (Feb. 2008). “Design of the Scarab Rover for Mobility and Drilling in the Lunar Cold Traps”. In: *8th International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS 2008)*. Hollywood, LA, USA.
- Bartsch, Sebastian et al. (2010). “Performance Evaluation of an Heterogeneous Multi-Robot System for Lunar Crater Exploration”. In: *10th International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS)*. ESA, pp. 30–37.
- BBN Technologies (2004). *Cougaar Architecture Document*. Tech. rep. 11.4 (23 December 2004).

- Beetz, Michael, Ulrich Klank, et al. (Oct. 2011). "Robotic roommates making pancakes". In: *2011 11th IEEE-RAS International Conference on Humanoid Robots*. Bled, Slovenia: IEEE, pp. 529–536. ISBN: 978-1-61284-866-2. DOI: 10.1109/Humanoids.2011.6100855.
- Beetz, Michael, Lorenz Mösenlechner, and Moritz Tenorth (Oct. 2010). "CRAM: A Cognitive Robot Abstract Machine for everyday manipulation in human environments". In: *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, pp. 1012–1017. ISBN: 978-1-4244-6674-0. DOI: 10.1109/IRoS.2010.5650146.
- Beldiceanu, Nicolas and Sophie Demassey (2018). *Global Constraint Catalog: Cin relation*. Available at: <http://www.emn.fr/x-info/sdemasse/gccat/Cin{ }relation.html>, (Accessed: 06 March 2018).
- Bellifemine, Fabio, Agostino Poggi, and Giovanni Rimassa (1999). "JADE—A FIPA-compliant agent framework". In: *Proceedings of International Conference and Exhibition on the Practical Application of Intelligent Agents and Multi-Agents (PAAM)*.
- Bellwood, David R., Terry P. Hughes, and Andrew S. Hoey (Dec. 2006). "Sleeping functional group drives coral-reef recovery". In: *Current biology : CB* 16.24, pp. 2434–9. ISSN: 0960-9822. DOI: 10.1016/j.cub.2006.10.030.
- Belmonte, Matthew (1996). *Twiddle algorithm implementation*. Available at: <http://www.netlib.no/netlib/toms/382>, (Accessed: 22 April 2018).
- Birnschein, Timo et al. (2014). "Enhancing Mobility Using Innovative Technologies and Highly Flexible Autonomous Vehicles". In: pp. 49–58. DOI: 10.1007/978-3-319-08087-1_5.
- Björkelund, Anders et al. (2011). "Knowledge and skill representations for robotized production". In: *IFAC Proceedings Volumes (IFAC-PapersOnline)* 18.PART 1, pp. 8999–9004. ISSN: 14746670. DOI: 10.3182/20110828-6-IT-1002.01053.
- Blum, Avrim L. and Merrick L. Furst (1997). "Fast planning through planning graph analysis". In: *Artificial Intelligence* 90.1-2, pp. 281–300. ISSN: 00043702. DOI: 10.1016/S0004-3702(96)00047-1.
- Brandt, David, David Johan Christensen, and Henrik Hautop Lund (Aug. 2007). "ATRON Robots: Versatility from Self-Reconfigurable Modules". In: *2007 International Conference on Mechatronics and Automation*. IEEE, pp. 26–32. ISBN: 978-1-4244-0827-6. DOI: 10.1109/ICMA.2007.4303511.
- Brinkmann, Wiebke, Florian Cordes, et al. (Mar. 2018). "Modular payload-items for payload-assembly and system enhancement for future planetary missions". In: *2018 IEEE Aerospace Conference*. Big Sky, Montana, USA: IEEE, pp. 1–12. ISBN: 978-1-5386-2014-4. DOI: 10.1109/AERO.2018.8396614.
- Brinkmann, Wiebke, Thomas M. Roehr, et al. (Mar. 2018). "Design and evaluation of an end-effector for a reconfigurable multi-robot system for future planetary missions". In: *2018 IEEE Aerospace Conference*. Big Sky, Montana, USA: IEEE, pp. 1–10. ISBN: 978-1-5386-2014-4. DOI: 10.1109/AERO.2018.8396415.
- Brown, G. Malcolm et al. (1954). "The Circulation in Cold Acclimatization". In: *Circulation* 9.6, pp. 813–822. DOI: 10.1161/01.CIR.9.6.813.
- Brown, R.G. and J.S. Jennings (1995). "A pusher/steerer model for strongly cooperative mobile robot manipulation". In: *International Conference on Intelligent Robots and Systems*, pp. 562–568. DOI: 10.1109/IRoS.1995.525941.
- Burroughes, Guy (2017). "Reconfigurable Autonomy for Future Planetary Rovers". Doctoral dissertation. University of Surrey.
- Camazine, Scott et al. (2001). *Self-organization in biological systems*. 1st ed. Princeton, NJ, USA: Princeton University Press. ISBN: 9780691116242. DOI: 10.1134/S1062359012020069.

- CCSDS MOIMS SM&C and IEEE FIPA Comparison (2007). Available at: <http://cwe.ccsds.org/moims/docs/MOIMS-SMandC/Documents/SMandCandFIPA/CCSDSMCFIPAComparisonIssue1-0.zip>, (Accessed: 12 June 2018).
- Chase, Phillip J. (1970). "Algorithm 382: combinations of M out of N objects [G6]". In: *Communications of the ACM* 13.6, p. 368. ISSN: 00010782. DOI: 10.1145/362384.362502.
- Chen, Bo, Harry H Cheng, and Joe Palen (Dec. 2006). "Mobile-C: a mobile agent platform for mobile C-C++ agents". In: *Software: Practice and Experience (Software Pract Ex)* 36.15, pp. 1711–1733. ISSN: 0038-0644. DOI: 10.1002/spe.v36:15.
- Chennareddy, S. Sankhar Reddy, Anita Agrawal, and Anupama Karuppiah (2017). "Modular Self-Reconfigurable Robotic Systems: A Survey on Hardware Architectures". In: *Journal of Robotics* 2017. ISSN: 16879619. DOI: 10.1155/2017/5013532.
- Christensen, Anders Lyhne, Rehan O'Grady, and Marco Dorigo (Dec. 2007). "Morphology control in a multirobot system". In: *Robotics & Automation Magazine, IEEE* 14.4, pp. 18–25. ISSN: 1070-9932. DOI: 10.1109/M-RA.2007.908970.
- Conitzer, Vincent and Tuomas Sandholm (2006). "Complexity of constructing solutions in the core based on synergies among coalitions". In: *Artificial Intelligence* 170.6-7, pp. 607–619. ISSN: 00043702. DOI: 10.1016/j.artint.2006.01.005.
- Cordes, Florian et al. (Jan. 2011). "LUNARES: Lunar crater exploration with heterogeneous multi robot systems". In: *Intelligent Service Robotics* 4.1, pp. 61–89. ISSN: 18612776. DOI: 10.1007/s11370-010-0081-4.
- Cormen, Thomas H. et al. (2009). *Introduction to Algorithms*. 3rd ed. Vol. 44. 2. MIT Press, xvii, 1028 p. ISBN: 978-0-262-03384-8. DOI: 10.1017/CB09781107415324.004. arXiv: arXiv:1011.1669v3.
- D'Alessandro, Angelo et al. (Oct. 2016). "AltitudeOmics: Red Blood Cell Metabolic Adaptation to High Altitude Hypoxia". In: *Journal of Proteome Research* 15.10, pp. 3883–3895. ISSN: 1535-3893. DOI: 10.1021/acs.jproteome.6b00733.
- Davey, Jay, Ngai Kwok, and Mark Yim (Oct. 2012). "Emulating self-reconfigurable robots - design of the SMORES system". In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, pp. 4464–4469. ISBN: 978-1-4673-1736-8. DOI: 10.1109/IRoS.2012.6385845.
- Dechter, Rina (2003). "Temporal Constraint Networks". In: *Constraint Processing*. Ed. by Rina Dechter. The Morgan Kaufmann Series in Artificial Intelligence. San Francisco: Morgan Kaufmann. Chap. 12, pp. 333–362. ISBN: 978-1-55860-890-0. DOI: 10.1016/B978-155860890-0/50013-X.
- DeLoach, Scott A. (2009). "OMACS: A Framework for Adaptive, Complex Systems". In: *Handbook of Research on Multi-Agent Systems*. IGI Global, pp. 76–104. DOI: 10.4018/978-1-60566-256-5.ch004.
- DeLoach, Scott A. and Valeriy A. Kolesnikov (2006). "Using Design Metrics for Predicting System Flexibility". In: *Fundamental Approaches to Software Engineering*. Ed. by Luciano Baresi and Reiko Heckel. Vol. 3922. 0347545. Springer Berlin Heidelberg, pp. 184–198. DOI: http://dx.doi.org/10.1007/11693017_15.
- DeLoach, Scott A., Walamitien H. Oyen, and Eric T. Matson (2008). "A capabilities-based model for adaptive organizations". In: *Journal of Autonomous Agents and Multiagent Systems* 16.1, pp. 13–56. DOI: 10.1007/s10458-007-9019-4.A.
- Desrosiers, Jacques and Marco E. Lübbecke (2005). "A primer in column generation". In: *Column Generation* 3, pp. 1–32. DOI: 10.1007/0-387-25486-2_1.

- Dezső, Balázs, Alpár Jüttner, and Péter Kovács (July 2011). “LEMON – an Open Source C++ Graph Template Library”. In: *Electronic Notes in Theoretical Computer Science* 264.5, pp. 23–45. ISSN: 15710661. DOI: 10.1016/j.entcs.2011.06.003.
- DFKI GmbH Robotics Innovation Center (2018). *D-Rock: Model, methods and tools for the model based software development of robots*. Available at: <https://robotik.dfki-bremen.de/en/research/projects/d-rock.html>, (Accessed: 3 January 2018).
- Dignum, Frank (2001). “Agents, Markets, Institutions, and Protocols”. In: *Agent Mediated Electronic Commerce. Lecture Notes in Computer Science, vol 1991*. Ed. by Frank Dignum and C. Sierra. Springer, Berlin, Heidelberg, pp. 98–114. ISBN: 3140247370. DOI: 10.1007/3-540-44682-6_6.
- Dignum, Virginia, ed. (2009). *Handbook of Research on Multi-Agent Systems: Semantics and Dynamics of Organizational Models*. IGI Global. ISBN: 9781605662572.
- Dignum, Virginia, Frank Dignum, and John-Jules Meyer (2004). “An agent-mediated approach to the support of knowledge sharing in organizations”. In: *The Knowledge Engineering Review* 19.2, pp. 147–174.
- Dorigo, Marco et al. (2005). “The SWARM-BOTS Project”. In: *Swarm Robotics*. Springer Verlag, pp. 31–44. DOI: 10.1007/978-3-540-30552-1_4.
- Drexler, Michael (Aug. 2012). “Synchronization in Vehicle Routing—A Survey of VRPs with Multiple Synchronization Constraints”. In: *Transportation Science* 46.3, pp. 297–316. ISSN: 0041-1655. DOI: 10.1287/trsc.1110.0400.
- (2013). “Applications of the vehicle routing problem with trailers and transshipments”. In: *European Journal of Operational Research* 227.2, pp. 275–283. DOI: 10.1016/j.ejor.2012.12.015.
- Dunin-Keplicz, B, Alina Strachocka, and R Verbrugge (2011). “Modeling Deliberation in Teamwork”. In: *Proceedings of the Tenth Symposium on Logical Formalizations of Commonsense Reasoning*. Ed. by E. Davis, P. Doherty, and E. Erdem. AAAI Spring Symposium Series, pp. 114–118.
- Dunin-Keplicz, Barbara Maria and Rineke Verbrugge (2010). *Teamwork in Multi-Agent Systems: A Formal Approach*. 1st. Wiley Publishing.
- Eich, Markus et al. (2014). “Towards coordinated multirobot missions for lunar sample collection in an unknown environment”. In: *Journal of Field Robotics* 31.1, pp. 35–74. ISSN: 15564959. DOI: 10.1002/rob.21491.
- Elkind, Edith, Talal Rahwan, and Nicholas R Jennings (2013). “Computational Coalition Formation”. In: *Multiagent Systems*. Ed. by Gerhard Weiss. 2nd ed. MIT Press. Chap. 8, pp. 329–380.
- ESA (2018a). *Building a lunar base with 3D printing*. Available at: http://www.esa.int/Our_Activities/Space_Engineering_Technology/Building_a_lunar_base_with_3D_printing, (Accessed: 4 April 2018).
- (2018b). *Mars Express*. Available at: <http://blogs.esa.int/mex/2012/08/05/time-delay-between-mars-and-earth>, (Accessed: 5 April 2018).
- Estlin, Tara et al. (Apr. 2007). “Increased Mars Rover Autonomy using AI Planning, Scheduling and Execution”. In: *Proceedings 2007 IEEE International Conference on Robotics and Automation*. IEEE, pp. 4911–4918. ISBN: 1-4244-0602-1. DOI: 10.1109/ROBOT.2007.364236.
- European Cooperation for Space Standardization (2005). *ECSS-E-70-11A*. Available at: <http://www.ecss.nl/wp-content/uploads/standards/ecss-e/ECSS-E-70-11A5August2005.pdf>, (Accessed: 4 April 2018).
- (2008a). *ECSS-E-ST-70-32-C*. Available at: <http://www.ecss.nl/wp-content/uploads/standards/ecss-e/ECSS-E-ST-70-32C31July2008.pdf>, (Accessed: 4 April 2018).

- (2008b). *Space project management: Risk management*. Available at: <http://www.ecss.nl/wp-content/uploads/standards/ecss-m/ECSS-M-ST-80C31July2008.pdf>, (Accessed: 24 April 2018).
- Evans, J. Stuart (1991). “Strategic Flexibility for High Technology Manoeuvres: A Conceptual Framework”. In: *Journal of Management Studies* 28.1, pp. 69–89. doi: 10.1111/j.1467-6486.1991.tb00271.x.
- Fleischer, Lisa and Martin Skutella (2007). “Quickest Flows Over Time”. In: *SIAM Journal on Computing* 36.6, pp. 1600–1630. issn: 0097-5397. doi: 10.1137/S0097539703427215.
- Ford, Lester Randolph and Delbert Ray Fulkerson (1963). *Flows in networks*. Tech. rep. Santa Monica, California: The RAND Corporation, p. 152.
- Förderverein Freie Netzwerke e. V. (2018). *freifunk.net*. Available at: <https://wiki.freifunk.net/Kategorie:English>, (Accessed: 5 Juni 2018).
- Foundation for Intelligent Physical Agents (Dec. 2002a). *FIPA Abstract Architecture Specification*. Available at: <http://www.fipa.org/specs/fipa00001/SC00001L.pdf>, (Accessed: 27 June 2018).
- (2002b). *The Foundation for Intelligent Physical Agents: An Overview*. Available at: <http://www.fipa.org>, (Accessed: 1 July 2018).
- Fox, Dieter et al. (1999). “Collaborative Multi-Robot Localization”. In: *Mustererkennung 1999*. Ed. by Wolfgang Förstner et al. Springer, Berlin, Heidelberg, pp. 15–26. isbn: 978-3-540-66381-2. doi: 10.1007/978-3-642-60243-6_2.
- Fox, M and D Long (2003). “PDDL2. 1: An Extension to PDDL for Expressing Temporal Planning Domains.” In: *Journal of Artificial Intelligence Research (JAIR)*.
- Franks, Nigel R. and Ana B. Sendova-Franks (2000). “Queen transport during ant colony emigration: a group-level adaptive behavior”. In: *Behavioral Ecology* 11.3, pp. 315–318. issn: 1465-7279. doi: 10.1093/beheco/11.3.315.
- Free Software Foundation (2015). *GLPK (GNU Linear Programming Kit)*. Available at: <https://www.gnu.org/software/glpk>, (Accessed: 1 October 2015).
- Frisancho, A. Roberto (1993). *Human Adaptation and Accommodation*, p. 532. isbn: 0472095110.
- Gantner, Zeno, Matthias Westphal, and Stefan Wölfl (2008). “GQR-A fast reasoner for binary qualitative constraint calculi”. In: *Proceedings of the AAAI’08 Workshop on Spatial and Temporal Reasoning*. Vol. 8, pp. 24–29. isbn: 978-1-57735-379-9.
- Gao, Yang et al. (Aug. 2016). “Mission Operations and Autonomy”. In: *Contemporary Planetary Robotics*. Weinheim, Germany: Wiley-VCH Verlag GmbH & Co. KGaA, pp. 321–401. doi: 10.1002/9783527684977.ch6.
- Gartner (2018). *5 Trends Emerge in the Gartner Hype Cycle for Emerging Technologies, 2018*. Available at: <https://www.gartner.com/smarterwithgartner/5-trends-emerge-in-gartner-hype-cycle-for-emerging-technologies-2018/>, (Accessed: 20 October 2018).
- Genesereth, Michael R. (1998). *Knowledge Interchange Format*. Available at: <http://logic.stanford.edu/kif/dpans.html>, (Accessed: 15 Juli 2018).
- Ghallab, Malik, Dana Nau, and Paolo Traverso (2004). *Automated Planning: Theory and Practice*. Morgan Kaufmann Publishers Inc., p. 663. isbn: 1-55860-856-7.
- Gislén, Anna et al. (May 2003). “Superior Underwater Vision in a Human Population of Sea Gypsies”. In: *Current Biology* 13.10, pp. 833–836. issn: 09609822. doi: 10.1016/S0960-9822(03)00290-2.
- Golbreich, Christine, Evan K. Wallace, and Peter F. Patel-Schneider (2012). *OWL 2 Web Ontology Language New Features and Rationale*. Available at: <https://www.w3.org/TR/owl2-new-features>, (Accessed: 27 April 2018).

- Gu, Yunhong and Robert L. Grossman (2007). "UDT: UDP-based data transfer for high-speed wide area networks". In: *Computer Networks* 51, pp. 1777–1799. ISSN: 13891286. DOI: 10.1016/j.comnet.2006.11.009.
- Gumstix (2015). *Overo Fire Technical Specifications*. Available at: <http://media.gumstix.com/docs/gs3503f.pdf>, (Accessed: 1 August 2018).
- Gupta, Rakesh and MJ Kochenderfer (2004). "Common sense data acquisition for indoor mobile robots". In: *AAAI Proceedings of the 19th National Conference on Artificial Intelligence*. Vol. 94041. San Jose, California: AAAI Press, pp. 605–610.
- Hager, Mary C. and Gene S. Helfman (Nov. 1991). "Safety in numbers: shoal size choice by minnows under predatory threat". In: *Behavioral Ecology and Sociobiology* 29.4, pp. 271–276. ISSN: 0340-5443. DOI: 10.1007/BF00163984.
- Hartanto, Ronny (2009). "Fusing DL Reasoning with HTN Planning as a Deliberative Layer in Mobile Robotics". Dissertation. University of Osnabrück.
- Helmert, Malte (2006). "The fast downward planning system". In: *Journal of Artificial Intelligence Research* 26, pp. 191–246. ISSN: 10769757. DOI: 10.1613/jair.1705. arXiv: arXiv:1109.6051v1.
- Hernández, Carlos, Julita Bermejo-Alonso, and Ricardo Sanz (2018). "A self-adaptation framework based on functional knowledge for augmented autonomy in robots". In: *Integrated Computer-Aided Engineering* 25.2, pp. 157–172. ISSN: 18758835. DOI: 10.3233/ICA-180565.
- Hoffmann, Jörg and Bernd Nebel (2001). "The {FF} Planning System: Fast Plan Generation Through Heuristic Search". In: *Journal of Artificial Intelligence Research (JAIR)* 14, pp. 253–302.
- Horling, Bryan and Victor Lesser (2004). "A survey of multi-agent organizational paradigms". In: *Knowledge Engineering Review* 19.4, pp. 281–316. DOI: 10.1017/S0269888905000317.
- Horridge, Matthew and Sean Bechhofer (2011). "The OWL API: A Java API for OWL ontologies". In: *Semantic Web* 2.1, pp. 11–21. ISSN: 15700844. DOI: 10.3233/SW-2011-0025.
- Horrocks, Ian, Oliver Kutz, and Ulrike Sattler (2006). "The Even More Irresistible SROIQ." In: *10th International Conference on Principles of Knowledge Representation and Reasoning*. Lake District, UK: AAAI Press, pp. 57–67.
- Horrocks, Ian, Peter F. Patel-Schneider, et al. (2004). *SWRL: A Semantic Web Rule Language Combining OWL and RuleML*. Available at: <https://www.w3.org/Submission/SWRL>, (Accessed: 27 April 2018).
- Hübner, Jomi Fred, Jaime Simão Sichman, and Olivier Boissier (2004). "Using the MOISE+ for a cooperative framework of MAS reorganisation". In: *17th Brazilian Symposium on Artificial Intelligence (SBIA'04)*. Ed. by Ana L. C. Bazzan and Sofiane Labidi. Vol. 3171. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 506–515. ISBN: 978-3-540-23237-7. DOI: 10.1007/b100195.
- IBM (2005). *An architectural blueprint for autonomic computing*. Available at: <https://www-03.ibm.com/autonomic/pdfs/AC%20Blueprint%20White%20Paper%20V7.pdf>, (Accessed: 26 April 2018).
- IETF Zeroconf Working Group (2013). *Zero Configuration Networking (Zeroconf)*. Available at: <http://www.zeroconf.org>, (Accessed: 8 June 2018).
- Ilardo, Melissa and Rasmus Nielsen (2018). "Human adaptation to extreme environmental conditions". In: *Current Opinion in Genetics and Development* 53, pp. 77–82. ISSN: 18790380. DOI: 10.1016/j.gde.2018.07.003.
- International Space Exploration Coordination Group (2013). *The Global Exploration Roadmap*. Tech. rep. August, p. 50.

- ISO/IEC (1996). *ISO/OSI 7498-1*. Available at: [http://standards.iso.org/ittf/PubliclyAvailableStandards/s020269{}_ISO{}_IEC{}_7498-1{}_1994\(E\).zip](http://standards.iso.org/ittf/PubliclyAvailableStandards/s020269{}_ISO{}_IEC{}_7498-1{}_1994(E).zip), (Accessed: 1 August 2018).
- Joyeux, Sylvain and Felix Rehrmann (2012). *Rock Standards: RG7 Bundles*. Available at: <http://rock.opendfki.de/trac/wiki/WikiStart/Standards/RG7>, (Accessed: 1 October 2018).
- Joyeux, Sylvain, Jakob Schwendner, and Thomas M. Roehr (2014). "Modular Software for an Autonomous Space Rover". In: *12th International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS 2014)*. Montreal, Canada.
- Karim Talal, Bendaoud and Merzougui Rachid (2013). "Service Discovery – A Survey and Comparison". In: *International Journal of UbiComp* 4.3, pp. 23–39. ISSN: 09762213. DOI: 10.5121/ij.u.2013.4303. arXiv: 1308.2912.
- Kennington, Jeff L. (1978). "A Survey of Linear Cost Multicommodity Network Flows". In: *Operations Research* 26.2, pp. 209–236. DOI: 10.1287/opre.26.2.209.
- Kortman, Martin et al. (2015). "Building block-based "iBoss" approach: fully modular systems with standard interface to enhance future satellites". In: *66th International Astronautical Congress (IAC)*, pp. 1–11. ISBN: 9781510818934.
- Krötzsch, Markus, Frantisek Simancik, and Ian Horrocks (Jan. 2012). "A Description Logic Primer". In: *Computing Research Repository (CoRR)* abs/1201.4.June, pp. 1–17. arXiv: 1201.4089.
- Kurokawa, Haruhisa et al. (2007). "Self-reconfigurable modular robot M-TRAN: distributed control and communication". In: *Proceedings of the 1st International Conference on Robot Communication and Coordination*. RoboComm '07. Piscataway, NJ, USA: IEEE Press, 21:1–21:7. ISBN: 978-963-9799-08-0.
- Leskovec, Jure and Rok Sosič (July 2016). "SNAP". In: *ACM Transactions on Intelligent Systems and Technology* 8.1, pp. 1–20. ISSN: 21576904. DOI: 10.1145/2898361.
- Liskov, Barbara H. and Jeannette M. Wing (Nov. 1994). "A behavioral notion of subtyping". In: *ACM Transactions on Programming Languages and Systems* 16.6, pp. 1811–1841. ISSN: 01640925. DOI: 10.1145/197320.197383.
- Liu, Jinguo, Xin Zhang, and Guangbo Hao (2016). "Survey on research and development of reconfigurable modular robots". In: *Advances in Mechanical Engineering* 8.8, pp. 1–21. ISSN: 1687-8140. DOI: 10.1177/1687814016659597.
- Lougee-Heimer, Robin (2003). "The Common Optimization INterface for Operations Research". In: *IBM Journal of Research and Development* 47.1, pp. 57–66. DOI: 10.1147/rd.471.0057.
- LP Solve Community (2018). *CPLEX LP file format*. Available at: <http://lpsolve.sourceforge.net/5.0/CPLEX-format.htm>, (Accessed: 25 March 2018).
- Lüssing, Linus (2013). *batman-adv scalability: Layer 2 Mesh Networks - Myths and Risks*. Available at: <https://wiki.freifunk.net/images/e/e1/Batman-adv-scalability.pdf>, (Accessed: 16 June 2018).
- Manola, Frank and Eric Miller (2004). *RDF Primer*. Available at: <https://www.w3.org/TR/rdf-primer>, (Accessed: 18 April 2018).
- MBARI (2018). *Monterey Bay Aquarium Research Institute: Vehicle Technology*. Available at: <https://www.mbari.org/technology/emerging-current-tools/vehicle-technology/>, (Accessed: 7 September 2018).
- Mears, Christopher, Maria Garcia De La Banda, and Mark Wallace (2014). "Lightweight dynamic symmetry breaking". In: *Constraints*. Vol. 19. 3, pp. 195–242. DOI: 10.1007/s10601-013-9154-2.

- Mears, Christopher, Maria Garcia de la Banda, et al. (2008). "A Novel Approach For Detecting Symmetries in CSP Models". In: *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*. Ed. by Laurent Perron and Michael A. Trick. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 158–172. ISBN: 978-3-540-68155-7. DOI: 10.1007/978-3-540-68155-7_14.
- Meiri, Itay (Nov. 1996). "Combining qualitative and quantitative constraints in temporal reasoning". In: *Artificial Intelligence* 87.1-2, pp. 343–385. ISSN: 00043702. DOI: 10.1016/0004-3702(95)00109-3.
- Meyna, Arno and Bernhard Pauli (2010). *Taschenbuch der Zuverlässigkeitstechnik*. 2nd ed. Munich, Germany: Carl Hanser Verlag. ISBN: 978-3-446-41966-7.
- Mlot, Nathan J., Craig A. Tovey, and David L. Hu (May 2011). "Fire ants self-assemble into waterproof rafts to survive floods". In: *Proceedings of the National Academy of Sciences* 108.19, pp. 7669–7673. ISSN: 0027-8424. DOI: 10.1073/pnas.1016658108.
- Mondada, Francesco et al. (2005). "The Cooperation of Swarm-Bots". In: *IEEE Robotics & Automation Magazine* 12.2, pp. 21–28. ISSN: 1070-9932. DOI: 10.1109/MRA.2005.1458313.
- Moore, Lorna G. (2000). "Comparative human ventilatory adaptation to high altitude". In: *Respiration Physiology* 121.2-3, pp. 257–276. ISSN: 00345687. DOI: 10.1016/S0034-5687(00)00133-X.
- Motik, Boris, Bernardo Cuenca Grau, et al. (2012). *OWL 2 Web Ontology Language Profiles*.
- Motik, Boris, Peter F. Patel-Schneider, et al. (2012). *Direct Model-Theoretic Semantics for OWL 2*. Available at: <https://www.w3.org/TR/2012/REC-owl2-direct-semantics-20121211>, (Accessed: 26 April 2018).
- Moubarak, Paul and Pinhas Ben-Tzvi (Dec. 2012). "Modular and reconfigurable mobile robotics". In: *Robotics and Autonomous Systems* 60.12, pp. 1648–1663. ISSN: 09218890. DOI: 10.1016/j.robot.2012.09.002.
- Murata, S, K Kakomura, and H Kurokawa (2008). "Toward a scalable modular robotic system - Navigation, docking, and integration of M-TRAN". In: *IEEE Robotics & Automation Magazine* 14.4, pp. 56–63.
- NASA (2018). *Data Rates>Returns - Mars Science Laboratory*. Available at: <https://mars.nasa.gov/msl/mission/communicationwithearth/data>, (Accessed: 1 August 2018).
- NASA Jet Propulsion Laboratory (2018). *Mars Exploration Rovers*. Available at: <http://www.jpl.nasa.gov/missions/mer/>, (Accessed: 27 March 2018).
- Newman, Mark (Mar. 2010). *Networks*. Oxford University Press. ISBN: 9780199206650. DOI: 10.1093/acprof:oso/9780199206650.001.0001.
- Object Management Group (2018). *Distributed Data Service (DDS)*. Available at: <http://www.omg.org/spec/DDS>, (Accessed: 5 June 2018).
- Ogden, J. C. and P. R. Ehrlich (1977). "The behavior of heterotypic resting schools of juvenile grunts (*Pomadasyidae*)". In: *Marine Biology* 42.3, pp. 273–280. ISSN: 0025-3162. DOI: 10.1007/BF00397751.
- Open Mesh Inc. (2018). *Open Mesh OM2P*. Available at: <https://www.openmesh.com/resource-downloads/OM-Series-Datasheet.pdf>, (Accessed: 25 February 2018).
- Open-mesh (2012). *B.A.T.M.A.N. (better approach to mobile ad-hoc networking)*. Available at: <https://www.open-mesh.org>, (Accessed: 25 February 2018).
- OpenWRT/LEDE Community (2018). *OpenWRT Homepage*. Available at: <https://openwrt.org>, (Accessed: 25 February 2018).
- Oxford University Press (2018). *Oxford Dictionary*. Available at: <https://en.oxforddictionaries.com>, (Accessed: 26 March 2018).

- Parker, Lynne E (2006). "Multiple Mobile Robot Systems". In: *Springer Handbook of Robotics*. Ed. by Bruno Siciliano and Oussama Khatib. Springer. Chap. 40, pp. 921–941.
- Parragh, Sophie N. and Jean-François Cordeau (July 2017). "Branch-and-price and adaptive large neighborhood search for the truck and trailer routing problem with time windows". In: *Computers & Operations Research* 83, pp. 28–44. issn: 03050548. doi: 10.1016/j.cor.2017.01.020.
- Patel, Rajendra, Mikael Hedelind, and Pablo Lozan-Villegas (2012). "Enabling robots in small-part assembly lines: The ROSETTA approach - an industrial perspective". In: *Proceedings of the 7th German Conference on Robotics (ROBOTIK)*, pp. 1–5.
- Pecora, Federico (2018). *Meta-CSP*. Available at: <http://federicopecora.github.io/meta-csp-framework>, (Accessed: 14 May 2018).
- Pednault, Edwin P. D. (1987). "Formulating multi-agent, dynamic-world problems in the classical planning framework". In: *Reasoning About Actions and Plans*. Elsevier, pp. 47–82. doi: 10.1016/B978-0-934613-30-9.50006-8.
- Pereira, Pedro Henrique Cipresso, João Lucas Leão Feitosa, and Beatrice Padovani Ferreira (2011). "Mixed-species schooling behavior and protective mimicry involving coral reef fish from the genus *Haemulon* (Haemulidae)". In: *Neotropical Ichthyology* 9.4, pp. 741–746. issn: 16796225. doi: 10.1590/S1679-62252011005000037.
- Persson, Jacob et al. (2010). "A Knowledge Integration Framework for Robotics". In: *Proceedings of the International Symposium on Robotics (ISR 2010)*, pp. 1–8. isbn: 9783800732739.
- Planthaber, Steffen, Martin Mallwitz, and Elsa Andrea Kirchner (2018). "Immersive Robot Control in Virtual Reality to Command Robots in Space Missions". In: *Journal of Software Engineering and Applications* 11.07, pp. 341–347. issn: 1945-3116. doi: 10.4236/jsea.2018.117021.
- Poettering, Lennart, Trent Lloyd, and Sebastien Estienne (2012). *Avahi*. Available at: <http://www.avahi.org>, (Accessed: 7 June 2018).
- Poslad, Stefan, Phil Buckle, and Rob Hadingham (2000). "The FIPA-OS agent platform: Open Source for Open Standards". In: *Proceedings of the 5th International Conference on Practical Application of Intelligent Agents and Multi-Agent Technology (PAAM)*. Manchester, UK, pp. 355–368.
- Prestes, Edson et al. (2013). "Towards a core ontology for robotics and automation". In: *Robotics and Autonomous Systems* 61.11, pp. 1193–1204. issn: 0921-8890. doi: 10.1016/j.robot.2013.04.005.
- PROJ Contributors and Open Source Geospatial Foundation (2018). *PROJ coordinate transformation software library*. Available at: <https://proj4.org/operations/projections/merc.html>, (Accessed: 22 August 2018).
- Prud'hommeaus, Eric and Andy Seaborne (2008). *SPARQL Query Language for RDF*. Available at: <https://www.w3.org/TR/rdf-sparql-query>, (Accessed: 20 April 2018).
- Pulliam, H. R. and T. Caraco (1984). "Living in groups: is there an optimal group size?" In: *Behavioural ecology: an evolutionary approach*, pp. 122–147. isbn: 0865427313.
- Putten, Bart-Jan van et al. (2009). "OperA and Brahms : A Symphony? Integrating Organizational and Emergent Views on Agent-Based Modeling". In: *Agent-Oriented Software Engineering IX*. Ed. by Michael Luck and Jorge J. Gomez-Sanz. Springer Berlin Heidelberg, pp. 257–271. doi: 10.1007/978-3-642-01338-6_19.
- Putz, Peter and A. Elfving (1991). *Control Techniques 2*. Tech. rep. CT2/CDR/DO/BL: Dornier GmbH.
- Quartz (2017). *John Deere has learnt there's no shortcuts to autonomous vehicles*. Available at: <https://qz.com/1042343/after-trying-to-build-self-driving-tractors-for->

- more - than - 20 - years - john - deere - has - learned - a - hard - truth - about - autonomy/, (Accessed: 2 November 2018).
- Quigley, Morgan et al. (2009). "ROS: an open-source Robot Operating System". In: *IEEE International Conference on Robotics and Automation (ICRA) Workshop on Open Source Software*. Kobe, Japan.
- Rahwan, Talal, TP Tomasz P. Michalak, et al. (2011). "Constrained Coalition Formation." In: *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*, pp. 719–725.
- Rahwan, Talal, Sarvapali Dyanand Ramchurn, et al. (Jan. 2009). "An Anytime Algorithm for Optimal Coalition Structure Generation". In: *Journal of Artificial Intelligence Research* 34, pp. 521–567. arXiv: 1401.3466.
- Rausand, Marvin and Arnljot Høyland (2009). *System reliability theory: models and statistical methods*. Ed. by David J. Balding et al. 2nd ed. Hoboken, New Jersey, USA: John Wiley & Sons, Inc.
- Ricart, Glenn and Ashok K. Agrawala (1981). "An optimal algorithm for mutual exclusion in computer networks". In: *Communications of the ACM* 24.1, pp. 9–17. ISSN: 00010782. DOI: 10.1145/358527.358537.
- Richter, Silvia and Matthias Westphal (Sept. 2010). "The LAMA planner: guiding cost-based anytime planning with landmarks". In: *Journal of Artificial Intelligence Research* 39.1, pp. 127–177. ISSN: 1076-9757.
- Rina Detcher (2003). *Constraint Processing*. Vol. 33. Morgan Kaufmann Publishers Inc., p. 480. ISBN: 978-1-55860-890-0.
- Roberts, Patrick and Brian A. Stewart (Aug. 2018). "Defining the 'generalist specialist' niche for Pleistocene Homo sapiens". In: *Nature Human Behaviour* 2.8, pp. 542–550. ISSN: 2397-3374. DOI: 10.1038/s41562-018-0394-4.
- Rockel, S et al. (2013). "An Ontology-based Multi-level Robot Architecture for Learning from Experiences". In: *AAAI Spring Symposium: Designing Intelligent Robots: reintegrating AI 2*, pp. 52–57. DOI: 10.1.1.403.4926.
- Roehr, Thomas M. (2010). *FIPA ACL Library*. Available at: <https://github.com/rock-multiagent/fipa{ }acl>, (Accessed: 8 June 2018).
- (2013a). *FIPA Services Library*. Available at: <http://github.com/rock-multiagent/fipa{ }services>, (Accessed: 8 June 2018).
- (2013b). *FIPA Services Rock Component*. Available at: <http://github.com/rock-multiagent/orogen-fipa{ }services>, (Accessed: 8 June 2018).
- (2018). "Constraint-based Mission Planning for a Reconfigurable Multi-Robot System". In: *Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial* 21.62, pp. 25–39. DOI: 10.4114/intartif.vol21iss62pp25-39.
- (2019). *Owlapi Library*. Available at: <https://github.com/rock-knowledge-reasoning/knowledge-reasoning-owlapi>, (Accessed: 10 February 2019).
- Roehr, Thomas M., Florian Cordes, and Frank Kirchner (Jan. 2014). "Reconfigurable Integrated Multirobot Exploration System (RIMRES): Heterogeneous Modular Reconfigurable Robots for Space Exploration". In: *Journal of Field Robotics* 31.1, pp. 3–34. ISSN: 15564959. DOI: 10.1002/rob.21477.
- Roehr, Thomas M., Florian Cordes, Frank Kirchner, and Ingo Ahrns (June 2010). "Cooperative Docking Procedures for a Lunar Mission". In: *Joint Conference of ISR 2010 (41st International Symposium on Robotics) and ROBOTIK 2010 (6th German Conference on Robotics)*. Munich, Germany, pp. 564–571.

- Roehr, Thomas M. and Ronny Hartanto (2014). "Towards safe autonomy in space exploration using reconfigurable multi-robot systems". In: *12th International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS 2014)*. ESA.
- (2015). "Using an organization-model for PDDL-based planning with a reconfigurable multi-robot system". In: *2015 Space Robotics Symposium*. Glasgow.
- Roehr, Thomas M. and Satia Herfert (2016). "A FIPA-Based Communication Infrastructure for a Reconfigurable Multi-robot System". In: *Robot 2015: Second Iberian Robotics Conference: Advances in Robotics, Volume 1*. Ed. by Luís Paulo Reis et al. Cham: Springer International Publishing, pp. 665–676. ISBN: 978-3-319-27146-0. DOI: 10.1007/978-3-319-27146-0_51.
- Roehr, Thomas M. and Frank Kirchner (2016). "Spatio-Temporal Planning for a Reconfigurable Multi-Robot System". In: *4th Workshop on Planning and Robotics (PlanRob)*. Ed. by Alberto Finzi and Erez Karpas. London, pp. 135–146.
- Roehr, Thomas M. and Darko Makreshanski (2010). *Service Discovery*. Available at: <http://github.com/rock-core/tools-service{ }discovery>, (Accessed: 8 June 2018).
- Roehr, Thomas M., Petre Munteanu, et al. (2019). *Graph Analysis Library*. Available at: <https://github.com/rock-core/tools-graph{ }analysis>, (Accessed: 10 February 2019).
- Roehr, Thomas M. and Pierre Willenbrock (2018). "Binary Packaging for the Robot Construction Kit". In: *4th International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS 2018)*. Madrid, ES: ESA.
- Roehr, Thomas M., Pierre Willenbrock, and Sylvain Joyeux (2018). *Apaka: automated packaging for autoproj*. Available at: <https://github.com/rock-core/apaka>, (Accessed: 18 June 2018).
- Rothenbächer, Ann-Kathrin, Michael Drexler, and Stefan Irnich (Oct. 2018). "Branch-and-Price-and-Cut for the Truck-and-Trailer Routing Problem with Time Windows". In: *Transportation Science* 52.5, pp. 1174–1190. ISSN: 0041-1655. DOI: 10.1287/trsc.2017.0765.
- Russell, Stuart and Peter Norvig, eds. (2003). *Artificial Intelligence: A Modern Approach*. 3rd ed. Pearson Education. ISBN: 0137903952.
- Sandholm, Tuomas et al. (July 1999). "Coalition structure generation with worst case guarantees". In: *Artificial Intelligence* 111.1-2, pp. 209–238. ISSN: 00043702. DOI: 10.1016/S0004-3702(99)00036-3.
- Schagatay, E. (2014). "Human breath-hold diving ability and the underlying physiology". In: *Human Evolution* 29.1-3, pp. 125–140. ISSN: 1824310X.
- Schervan, Thomas A. et al. (2017). "Design Strategy, Numerical Analysis and Testing of a Modular Satellite Structure". In: *Proceedings of the 68th International Astronautical Congress (IAC)*. September, pp. 25–29.
- Schmidt, Nicole and Arndt Lüder (2015). *AutomationML in A Nutshell*. Available at: <https://www.automationml.org/o:red/uploads/dateien/1447420977-AutomationML{ }20in{ }20a{ }20Nutshell{ }151104.pdf>, (Accessed: 20 April 2018).
- Schulte, Christian and Guido Tack (2012). "View-based propagator derivation". In: *Constraints* 18.1, pp. 75–107. ISSN: 1383-7133. DOI: 10.1007/s10601-012-9133-z. arXiv: arXiv:0908.2050v1.
- Schulte, Christian, Guido Tack, and Michael Z. Lagerkvist (2018). *Modeling and Programming with Gecode*. Available at: <http://www.gecode.org/doc-latest/MPG.pdf>, (Accessed: 13 March 2018).
- Schwalb, Eddie and Rina Dechter (June 1997). "Processing disjunctions in temporal constraint networks". In: *Artificial Intelligence* 93.1-2, pp. 29–61. ISSN: 00043702. DOI: 10.1016/S0004-3702(97)00009-X.

- Schwendner, Jakob, Sylvain Joyeux, et al. (2012). *Numeric Library*. Available at: <http://github.com/rock-core/base-numeric.git>, (Accessed: 1 July 2018).
- Schwendner, Jakob, Thomas M. Roehr, et al. (2014). "The Artemis Rover as an Example for Model Based Engineering in Space Robotics". In: *2014 IROS Workshop on Modelling, Estimation, Perception and Control of All Terrain Mobile Robots*. Hongkong, CN, pp. 179–185. ISBN: ISSN 0946-0098.
- Searle, John R., Ferenc Kiefer, and Manfred Bierwisch, eds. (1980). *Speech Act Theory and Pragmatics*. Dordrecht: Springer Netherlands. ISBN: 978-90-277-1045-1. DOI: 10.1007/978-94-009-8964-1.
- Seither, Daniel, Andre Konig, and Matthias Hollick (Oct. 2011). "Routing performance of Wireless Mesh Networks: A practical evaluation of BATMAN advanced". In: *2011 IEEE 36th Conference on Local Computer Networks*. IEEE, pp. 897–904. ISBN: 978-1-61284-928-7. DOI: 10.1109/LCN.2011.6115569.
- Shrot, Tammar, Yonatan Aumann, and Sarit Kraus (2010). "On Agent Types in Coalition Formation Problems". In: *Proceedings of 9th International Conference on Autonomous Agents and Multiagent System (AAMAS)*. AAMAS, pp. 757–764.
- Siek, Jeremy, Lie-Quan Lee, and Andrew Lumsdaine (2000). *Boost Graph Library (BGL)*. Available at: <http://www.boost.org/libs/graph>, (Accessed: 12 October 2018).
- Singh, Moirangthem Sailash and Viswanath Talasila (Dec. 2015). "A practical evaluation for routing performance of BATMAN-ADV and HWMN in a Wireless Mesh Network test-bed". In: *2015 International Conference on Smart Sensors and Systems (IC-SSS)*. IEEE, pp. 1–6. ISBN: 978-1-4673-9328-7. DOI: 10.1109/SMARTSENS.2015.7873617.
- So, Y and E Durfee (1993). "An organizational self-design model for organizational change". In: *Workshop on AI and Theories of Groups and Organizations: Conceptual and Empirical Research*, pp. 8–15.
- Soetens, P (2012). *RTT: Real-Time Toolkit*. Available at: <http://www.oroocos.org/rtt>, (Accessed: 13 April 2014).
- Sonsalla, Roland, Florian Cordes, Leif Christensen, Steffen Planthaber, et al. (2014). "Towards a Heterogeneous Modular Robotic Team in a Logistic Chain for Extraterrestrial Exploration". In: *Proceedings of the International Symposium on Artificial Intelligence, Robotics and Automation in Space*. Montreal, Canada: ESA.
- Sonsalla, Roland, Florian Cordes, Leif Christensen, Thomas M. Roehr, et al. (2017a). "Field Testing of a Cooperative Multi-Robot Sample Return Mission in Mars Analogue Environment". In: *14th Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA 2017)*. Leiden, NL.
- (2017b). "Field Testing of a Cooperative Multi-Robot Sample Return Mission in Mars Analogue Environment". In: *14th Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA 2017)*.
- Spenneberg, Dirk and Frank Kirchner (2007). "The Bio-Inspired {SCORPION} Robot: Design, Control & Lessons Learned". In: ed. by Houxiang Zhang. Vol. Climbing &. I-Tech Education and Publishing, Wien, Austria, pp. 197–218.
- Stanford Center for Biomedical Informatics Research (2015). *Protégé*. Available at: <http://protege.stanford.edu>, (Accessed: 1 October 2015).
- Stock, Sebastian (2016). "Hierarchische hybride Planung für mobile Roboter". Doctoral dissertation. University Osnabrück.
- Stock, Sebastian et al. (2015). "Hierarchical Hybrid Planning in a Mobile Service Robot". In: *KI 2015 – Advances in Artificial Intelligence*. Ed. by Steffen Hölldobler et al. Springer International Publishing, pp. 309–315. DOI: 10.1007/978-3-319-24489-1_28.

- Suh, John W., Samuel B. Homans, and Marc Yim (2002). "Telecubes: mechanical design of a module for self-reconfigurable robotics". In: *Proceedings of the 2002 IEEE International Conference on Robotics and Automation*. Vol. 4. Washington, DC, USA: IEEE, pp. 4095–4101. ISBN: 0-7803-7272-7. DOI: 10.1109/ROBOT.2002.1014385.
- Suzuki, Ichiro and Tadao Kasami (1985). "A distributed mutual exclusion algorithm". In: *ACM Transactions on Computer Systems* 3.4, pp. 344–349. ISSN: 07342071. DOI: 10.1145/6110.214406.
- Tenorth, Moritz and Michael Beetz (May 2013). "KnowRob: A knowledge processing infrastructure for cognition-enabled robots". In: *The International Journal of Robotics Research* 32.5, pp. 566–590. ISSN: 0278-3649. DOI: 10.1177/0278364913481635.
- Tilk, Christian et al. (Mar. 2018). "Branch-and-Price-and-Cut for the Active-Passive Vehicle-Routing Problem". In: *Transportation Science* 52.2, pp. 300–319. ISSN: 0041-1655. DOI: 10.1287/trsc.2016.0730.
- Toth, Paolo and Daniele Vigo, eds. (2014). *Vehicle Routing: Problems, Methods, and Applications, Second Edition*. 2nd ed. MOS-SIAM. ISBN: 1611973597. DOI: 10.1137/1.9781611973594.
- Truszkowski, Walt, Mike Hinchey, and James Rash (2004). "NASA's Swarm Missions: The Challenge of Building Autonomous Software". In: *IT Professional* 6.October, pp. 47–52.
- Tsarkov, D and I Horrocks (2006). "FaCT++ Description Logic Reasoner: System Description". In: *Automated reasoning*. Ed. by Ulrich Furbach and Natarajan Shankar. Vol. 4130. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 292–297. ISBN: 978-3-540-37187-8. DOI: 10.1007/11814771.
- Ueda, Suguru et al. (2011). "Concise characteristic function representations in coalitional games based on agent types". In: *IJCAI International Joint Conference on Artificial Intelligence*, pp. 393–399. ISBN: 9781577355120. DOI: 10.5591/978-1-57735-516-8/IJCAI11-074.
- Vassev, Emil et al. (2012). "Swarm Technology at NASA: Building Resilient Systems". In: *IT Professional* 14.2, pp. 36–42. DOI: 10.1109/MITP.2012.18.
- Vázquez-Salceda, Javier and Frank Dignum (2003). "Modelling Electronic Organizations". In: *Multi-Agent Systems and Applications III*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 584–593. ISBN: 3-540-40450-3. DOI: 10.1007/3-540-45023-8_56.
- Vilain, Marc, Henry Kautz, and Peter van Beek (2013). "Constraint Propagation Algorithms for Temporal Reasoning: A Revised Report". In: *Readings in Qualitative Reasoning About Physical Systems*, pp. 373–381. DOI: 10.1016/B978-1-4832-1447-4.50034-1.
- W3C OWL Working Group (2012). *OWL 2 Web Ontology Language*. Available at: <http://www.w3.org/TR/owl2-overview>, (Accessed: 11 April 2018).
- Washington, Richard et al. (1999). "Autonomous Rovers for Mars Exploration". In: *Aerospace Conference, 1999. Proceedings. 1999 IEEE* 1, pp. 237–251. DOI: 10.1109/AERO.1999.794236.
- Weiss, Gerhard, ed. (2009). *Multiagent Systems*. 2nd ed. Cambridge, Massachusetts: MIT Press, p. 867. ISBN: 978-0-262-01889-0.
- Wenzel, Wiebke, Florian Cordes, Alexander Dettmann, et al. (June 2011). "Evaluation of a dust-resistant docking mechanism for surface exploration robots". In: *2011 15th International Conference on Advanced Robotics (ICAR)*. Tallinn, Estonia: IEEE, pp. 495–500. ISBN: 978-1-4577-1159-6. DOI: 10.1109/ICAR.2011.6088561.
- Wenzel, Wiebke, Florian Cordes, and Frank Kirchner (Sept. 2015). "A robust electro-mechanical interface for cooperating heterogeneous multi-robot teams". In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Vol. 2015-Decem. i. IEEE, pp. 1732–1737. ISBN: 978-1-4799-9994-1. DOI: 10.1109/IROS.2015.7353601.
- Wikipedia (2018a). *Adaptive Immune System*. Available at: https://en.wikipedia.org/wiki/Adaptive_immune_system, (Accessed: 1 September 2018).

- Wikipedia (2018b). *Transformers*. Available at: <https://en.wikipedia.org/wiki/Transformers>, (Accessed: 4 July 2018).
- Wilcox, Brian H et al. (May 2007). “Athlete: A cargo handling and manipulation robot for the moon”. In: *Journal of Field Robotics* 24.5, pp. 421–434. ISSN: 15564959. DOI: 10.1002/rob.20193.
- Winikoff, Michael (2005). “Jack™ Intelligent Agents: An Industrial Strength Platform”. In: *Multi-Agent Programming*, pp. 175–193.
- Winter, Matthias et al. (2015). “ExoMars Rover Vehicle: detailed description of the GNC System”. In: *Proceedings of the 2015 ASTRA Conference*. 1, pp. 1–8.
- Wooldridge, Michael and Nicholas R. Jennings (Aug. 1999). “The cooperative problem-solving process”. In: *Journal of Logic and Computation* 9.4, pp. 563–592. ISSN: 0955-792X. DOI: 10.1093/logcom/9.4.563.
- Wurm, Kai M. et al. (May 2013). “Coordinating heterogeneous teams of robots using temporal symbolic planning”. In: *Autonomous Robots* 34.4, pp. 277–294. ISSN: 0929-5593. DOI: 10.1007/s10514-012-9320-1.
- Yim, Mark, D.G. Duff, and K.D. Roufas (Apr. 2000). “PolyBot: a modular reconfigurable robot”. In: *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*. Vol. 1. San Francisco, CA: IEEE, pp. 514–520. ISBN: 0-7803-5886-4. DOI: 10.1109/ROBOT.2000.844106.
- Yim, Mark, Kimon Roufas, et al. (2003). “Modular reconfigurable robots in space applications”. In: *Autonomous Robots* 14.2-3, pp. 225–237. ISSN: 09295593. DOI: 10.1023/A:1022287820808.
- Zhang, Tan, Wenjun Zhang, and Madan Gupta (2017). “Resilient Robots: Concept, Review, and Future Directions”. In: *Robotics* 6.4, p. 22. ISSN: 2218-6581. DOI: 10.3390/robotics6040022.
- Zhong, Christopher and Scott A. DeLoach (2011). “Runtime models for automatic reorganization of multi-robot systems”. In: *Proceedings of the 6th International Symposium on Software Engineering for Adaptive and Self-Managing Systems - SEAMS '11*. New York, New York, USA: ACM Press, pp. 20–29. ISBN: 9781450305754. DOI: 10.1145/1988008.1988012.
- Zolli, Andrew and Ann Marie Healy (2012). *resilience*. Headline Publishing Group.
- Zykov, Victor, Andrew Chan, and Hod Lipson (2007). “Molecubes: An Open-Source Modular Robotics Kit”. In: *2007 IEEE/RSJ Conference on Intelligent Robots and Systems (IROS) Self-Reconfigurable Robotics Workshop*. San Diego, CA, USA.