Virtual Movement from Natural Language Text

DOCTORAL THESIS

by

Himangshu Sarma

Submitted in partial fulfilment of the requirements for the
Degree of Doctor of Engineering (Dr. -Ing)
on August 2018

in the

Faculty 3: Mathematics/Computer Science
University of Bremen, Germany
Defense Date: 10 July 2019

Supervisors:
Prof. Rainer Malaka
Prof. Arun Baran Samaddar

Acknowledgement

I would like to express my sincere gratitude to my supervisors Prof. Rainer Malaka and Prof. Arun Baran Samaddar for their constant support, their trust, their valuable feedback, their encouragement and their innumerable advice. Next, I would like to thank Robert and Jan for their all valuable helps to complete my work. This thesis would not have been possible without their valuable feedback. Their guidance is one of the main factors that shaped up my PhD.

My sincere thanks goes to Frederic, Benjamin, Aneta, Katya, Dirk, Nina, Tanja, Roland, Ambreen, Marc, Dmitry, Irmgard, Insa, Gerald, Svenja and as well as all the members of Digital Media Lab, University of Bremen for their help and valuable advice during my stay in Digital Media Lab. I would also like to express my sincere gratitude to the European Commission in the framework of Erasmus Mundus cLINK project for financial support. I would also like to thank Dr. Ingrid Rügge from the International Graduate School for Dynamics in Logistics (IGS) for her help and motivation during my PhD work.

And last but not the least, I would like to thank to my parents, sisters, and Nava Da for their constant support and encouragement during this journey.

- Himangshu Sarma

List of Acronyms and Abbreviations

	0-9
2D	Two-Dimensional
3D	Three-Dimensional
	A
ALARM	A Logical Alarm Reduction Mechanism
AML	Avatar Markup Language
APML	Affective Presentation Markup Language
ASAP	Artificial Social Agent Platform
	B
BAML	Body Animation Markup Language
BEAT	Behavior Expression Animation Toolkit
BML	Behavior Markup Language
BN	Bayesian networks
	C
CLAWS	Constituent Likelihood Automatic Word-tagging System
CML	Character Markup Language
CPT	Conditional probability table
	D
DAG	Directed acyclic graph
DMML	Dialogue Manager Markup Language
	E
ECG	Embodied construction grammar
EML	Emotion Markup Language
EMOTE	Expressive MOTion Engine
	F
FAML	Facial Animation Markup Language
FCG	Fluid construction grammar

	G
GML	Gesture Markup Language
	H
H Anim	Humanoid Animation
HamNoSys	Hamburg Notation System for Sign Languages
HC	Human Computation
HumanML	Human Markup Language
	I
IP	Internet Protocol
	M
MPML	Multimodal Presentation Markup Language
MPML-VR	Multimodal Presentation Markup Language for Virtual Reality
MURML	Multimodal Utterance Representation Markup Language
	N
NLU	Natural Language Understanding
NP	Noun Phrase
	P
PEISC	Physical Exercise Instruction Sheet Corpus
POS	Part-of-Speech
PP	Preposition Phrase
	R
RGB	Red Green Blue
	S
SAM	Solid Agents in Motion
SamIam	Sensitivity analysis, modeling, inference and more
SIGML	Signing Gesture Markup Language
SML	Speech Markup Language
STEP	Scripting Technology for Embodied Persona
	V
VB	Verb
VHML	Virtual Human Markup Language
VP	Verb Phrase
VR	Virtual Reality
VRML	Virtual Reality Modeling Language
	X
XML	Xtended Mark-up Language
XSTEP	XML-based Markup Language for Embodied Agents

Abstract

It is a challenging task for machines to follow a textual instruction. Properly understanding and using the meaning of the textual instruction in some application areas, such as robotics, animation, etc. is very difficult for machines. The interpretation of textual instructions for the automatic generation of the corresponding motions (e.g. exercises) and the validation of these movements are difficult tasks.

To achieve our initial goal of having machines properly understand textual instructions and generate some motions accordingly, we recorded five different exercises in random order with the help of seven amateur performers using a Microsoft Kinect device. During the recording, we found that the same exercise was interpreted differently by each human performer even though they were given identical textual instructions. We performed a quality assessment study based on the derived data using a crowdsourcing approach. Later, we tested the inter-rater agreement for different types of visualization, and found the RGB-based visualization showed the best agreement among the annotators' animation with a virtual character standing in second position. In the next phase we worked with physical exercise instructions. Physical exercise is an everyday activity domain in which textual exercise descriptions are usually focused on body movements. Body movements are considered to be a common element across a broad range of activities that are of interest for robotic automation.

Our main goal is to develop a text-to-animation system which we can use in different application areas and which we can also use to develop multiple-purpose robots whose operations are based on textual instructions. This system could be also used in different text to scene and text to animation systems. To generate a text-based animation system for physical exercises the process requires the robot to have natural language understanding (NLU) including understanding non-declarative sentences. It also requires the extraction of semantic information from complex syntactic structures with a large number of potential interpretations. Despite a comparatively high density of semantic references to body movements, exercise instructions still contain large amounts of underspecified information. Detecting, and bridging and/or filling such underspecified elements is extremely challenging when relying on methods from NLU alone. However, humans can often add such implicit information with ease due to its embodied nature.

We present a process that contains the combination of a semantic parser and a Bayesian network. In the semantic parser, the system extracts all the information present in the instruction to generate the animation. The Bayesian network adds some brain to the system to extract the information that is implicit in the instruction. This information is very important for correctly generating the animation and is very easy for a human to extract but very difficult for machines. Using crowdsourcing, with the help of human brains, we updated the Bayesian network. The combination of the semantic parser and the Bayesian network explicates the information that is contained in textual movement instructions so that an animation execution of the motion sequences performed by a virtual humanoid character can be rendered. To generate the animation from the information we basically used two different types of Markup languages. Behaviour Markup Language is used for 2D animation. Humanoid Animation uses Virtual Reality Markup Language for 3D animation.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Pervasive automation, be it for general industrial automation or for robots performing everyday activities, includes a variety of topical research challenges [Tenorth et al., 2014]. Current state-of-the-art robotics faces substantial limitations. Most importantly, robots are primarily designed and programmed for specific tasks. This limits the scope of their application. However, multiple-purpose robots could perform different types of tasks in both households and industrial environments. These robots could perform in a flexible and robust manner. This kind of robots would be beneficial for the real world and have a good potential market. Although considerable developments have been undertaken in this area aiming at more efficient multiple-purpose re-appropriation and task execution instructions, these robots rely on learning by demonstration with a very precise and limited scope [Ju et al., 2014].

This has become the primary motivation for our work, which ultimately aims to develop multiple-purpose robots. These robots would be able to perform different tasks using various sources of text-based instructions with the main input limited to natural language text. The automatic extraction of movement plans from textual instructions for robot automation is a novel research area [Beetz et al., 2011]. This approach is promising with regards to scaling and automating new task ability acquisition, since a large number of instructions for a wide range of activities are readily available

online for various domains (such as, Instructables[1], WikiHow[2], etc.). These instructions could be of great use for robotic performances [Beetz et al., 2011]. Also, training of a robot is very crucial. Diverse methodologies are normally use for training a robot, such as, reinforcement learning, learning by demonstration, imitation learning and so on., which are generally very expensive [Bruce et al., 2017]. Computer animation can be introduced to train robots through imitation learning which will be cost-effective also. Using computer animation we can generate human like character and movements which can be use for robot training. As an intermediate step, in this project we only focus to present a prototype processing system for generating adequate digital avatar movement sequence executions that presents a prerequisite for mapping to a physical robotic body. Our research is based on a combination of a fully automated text-to-animation system with humans in the loop for improving results at an acceptable cost.

It is a difficult task to select a textual input for the animation system that provides a movement of human character and typical instruction that fits our motivation. After our first study, we found that the instructions for physical exercise match perfectly our search and motivation. Picking a domain as a physical exercise isn't just proper in our area, yet in addition helps the helth division, which also plays a key role in everyday activities of humans. Exercise is critical for everybody from kids to old individuals. It causes them to build up their physical and mental limit rapidly. Exercise can likewise anticipate diverse ailments, for example, heart disease, cardiovascular disease, cancer, high blood pressure, diabetes, and so on. Thusly, normal exercise is essential for physical wellness and great wellbeing. Building up a text-to-animation system for physical exercise instructions will be useful from various perspectives to the general public from robotics to exercise as well health where physical exercise plays a key role.

As per the American Heart Association, in the absence of regular physical activity, the body slowly loses its strength, stamina, and ability to function well. People who are physically active and maintain a healthy weight live about seven years longer than those who are not active and are obese[3].

[1] http://www.instructables.com/ ; Access Date: 1^{st} July 2018

[2] http://www.wikihow.com/Main-Page ; Access Date: 1st July 2018

[3] http://www.heart.org/HEARTORG/HealthyLiving/PhysicalActivity/FitnessBasics/Physical-activity-improves-quality-of-life_UCM_307977_Article.jsp#.WzScg599I8o ; Access Date: 1st July 2018

"Your lifetime risk for general dementia is literally cut in half if you participate in physical activity. Aerobic exercise seems to be the key. With Alzheimer's, the effect is even greater: Such exercise reduces your odds of getting the disease by more than 60 percent. How much exercise? Once again, a little goes a long way. The researchers showed you have to participate in some form of exercise just twice a week to get the benefit. Bump it up to a 20−minute walk each day, and you can cut your risk of having a stroke−one of the leading causes of mental disability in the elderly−by 57 percent." [Richter, 2015]

Natural language understanding (NLU) is an important element of such an automatic text to animation system. Despite notable recent progress like WordsEye [Coyne and Sproat, 2001], Carsim [Åkerberg et al., 2003], general purpose NLU is still a challenge for computer systems, e.g. contextualized disambiguation [Porzel, 2010]. Different kinds of semantic parsers are available that provide adequate semantic information contained in a textual form. While most of the work has been focused on understanding declarative utterances and interrogatives, less attention has been paid to the text type of instructions and the ensuing imperative forms. Therefore, typical semantic parsers usually give satisfying results for declarative sentence structures. We are aiming to understand imperative structures that contain a lot of implicit information, even under limiting pre-conditions such as a specific focus on exercising. As indicated above, implicit information is difficult to extract using a semantic parser since it often relates to contextual or experiential knowledge. Hence, humans are far better at filling in the gaps and understanding these type of textual instructions. Methods from human computation can thus be employed to fill gaps in automated NLU pipelines that cannot yet be filled by digital computation alone. Human input can be used to improve the underlying models apart from facilitating task-specific solutions. This can further assist in the development of more general approaches to scalable NLU.

In the first proof of concept case study we focus on creating a pipeline to extract action specifications from text-based instructions in the domain of exercise instructions. Instruction sheets that are typically employed for exercise instruction in physiotherapy, rehabilitation or prevention (PRP) form the primary input. Accordingly, we developed a text to animation system for physical exercises. The system automatically generates the corresponding animations from textual descriptions of the physical exercises. Exercise instructions were chosen as an initial domain since they contain a clear focus on body movements. The individual elements of such movements can function as building blocks

for a wide variety of activities. Due to the explicit focus, physical exercise instructions can be expected to include few elements that are not concerned with movement instructions and thus contain comparatively little noise in the form of semantic variability. This can ease the process of initially establishing the text to animation system. As an additional benefit, the resulting movement-sequence executions can be employed to provide an alternative and potentially more engaging and clearer exercise instruction modality for the application area of PRP [Smeddinck et al., 2014][Uzor and Baillie, 2013].

Work in this direction makes important contributions since natural language understanding has been the subject of a large number of research efforts. While domain-specific solutions exist the range of domains is limited, and the understanding within these domains is usually still limited to a predefined selection of constructions.

1.1 Motivation

As of now researchers has completed a great deal of work and structured distinctive semantic parsers or analyzer to discover the ground meaning of the sentences and words. Be that as it may, building a domain adapted semantic parser is still a challenging task [Wang et al., 2017]. In reality, till now natural language understanding did not work well enough at that time to power mainstream applications [Hirschberg and Manning, 2015]. In the MIT's Technology Review it is also said that "Machines that truly understand language would be incredibly useful. But we don't know how to build them." [KNIGHT, 2017] Thus, artificial agents such as software and machine find it difficult to understand language irrespective of the forms such as speech and text. If we focus on the text, understanding a textual instruction is very easy for the human brain as compared to a machine's abilities. Researchers working on text processing using deep language understanding have designed different semantic parsers to understand the semantic meaning of the text, but these systems are basically designed for declarative sentences instead of imperative sentences. An imperative sentence is very complex when compared to a declarative sentence as an imperative sentence contains implicit information that cannot be extracted using current systems. Therefore, I have focused on the NLU of imperative sentences.

The first study used text-based physical exercise instructions. Understanding these physical exercise instructions are the first target; after which we focused on generating a pipeline to extract specific action segments from text-based instructions with the help of human computation (HC) methods. As an input, users can provide text-based instructions of physical exercises. Using the instruction sheet provided, the system will automatically generate 2D/3D animations as the output. Using an HC approach, we determine which visualization serves best as an output of the video [Sarma et al., 2015]. Assessing the quality of human body movement performances is an important task in many application areas. This can range from sports applications to therapy, learning by demonstration in robotics and automated systems for generative animation. For example, the manual transformation of physical therapy exercises into computer-supported playful exercises in the form of so-called "exergames" or levels of exergames requires a lot of time and effort. This makes it impractical for therapists or smaller practices to transform their preferred sets of therapeutic exercises into exergames for their patients to use. Thus, my research set out to explore the potential of crowd-based quality of motion assessments as a necessary intermediate step in the extraction and validation of motions. The HC approach is promising in this regard since the task involves many aspects that are easy for humans, but difficult for machines [Krause and Smeddnick, 2011]. Also, we can use text-based instruction understanding in the field of robotics which is still held back from mainstream impact by a number of limitations regarding general purpose use; i.e., robots are designed only for specific (and limited) tasks. Therefore, multi-purpose robots could be beneficial for the real world. These robots can perform different tasks in households or industry, outside of strictly controlled manufacturing environments using different text-based instructions, where the main input is the textual instruction (e.g. from online instruction repositories).

For our case study, we took Physical exercise instructions sheets as our source. Usually, physical exercise instruction sheets are either in natural language text format or use a combination of text and image(s) as shown in Table 1.1. This study's focus is on only natural language text and specifically physical exercise instruction sheets. During our study, we found that if someone wants to do some exercise they typically do it in two ways: either by using instruction sheets or with the help of a physiotherapist. In both the cases, instruction sheets only provide rough guidance and no feedback, which leads to wrong or even harmful exercise performance and that also have a number of known obstacles [Uzor and Baillie, 2014]. As Physiotherapists are not constantly accessible adjacent, if

Table 1.1: Exercise Instruction Sheet

Starting Position	Begin this exercise by standing with your feet wider than shoulder width apart and your toes pointed forward.
Action	INHALE: Slowly lower your body and remember to bend slightly at your hips. Keep your weight back on your heels and your back as upright as possible. Make sure your knees don't cross the plane of your toes.
	EXHALE: Straighten legs and come up to the starting position to complete one rep.
Special Instructions	Do not go past 90 degrees at the bend in your knees because this causes additional stress on your joints. If you feel pain in your knees, just go down to where you don't feel pain and come back up. If you have difficulty performing this exercise you can also use a chair or wall to help with balance and the movement until you build sufficient strength.

we can design an animated version of the exercise from the instruction guidance sheets, people will confront less issues and have a superior affair while doing the exercise activities.

The application area of physical exercise instructions offers the benefit of a rather constrained focus on body movement. It was helpful with early explorations toward a more generally capable text to animation system aiming for broader applications in robotics. Also, motion-based digital applications and games for health have been getting more attention in recent years. This is because the games offer motivation, guidance, and objective analysis. Hence, they offer a promising approach to coping with challenges such as aging societies and the modern sedentary lifestyle [Smeddinck, 2016].

1.2 Research Question

The automatic extraction of movement plans from textual instructions is a novel research area. The main questions are, how can a robot understand textual instructions as humans do and embody them in some humanoid agent? Aiming to achieve this, we present a process and prototype system for

automating the process of generating adequate digital avatar movement-sequence executions from text-based instructions that are a prerequisite for mapping to a physical robotic body. The latter will be used as the application target of text to robotic enaction. This exploration is based on a combination of a fully automated text to animation system with humans in the loop for improving results at an acceptable cost.

1.3 Problem Definition and Approach

Currently, our text to animation system is limited to the physical exercise instructions domain only. All instructions are in natural language text only. the physical exercises will range from a very simple exercise, e.g., *"lift your arm"* to very complex exercises, such as *"Step 1 of King Pigeon Pose: Kneel upright, with your knees slightly narrower than hip width apart and your hips, shoulders, and head stacked directly above your knees. With your hands, press down against the back of your pelvis"*. Physical exercises are ranges from starting with one body part and evolving to a very complex exercise where different body parts are involved in a single exercise instruction. It is very difficult for a machine to generate the animation from a simple instruction even when only one body part is involved and a simple sentence structure is used. The primary reason for this is the need to understand the pragmatics or understand the proper meaning of that instruction. Therefore, we try to generate an animation for a simple exercise instruction with single and composite instructions for two poses. Also, we try to simplify the complex sentence structure of an exercise instruction before we proceed to generate the animation with the help of crowdsourcing.

To achieve the goal different areas were combined into one and a system consisting of four parts is proposed: (1) a semantic parser, (2) a Bayesian network, (3) an animation creation system, and (4) human computation for validation and feedback. In the first step, semantic information is extracted using embodied construction grammar [Chang et al., 2002]. The second step attempts a best guess explication of a complete semantic construct, filling in implicit location information using a Bayesian belief network [Friedman et al., 1997]. As a third step, the system generates an animation file using an appropriate XML-based markup language which is then employed to generate a variable number of best candidate animation videos as an output relating to the original textual exercise instructions.

As the final step, human computation serves to isolate best candidate renderings. The resulting human computation ratings can then be used to update the Bayesian network in order to improve the quality of future best candidate generation.

1.4 Thesis Outline

This thesis is organized as follows and is shown in Figure 1.1.

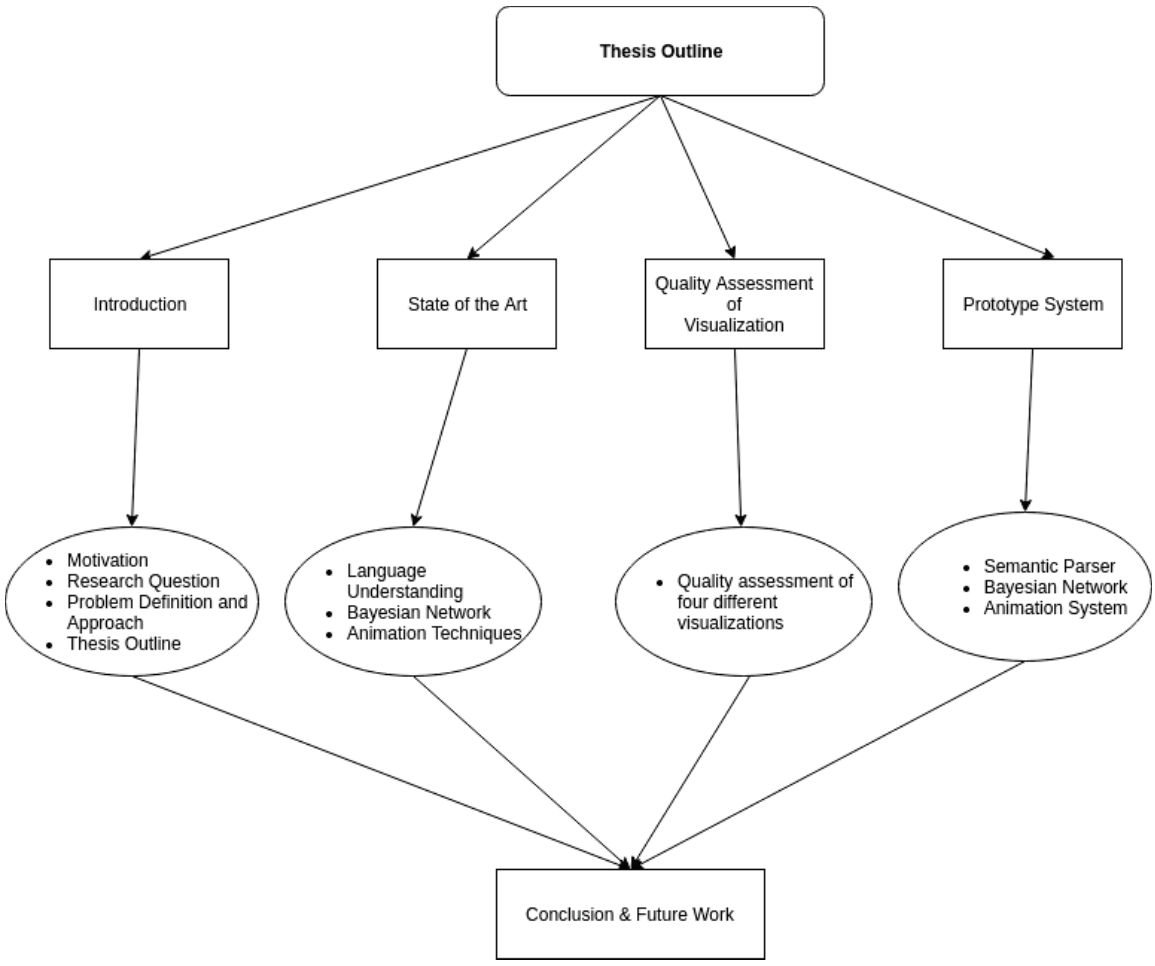


Figure 1.1: Thesis Outline

Chapter 2 reviews the literature and the current state of the art of the topic. In this chapter, we present a literature survey of the different approaches to text-to-animation and text-to-scene studied by other researchers. Also, we describe different semantic parsers and animation creation tools in this chapter.

Chapter 3 analyzes four different visualizations, i.e., RGB, Depth Image, Skeleton and Animated as candidates for our visualization.

Chapter 4 explains the framework of our text to animation system, which consists of three parts, i.e., a semantic parser, Bayesian network and animation system. We also provide the experimental results of all three parts of the text to animation system.

Chapter 5 concludes the thesis, it gives a summary of the results achieved as well as pointing out the assumptions and the limitations of our text to animation system. A list of issues is provided that have not yet been explored and also represents the future works we intend to pursue and which we can extend our architecture into a humanoid robot.

Chapter 2

State of the Art

Methods for Interpreting Language Using Textual Instructions

Different text to animation technique

Language is a simple and efficient medium to depict a scene or instruction [Coyne and Sproat, 2001]. However, learning or understanding a visual information scene much stronger than text is said to be 60,000 times faster than textual information visual information [Michard et al., 2017]. Therefore, animation from natural text will be useful, instead of traditional text based learning to depict a visual information. Although several reports of research projects use a combination of text and animation in a natural language, these all have a certain limit. Most of these projects are limited to the output of the image/scene instead of animation video. Below we describe some of the popular projects where a system creates some animation or scene/image based on natural language text.

1. ***CONFUCIUS:*** This is a multimodal storytelling text to animation system with some specific scene-based objects, e.g. a house, a field, etc. [Ma, 2006]. This system generates animations of virtual humans from single declarative sentences containing an action verb. This system fully relies on the action verb and the subject of the sentence.

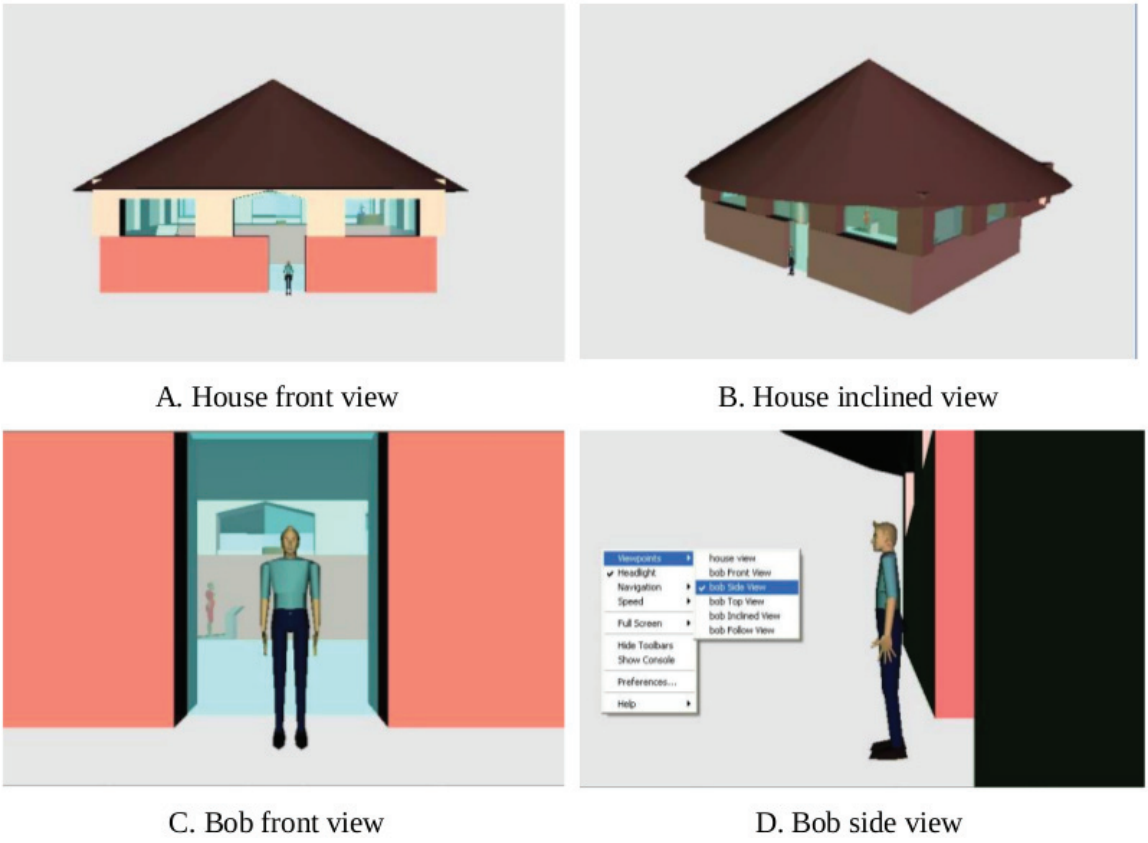


Figure 2.1: Different view of objects and character [Ma, 2006]

An example of a different view of a house and a character, Bob, is shown in Figure 2.1. The system is limited to a single sentence description with some limited action verbs and with a proper subjective sentence. Therefore, the system failed with multiple sentences. It also fails if the sentence is an imperative sentence. The system works for sentences such as "*Nancy ran across the field*".

2. ***Stanford's text to scene:*** As per the name, this is a text to 3D scene conversion system designed by a group of Stanford's researchers. The system generates a candidate scene and then the user can interact with it by a direct method or through textual commands. The author extracts the semantic information from text for a 3D scene (3D images) generation. This incorporates spatial knowledge and parses the natural text into a semantic representation [Chang et al., 2014]. This system addresses the inference of implicit spatial constraints by learning priors to spatial knowledge (e.g., typical positions of objects and common spatial relations). The user can interactively

Figure 2.2: Text to scene for query *"There is a table and a laptop"*

manipulate the generated scenes with textual commands, enabling it to refine and expand the learned priors [Chang et al., 2015]. This system totally depends upon a huge database of different scenes and cannot generate a new scene that is not contained in the database, which is limited to a room environment only as shown in Figure 2.2. So, technically this system is not

able to generate a run-time scene; it can only display the scene from the database at run-time. As you can see on the right side of Figure 2.2, when I searched for a query *"There is a table and a laptop"* a list of images were displayed in which every scene is inside a room environment. Also, on the left side, you can see how the scene has been saved in a hierarchical way in the database. This system is limited and thus is able to generate a room environment scene only. Therefore, if we search for some other query with a different environment, e.g., *"There is a boy in the field"*, the system fails to understand the search parameters and shows an error message.

3. ***CarSim:*** This is an automatic text to scene system for the purpose of supporting the analysis of road accidents[Åkerberg et al., 2003]. The CarSim system relies on a type of text where the scene has been described in detail and is limited to road accidents. Therefore, the researchers used some news articles about accidents to describe the scene. The system uses the descriptions to create 3D scenes with two different stages:

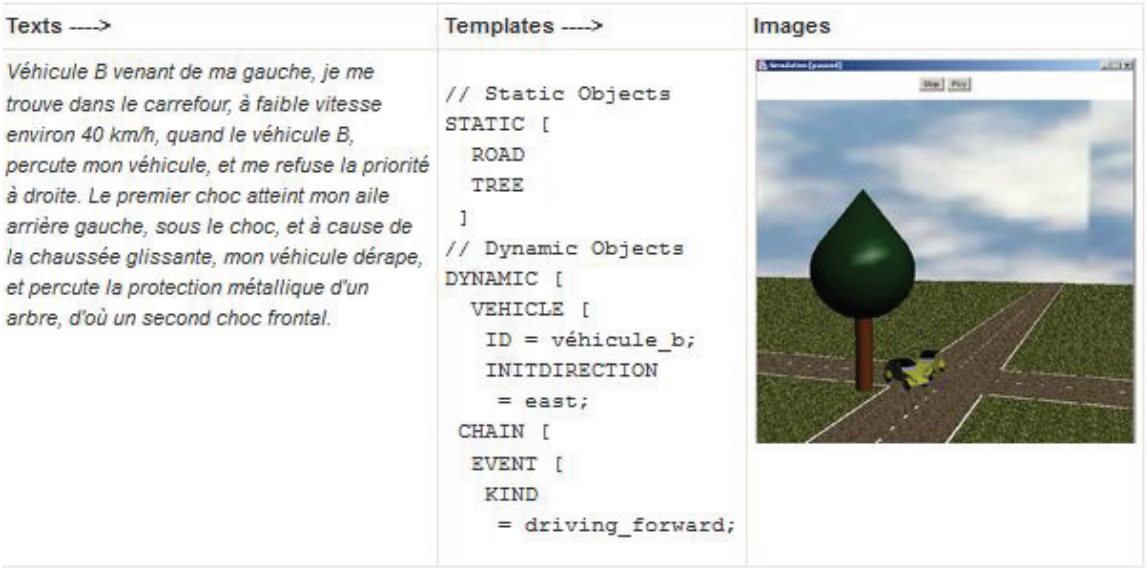


Figure 2.3: Text to scene using CarSim

- *Information Extraction:* This creates a tabular description of the scene.
- *Visual Simulator:* In this module the system creates the 3D animation.

An example of how the system generates a 3D scene is shown in Figure 2.3. The system is designed for two languages: French and English. In English, the authors used accident summaries

from the United States National Transportation Safety Board, which is a government accident research organization.

4. ***WordsEye:*** An automatic text to scene conversion system called *WordsEye* by AT&T Labs-Research [Coyne and Sproat, 2001] aims to create 3D scenes (3D images) from a textual description. It is based on a huge database of pre-designed 3D models and poses that depict entities and actions. It is limited to 3D images and is not able to create an animation video. AT&T Labs-Research has been working on this for the last decade and the system is not yet completely accurate. The project is still not complete. It displayed some noisy results for longer sentences. To describe this. I have tried to create a 3D scene for the sentence below as shown in Figure 2.4.

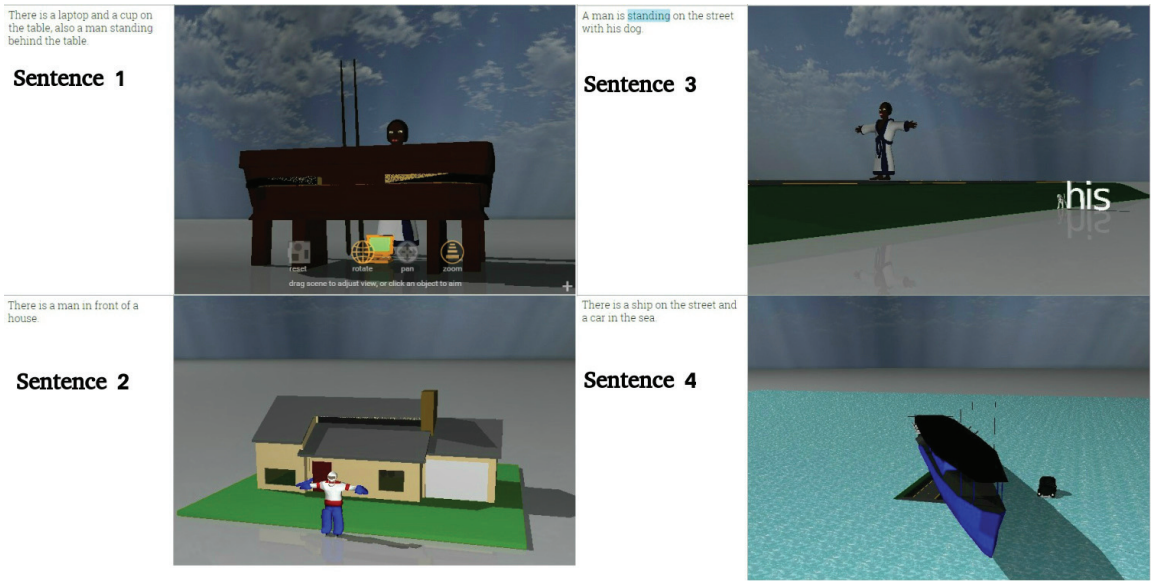


Figure 2.4: Sample results of WordsEye

- **Sentence 1** *There is a laptop and a cup on the table, also a man standing behind the table:* created a 3D scene for this sentence as shown in the top right of Figure 2.4. As you can see, in the image all the objects mentioned in the sentence are present in the scene but the laptop and the cup are not on the table.
- **Sentence 2** *There is a man in front of the house:* The 3D scene for this sentence is accurate, as shown in bottom right of Figure 2.4.
- **Sentence 3** *A man is standing on the street with his dog:* The result of this scene is very noisy as shown in the top right of Figure 2.4. In the scene, as you can see, the man

is standing on the street as per the description; but the system was confused about term *"his"*, or we can see the system failed to understand the linguistic description. Therefore, the system displayed *"his"* as a text with the dog separately. So, in this case the system totally failed.

- **Sentence 4** *There is a ship on the street and a car in the sea:* In this example I wanted to check how the system would create a scene that is impossible scene in real life. The system displayed the image perfectly without any feedback, even though it would be impossible to create the situation in real life.

In-spite of the novelty and advantages, as discussed, the aforesaid models have limitations in understanding of natural languages. Thus, understanding texts of a language is still a complex phenomena in the context of text-to-scene/animation. Instructions in the form of text require appropriate model to understand language [Beetz et al., 2011], which open-up a new perspective of language processing. Next section, describes various phases, models, and state-of-the-art of natural language understanding.

2.1 Language Understanding

Natural Language Understanding(NLU) is a method of natural language learning through the system or the software, regardless of the form. NLU still has various challenges such as struggle to understand language, context-awareness, reasoning, ambiguity, misspellings, common-sense as compare to humans [KNIGHT, 2017, Cambria and White, 2014]. NLU is an important and challenging part of the linguistic process. Understanding natural language is too hard that it falls under AI-complete [Yampolskiy, 2013]. Concentrating only on understanding texts also has a variety of issues such as the implicit and explicit significance of words/phrases/sentences, the structure of words/phrases/sentences [Norvig, 1987]. The extreme variability in the formation of languages also makes it difficult to understand textual languages [Linell, 2004].

In the field of artificial intelligence, teaching machines how to understand text is extremely important [Kadlec et al., 2016]. Therefore, the first part in developing a text-to-animation system is

language understanding. As the focus of the thesis is the text-to-animation from physical exercise, we need to create a repository of the instructions associated with physical exercise. During creation of the repository we faced difficulties to analyze the instructions because of the informal, imperative, and irregular structure of the instructions. Because of its syntax structure and ambiguity, short texts are crucial for machinery understanding [Hua et al., 2015]. Various approaches, including morphology, syntactic parser, semantic parser, pragmatics and so on, are often used to understand a text. These are highly helpful tool for extracting and computational understanding of natural language text.

Table 2.1: POS tagset of Penn Treebank

Sl. No.	**Tag Name**	**Description**	**Sl. No.**	**Tag Name**	**Description**
1	CC	Coordinating conjunction	19	PRP$	Possessive pronoun
2	CD	Cardinal number	20	RB	Adverb
3	DT	Determiner	21	RBR	Adverb, comparative
4	EX	Existential there	22	RBS	Adverb, superlative
5	FW	Foreign word	23	RP	Particle
6	IN	Preposition or subordinating conjunction	24	SYM	Symbol
7	JJ	Adjective	25	TO	to
8	JJR	Adjective, comparative	26	UH	Interjection
9	JJS	Adjective, superlative	27	VB	Verb, base form
10	LS	List item marker	28	VBD	Verb, past tense
11	MD	Modal	29	VBG	Verb, gerund or present participle
12	NN	Noun, singular or mass	30	VBN	Verb, past participle
13	NNS	Noun, plural	31	VBP	Verb, non-3rd person singular present
14	NNP	Proper noun, singular	32	VBZ	Verb, 3rd person singular present
15	NNPS	Proper noun, plural	33	WDT	Wh-determiner
16	PDT	Predeterminer	34	WP	Wh-pronoun

17	POS	Possessive ending	35	WP$	Possessive wh-pronoun
18	PRP	Personal pronoun	36	WRB	Wh-adverb

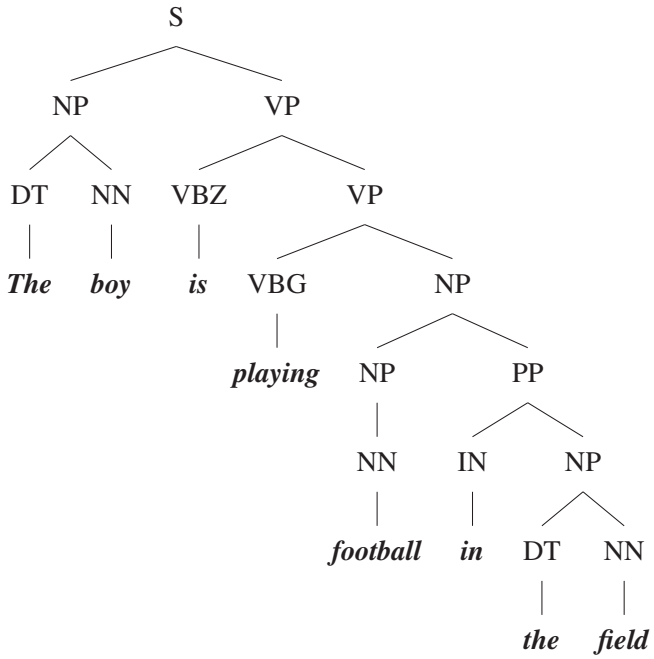


Figure 2.5: Syntactic structure of *'The boy is playing football in the field'*

- ***Syntactic Parser:*** This kind of parser is used to understand the syntactic structure of the sentence and tries to find out the dependency relation between different words of the sentence. There are some main parts of the syntactic parser, e.g., ***Syntax:*** that provides the rules used to combine the words into a sentence; ***Grammar(Context free Grammar):*** that eventually provides the rules of a specific language; and ***Parsing:***, a method for syntactic parsing of a sentence using the help of syntax and parsing.

Table 2.2: Derivation and rules used for syntactic structure of *'The boy is playing football in the field'*

Derivation	**Rules**
S	S –>NP VP
NP VP	NP –>DT NN
DT NN VP	DT –>The
The NN VP	NN –>boy
The boy VP	VP –>VBZ VP
The boy VBZ VP	VBZ –>is
The boy is VP	VP –>VBG NP
The boy is VBG NP	VBG –>playing
The boy is playing NP	NP –>NP PP
The boy is playing NP PP	NP –>NN
The boy is playing NN PP	NN –>football
The boy is playing football PP	PP –>IN NP
The boy is playing football IN NP	IN –>in
The boy is playing football in NP	NP –>DT NN
The boy is playing football in DT NN	DT –>the
The boy is playing football in the NN	NN –>field
The boy is playing football in the field	

There are a lot of syntactic parsers and part of speech (POS) taggers, that have been designed by a different research groups, e.g., link grammar, CLAWS POS tagger, Stanford parser, etc. POS tags are one of the most important elements in a syntactic parser to analyze the parsing of the sentence. The different parsers used different POS tagsets for their parser; for example, the CLAWS POS tagger uses CLAWS(Constituent Likelihood Automatic Word-tagging System) tagset[1] and the Stanford parser uses Penn Treebank POS tags. Penn Treebank is one of the famous POS tagsets are used in Penn Treebank, which is a collection of approximately 7,000,000 POS tagged words [Taylor et al., 2003]. Different POS tags used in Penn Treebank are shown in Table 2.1.

[1] `http://ucrel.lancs.ac.uk/claws1tags.html`

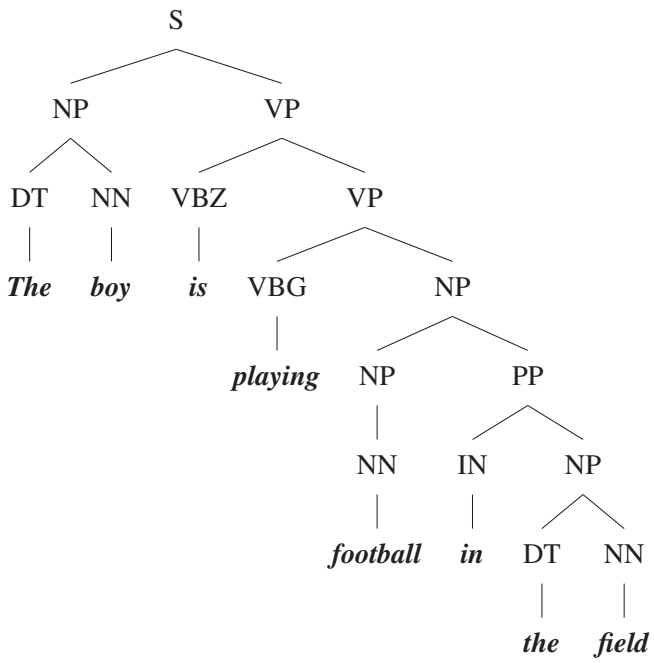


Figure 2.6: Syntactic structure of *'The boy is playing football in the field'*

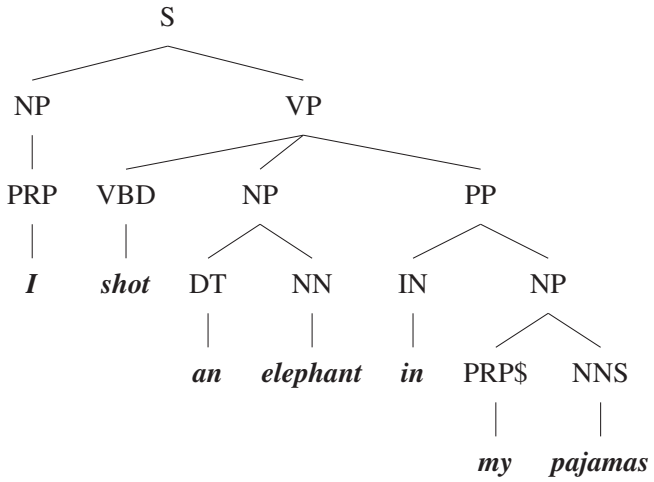


Figure 2.7: Syntactic structure of *'I shot an elephant in my pajamas'*

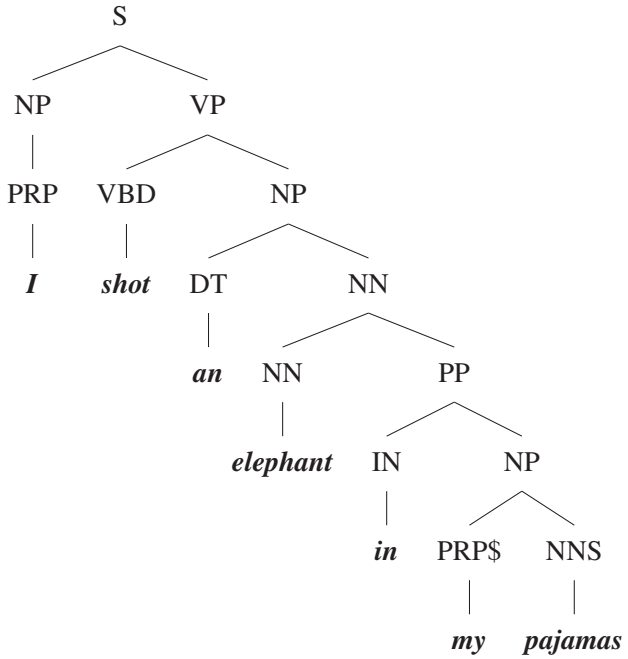


Figure 2.8: Syntactic structure of *'I shot an elephant in my pajamas'*

How a syntactic parser works for the sentence *'The boy is playing football in the field'* is shown in Figure 2.6 and how it derives, and what are the rules used for the derivation is shown in Table 2.2. As mentioned earlier, for the derivation of a sentence in syntactic parser context free grammar is used. But, when it comes to sentences like *'I shot an elephant in my pajamas'* the machines find it difficult to understand the sentence and its structure and meaning. This is because of the sentence's ambiguity, although the sentence is very easily understandable for a human brain. A human brain can easily understand that the subject—the person—shot an elephant when the subject was wearing pajamas, as shown in Figure 2.7. But machines can understand in multiple ways, e.g, that the subject or person shot an elephant and the elephant was wearing the subject's pajamas, as shown in Figure 2.8, which is impossible.

From the analysis, we can say that understanding natural language text is still a very complex problem for machines.

- ***Semantic Parser:*** This is used to extract the linguistic meaning of the sentence. To develop a semantic parser, we first need a syntactic parser to find the dependency relation between the words and the structure of the sentence.

Basically, we need a semantic parser which will able to extract the ground or basic or general meaning, i.e., the way a human understands an instruction. Without that understanding the basic meaning of an instruction cannot be generated for an animation. Different researchers/research groups have proposed different ways to extract semantic information from natural language text.

Researchers have also tried novel ways to find the semantics of the action verb; e.g., [Narayanan, 1997] proposed a new model which is able to use metaphoric projections of motion or action verbs to infer in real time the important features of abstract plans and events. This model is mainly an active representation of motion verbs that can be used to control real-time inferences in natural language understanding. The scientific community has produced a number of different syntactic and semantic parsers. The Stanford parser [De Marneffe et al., 2006] is a syntactic parser that is primarily employed for syntactic part of speech tagging. In general terms, the range of semantic parsers goes from shallow semantic parsers using support vector machines [Pradhan et al., 2004] via semantic role labelers [Björkelund et al., 2010] to fully symbolic semantic parsers, such as the *SEMAFOR* parser [Chen et al., 2010]. *SEMAFOR* is one of the popular semantic parsers, also known as a frame semantic probabilistic parser, using Frame Net [Baker et al., 1998] as a knowledge source. This parser tries to analyze the semantic information of the sentence using different Framenet frames, e.g. Person, Role, Performer, People_by_age etc. We also need to extract the textual instructions in different frames by which we can achieve our text to animation goal, which we are not able to extract using SEMAFOR or other typical semantic parsers.

In Figure 2.9 we have shown how a semantic parser works for the sentence "The boy is playing football in the field" and "The boy is playing football" for two different semantic parsers, SEMAFOR and a semantic roll label.

Beyond semantic PoS tagging, a semantic interpretation requires the application of a grammar, that is understandable for a machine as well as a robot. We can use this in our long-term goal, which is to design a multiple-purpose robot. Construction grammar [Goldberg, 1996] is a primarily linguistic approach to the semantic and pragmatic analysis of language. Construction grammars are mainly used in cognitive linguistics [Evans and Green, 2006], cognitive grammar [Langacker, 2008] and radical construction grammar [Croft, 2001]. Embodied construction grammar (ECG) [Bergen and Chang, 2005] and fluid construction grammar (FCG) [Steels,

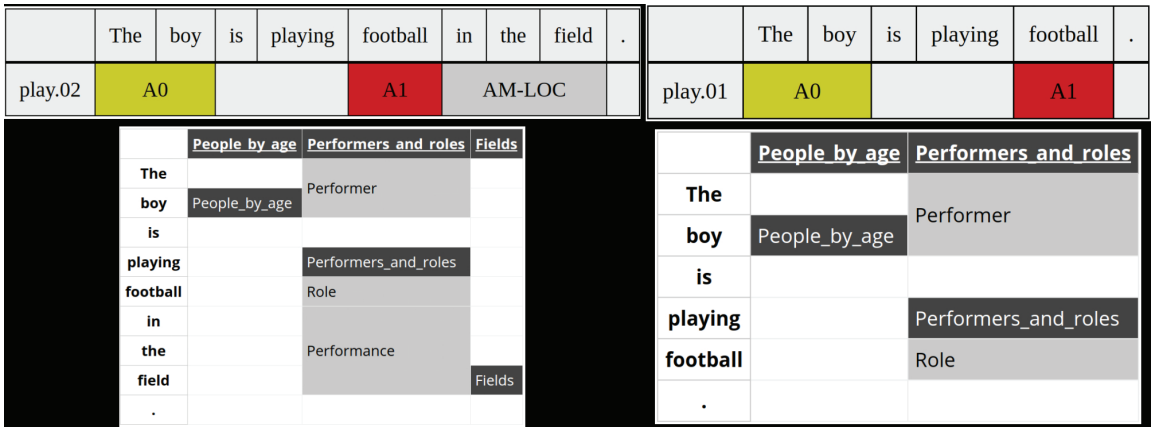


Figure 2.9: Semantic Parser: *Top:* Semantic Roll Labellers; *Bottom:* SEMAFOR

2011] are the two existing formal construction grammars that used for semantic parsing. ECG and FCG are both open source and are mainly designed for robotics work. We have found that we can build our grammar with either one of these two as part of achieving our long-term goal..

- ***Pragmatics:*** Pragmatics is mainly the study of how the human brain understands natural language. People don't say what they mean, or even normally. People typically use less words to express the scenario when some words are missing or implicit but humans usually understand the meaning in the context of the subject [Thomas, 2014]. Pragmatics is used to extract the information that is missing or implicit in natural language. This is a very complex process for machines although it is very easy for a human brain. As you can see from the semantic parser's results, it will be impossible to extract the implicit location from the sentence "The boy is playing football in the field," as shown in Figure 2.9. Therefore, to extract the implicit location, we need an intelligent system.

.

2.2 Bayesian Network

As discussed in the previous section, the understanding of natural languages is still not up to the mark for imperative sentence structure which contains implicit information. Therefore, the current state of

the art did not also obtain implicit information from the instructions for physical exercise.

If an automated language understanding system is tasked to understand the simple exercise instruction *lift your left arm*, it can be only extract the *Action* and *Bodypart*. A human can easily understand the implicit information as per the context of the sentence, such as from where and to where the arm should be lifted. Therefore, to understand the implicit information as per the context of the sentence or pragmatics, a Bayesian network is a useful tool. Bayesian network is used more and more to solve various issues, for instance, integrating multiple problems and systems components using information from various sources [Chen and Pollino, 2012]. Here, Bayesian network is used to extract the implicit or missing information which is not able to extract by the natural language understanding approach.

The conditional probability table (CPT) is the main part of the Bayesian network, which consists of some discrete random variables, in which the probability of a variable is calculated with the help of the other variables [Murphy, 2012]. CPT is also said to be the backbone of a Bayesian network.

Bayesian networks (BNs) are also known as a Bayes or belief networks. They follow a probabilistic directed acyclic graph (DAG) model [Jensen, 1996, Ruggeri et al., 2007] and typically consist of the following parts [Jensen, 2001]:

- a DAG;
- a CPT;
- variables and directed edges between variables;
- every variable has a mutually exclusive state.

BNs are frequently employed to model uncertainties developed from decision-making systems. One popular example of this is action planning and execution in robotics. In a BN, the CPT contains the critical transition probabilities, while the network structure models the domain elements to be considered. In order to form reliable and adequate CPTs, researchers usually apply a large dataset to analyze the domain in which they want to build the BN. However, it is not easy to compile sufficiently large datasets for every domain, and it can also be expensive.

Bayesian networks are used to find the knowledge about an uncertain domain or to find information about an unknown topic. Graphical models consist of two parts. They are:

- ***Node:*** In Bayesian network nodes are mainly representation of a variable such as bodypart, rain, wetgrass. There are two different types of variables, known as discrete and continuous [Aguilera et al., 2010].
- ***Edge:*** Edges are used for probabilistic dependencies between different Nodes or variables. Edges between nodes are added to indicate that one node has a direct impact on the other.

The uncertainty of a domain using Bayesian network can be found by using a probability between the various variables used in the graph. If *A* is a variable, *P(A)* is referred to as *A's* probability. Each variable has certain values, if *A* has *True* and *False* values of 0.4 and 0.6, it means that the probability of *A* is 40 percent True and 60 percent False. As mentioned, the primary means of prediction in Bayesian network is conditional probability. Based on the probability of variable *A* given by another Variable *B*, conditions are computed, it is denoted as *P(A|B)*. When *A* has a value *Cloudy* and *B* has a value *Rainy* and if the probability of *Cloudy* being *True* and *Rainy* being *True* is 80 percent then it is denoted as *P(Cloudy = True | Rainy = True) = 80%*

The next main element is to design the conditional probably table for each variable with its respective dependent variable once the conception of a graphical model has been completed as shown in Table 2.3.

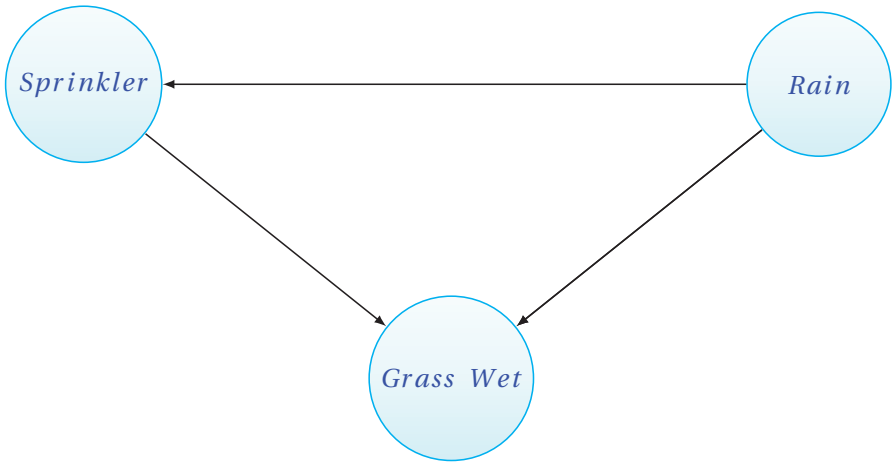


Table 2.3: Conditional Probability Table(CPT) for Grass Wet

Rain

T	***F***
0.2	0.8

Sprinkler

Rain	***T***	***F***
F	0.4	0.6
T	0.01	0.99

Grass Wet

Sprinkler	**Rain**	***T***	***F***
F	***F***	0.0	1.0
F	***T***	0.8	0.2
T	***F***	0.9	0.1
T	***T***	0.99	0.01

One of the popular examples of the Bayesian network is *GrassWet* with the help of either *Rain* or *Sprinkler* which is shown in above example.

A Bayesian network consists of three nodes: *Rain*, *Sprinkler* and *GrassWet*, where node *Rain* is independent, *Sprinkler* is dependent on *Rain*, and *GrassWet* is dependent upon both *Rain* and *Sprinkler*. The different probabilities for the network are shown in the Conditional Probability Table(CPT) of *Rain*, *Sprinkler* and *GrassWet*. As per the probability table, if there is any rain there is only a 1% chance the sprinkler will on. The probability of grass wet is 0% if there is not any rain and the sprinkler is off, but it increases to 80% if there is any rain, and 90% if the sprinkler is on; and if both the rain and sprinkler work together, the probability of grass wet rises to 99%.

There are many reports of researchers using a Bayesian network to design an intelligent system. The ALARM (A Logical Alarm Reduction Mechanism) network is one famous Bayesian network used to send text messages advising of a possible problem [Beinlich et al., 1989]. The Coma or Cancer network [Cooper, 1984] and the Asia network [Lauritzen and Spiegelhalter, 1988] are also some popular and well-known BNs. Usually, as we discussed earlier, in all these popular networks researchers developed a CPT to analyze a large dataset; but as we mention, refining the dataset using human computation or with the help crowdsourcing is a new approach.

There are lots of commercial and free academic tools available for designing a BN. Table 2.4 lists some popular free and commercial tools.

Table 2.4: List of tools to design Bayesian network

Sl.No	Tools	Commercial	Free
1	AgenaRisk [Birtles et al., 2014]	✔	
2	SamIam [Darwiche]		✔
3	GeNIe [Druzdzel, 1999a] [Druzdzel, 1999b]		✔
4	SMILE [Druzdzel, 1999b]		✔
5	Analytica [Chrisman et al., 2010]	✔	
6	HUGIN [Andersen et al., 1989]	✔	
7	MSBNx [Kadie et al., 2001]		✔
8	BayesiaLab [Conrady and Jouffe, 2013]	✔	
9	Javabayes [Cozman, 2001]		✔
10	Netica [Manual, 1997]	✔	
11	PULCINELLA [Saffiotti and Umkehrer, 1991]		✔

2.3 Animation Techniques

Animation has a considerable influence on modern society. Everyone from young children to older adults is influenced by animation for different purposes, e.g. education and entertainment. In this research, animation techniques also play a role in communication, as well as being a research topic in and of itself. If we look at the history of animation we find that that first animation was introduced in 3000 B.C.[2], shown in Figure 2.10. Text to animation systems are of interest in even more application areas, beyond robotics or games for health, such as animation used in movies.

Figure 2.10: SHAHR-E SUKHTEH: A bronze-age pottery bowl depicts goats leaping

In 1908, the first hand-drawn animation film *Fantasmagorie* was introduced, then six years later the first cartoon animation was introduced, named *Gertie the Dinosaur*.

[2]http://history-of-animation.webflow.io/ Access Date: 28th June 2018

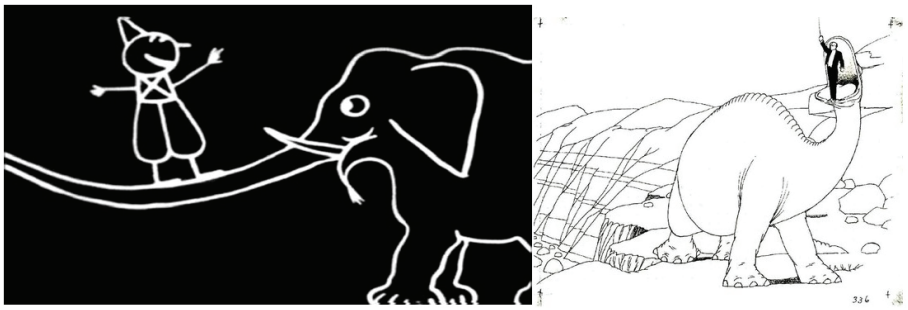


Figure 2.11: Left: Fantasmagorie; Right: Gertie the Dinosaur

But animation achieved to a different level when *Walt Disney and Ub Iwerks* introduced the character *Mickey Mouse* in *Steamboat Willie* in 1928.

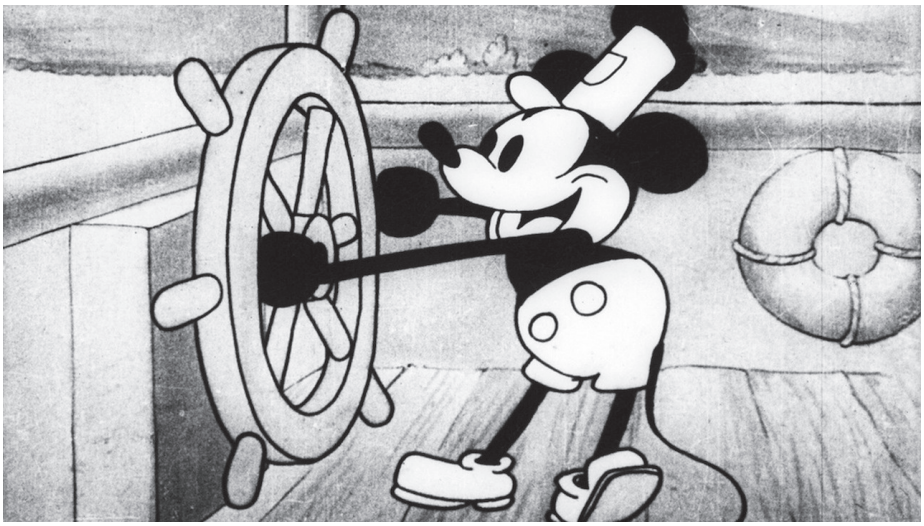


Figure 2.12: Mickey Mouse, in Steamboat Willie

In addition to the aforementioned areas of robotics and games for health, our proposed system can also contribute to the general field of animation by producing first draft motions (e.g. using screenplay or stage play scripts as the textual basis) that could then be further improved by animators. This is possible since game or film animation production uses very similar formalisms for defining and processing animations that can also be translated to the formats preferred by the robotics community. Nowadays, a number of mark-up languages exist for specifying animations. Due to the unique mix of important considerations, when considering both the immediate application area of text to generic virtual animation and the later application target of text to robotic action, we aim to find a formalism that would serve the needs of both application targets. Therefore, a list of markup languages which

can create animation is analyzed and shown in Table 2.5.

Table 2.5: List of MarkUp languages

Name	Speciality
Virtual Reality Modeling Language(VRML): It was also known as Virtual Reality Markup Language before 1995. It is mainly designed for the Internet, also called the world wide web. This was the first web-based 3D format. VRML is a kind of Markup language with the .wrl file extension. In 1997, it was certified by the International Organization for Standardization (ISO) [Brutzman, 1998].	Web, Humanoid animation
Humanoid animation(H Anim): It is designed for humanoid animation using VRML language. It models for 3D human figures. The file extension of this language is also .wrl or .x3d because it is a kind of VRML. During the design of H Anim, the focus was on achieving three goals: compatibility, flexibility and simplicity [Cobo and Bieri, 2002].	Humanoid animation
Multimodal Utterance Representation Markup Language(MURML): This is an animation system using either text or speech. This application is mainly designed for and limited to hand and head gestures [Kranstedt et al., 2002].	Humanoid animation
Behavior Markup Language(BML): It is a Markup language for controlling the verbal and nonverbal behavior of humanoid embodied conversational agents. The BML namespace is http://www.bml-initiative.org/bml/bml-1.0. I have shown an example of a BML block below [Kopp et al., 2006]. <bml xmlns="http://www.himangshu.org/bml/bml-1.0" id="bml1" > </bml>	Humanoid animation
Signing Gesture Markup Language(SIGML): This is a XML application that is mainly used to generate sign language gestures. It builds on HamNoSys [Elliott et al., 2004].	Sign Language
Character Markup Language(CML): This is a Markup language and animation scripting language designed for lifelike characters in different online applications or virtual reality worlds. CML is designed so that it can be easily understood by human animators and easily generated by software [Arafa and Mamdani, 2003].	Figure base animation not humanoid

Affective Presentation Markup Language(APML): It mainly specifies behavior at the meaning level. There are four main classes of communicative functions depending on the type of information, i.e., information about a speaker's belief, goal, affective state and meta-cognitive [De Carolis et al., 2004].	Facial animation
Expressive MOTion Engine(EMOTE): This is a 3D character animation, that applies effort and shape qualities to independently defined underlying movements and thereby generates more natural synthetic gestures. This system consists of four features, as described below [Chi et al., 2000]. (a) A given movement may have Effort and Shape parameters applied to it independently of its geometric definition. (b) A movement's Effort and Shape parameters may be varied along distinct numerical scales. (c) Different Effort and Shape parameters may be specified for different parts of the body involved in the same movement. (d) The Effort and Shape parameters may be phrased (coordinated) across a set of movements.	Arm movement
Hamburg Notation System for Sign Languages(HamNoSys): This is a phonetic transcription system, and it is also one of the well-known sign language applications for different hand shapes. HamNoSys was designed mainly in the context of the following goals, i.e., International use, iconicity, economy, integration, formal syntax and extensibility [Hanke, 2004].	Sign Language

Human Markup Language(HumanML): In 2002, OASIS introduced a Markup language known as HumanML. HumanML is designed for real time animated behaviors for 3D representations of humans [Brooks and Cagle, 2002]. The motivations for creating HumanML are: (a) clarifying human communication in digital information systems (b) bringing human perspectives, characteristics, qualities and values into information technology, and (c) identifying and focusing attention on uniquely human concerns.	Emotion
Avatar Markup Language(AML): Another Markup language, AML, is based on text to speech, facial animation and body animation. Researchers used low-level animation parameters and MPEG-4 to demonstrate AML. AML can also create 3D avatar animations easily and quickly. AML accepts four attributes and two sub-elements [Kshirsagar et al., 2002]. The attributes are: (a) face_id (b) body_id (c) root_path, and (d) name And the sub elements are: (a) <FA>: Facial Animation and/or (b) <BA>: Body Animation	Facial and body animation

Multi-modal Presentation Markup Languages(MPML): Another powerful Markup language is MPML, designed for verbal and non-verbal behavior of effective 2D cartoonstyle characters, presentation flow, and the integration of external objects. MPML also provides a visual editor which makes it easier for users to edit. The MPML system consists of four modules [Prendinger et al., 2004]. They are: (a) Script Loader: This module is used to load MPML scripts and for checking different syntactical errors. (b) Graph: In this module, the system generates the graphical representation from the MPML script. (c) Script Saver: This module is usually used to generate MPML script from graphs. (d) Converter: This module is used to convert MPML script to another language which is executable in a web browser.	Facial and Hand movements
Multimodal Presentation Markup Language for Virtual Reality(MPML-VR): MPML-VR is an extension of MPML that is presented in a 3D virtual space. The researchers also added various features to the MPML-VR agent [Okazaki et al., 2002]. They are: (a) Control of presentation space (b) VRML based (c) Animation (d) Locomotion (e) Speech and balloon (f) Emotional expression, and (g) Agent profile	Facial and Hand movements
Scripting Technology for Embodied Persona(STEP): This is a scripting language based on dynamic logic that is specially designed for embodied agents. STEP is implemented in such a way that it extends VRML/X3D with distributed logic programming [Huang et al., 2003].	Humanoid

Behavior Expression Animation Toolkit(BEAT): BEAT is mainly a text to embodied speech toolkit. It introduced a plug-in for nonverbal behavior generators and an XML-based processing pipeline [Cassell et al., 2004].	Behavior
XML- based Markup Language for Embodied Agents(XSTEP): XSTEP is a markup language for embodied agents. It is is an XML encoded STEP that is a 3D tool [Huang et al., 2003].	Humanoid
Functional Markup Language (FML): It is a mark-up language for texts which describes several phenomenons of discourse related to the content and the structure of information and interaction processes [Heylen et al., 2008].	Humanoid
Improv: This system was developed to create real-time behavior based animated actors. It also provides different tools to create actors in real time [Perlin and Goldberg, 1996]. Its improved architecture consists of two sub-parts as shown in Figure 2.16, they are: (a) Animation Engine: This engine provide different tools for generating realistic gestures and motions (b) Behavior Engine: This part is mainly designed for higher level capabilities and makes decisions about which animations to trigger.	Behavioural
Virtual Human Markup Language(VHML): Another popular markup language, VHML, is designed for Human-Computer Interaction and consists of different subsystems [Marriott, 2001]. They are: (a) DMML Dialogue Manager Markup Language, (b) FAML Facial Animation Markup Language, (c) BAML Body Animation Markup Language, (d) SML Speech Markup Language, (e) EML Emotion Markup Language, and (f) GML Gesture Markup Language.	Facial Animation, Body Animation

Solid Agents in Motion(SAM): It is a 3D programming language designed for parallel systems specification and animation. It is used mainly for exchanging messages from one computer port to another [Geiger et al., 1998]. The SAM cycle has two main parts. They are (a) Agent execution and (b) Communication	Exchanging messages

As we can see from the table and the analysis, we found that there are different types of mark-up languages are available to create different types of animation, e.g. humanoid, facial, arm, sign, and body animations. To generate animations for physical exercises, we mainly require humanoid animation. Choosing an animation that supports a text-to-animation system, who can generate the video from the textual physical exercise instructions is a challenging task. As mentioned, humanoid animation is required for creating videos from physical exercise instruction due to the whole body movement during exercise. As an output of the text-to-animation system the system will try to generate 2D and 3D videos. Hence, we focus on two different humanoid animation behavior markup language (BML), and H-Anim (a type of VRML) who can generate 2D and 3D animation respectively. We try to generate both kinds of animation from the system. In the future, we aim to extend the system to support any types of humanoid animation systems as well as import-export different types of humanoid characters in order to create animation without any markup language platform dependency.

On the basis of our study it is very difficult for machines to understand the pragmatics of natural language. Understanding the pragmatics of language or understanding of the natural language is an important part of any type of textual research.

Despite notable recent progress, such as WordsEye [Coyne and Sproat, 2001], Carsim [Åkerberg et al., 2003], general purpose NLU is still a challenge for computer systems, such as contextualized disambiguation [Porzel, 2010]. All of these systems are based on declarative sentences. They are only aimed at extracting syntactic and semantic data based on existing parsers. There are various types of semantic parsers available which provide adequate semantic information in sentences. While most

work has focused on the understanding of declarative and questionnaire expressions, less attention is paid to the text types of instructions and subsequent imperative forms. Typical semantic parsers therefore usually give satisfactory results for declarative sentence structures.

Chapter 3

Quality Assessment of Visualization

The development of a text-to-animation system for physical exercise instructions, a study of different types of exercise instructions must be carried out. It is very important for the same purpose to build a dataset or corpus of various types of exercise instruction sheets that are helpful for studying and can also be used as an input for the text-to-animation system. Thus, a Physical Exercise Instruction Sheet Corpus (PEISC) was build with approximately 1,000 physical exercise instructions drawn from publicly available databases. Some content (instructions) of the corpus are a combination of text and images as shown in Table 3.1,3.2,3.3,3.4,3.5. Also, the corpus is divided into different categories as per bodily movement of the exercise, such as standing, sitting, and so on.

After extracting the motions or poses, we must produce certain animations or videos from a natural language text with a particular form of visualization that can be comfortable for users or people to follow. Also, motivated by our use case of automatically generating movement patterns to be used in motion-based games as part of physiotherapy, rehabilitation, and prevention, we set out to explore the potential of crowd-based quality of motion assessments as a necessary intermediate step in the extraction and validation of motions. The videos can be displayed in different ways, depending on the scenario aspect. Hence, the selection of a special visualization out of various visuals, such as RGB, animation, skeleton and so on, without any reason, does not sound scientifically correct. It is therefore a good idea to study the various visualization processes using human beings in the loop who are the main

Table 3.1: Exercise Instruction Sheet: Squats

Starting Position	Begin this exercise by standing with your feet wider than shoulder width apart and your toes pointed forward.
Action	INHALE: Slowly lower your body and remember to bend slightly at your hips. Keep your weight back on your heels and your back as upright as possible. Make sure your knees don't cross the plane of your toes.
	EXHALE: Straighten legs and come up to the starting position to complete one rep.
Special Instructions	Do not go past 90 degrees at the bend in your knees because this causes additional stress on your joints. If you feel pain in your knees, just go down to where you don't feel pain and come back up. If you have difficulty performing this exercise you can also use a chair or wall to help with balance and the movement until you build sufficient strength.

Table 3.2: Exercise Instruction Sheet: Lateral Lunges

Starting Position	Begin by standing with your feet shoulder width apart, hands on hips.
Action	INHALE: Step out to the right and shift your body weight over your right leg, squatting to a 90 degree angle at the right knee. Try to sit down with your butt, keeping your back as upright as possible.
	EXHALE: Push off and bring your right leg back to center to complete one rep. Finish all reps on this side, and repeat on left side to complete one set.
Special Instructions	Keep your weight on your heels and make sure your knees don't go over the plane of your toes. Hold your arms out in front of you to help with balance.

Table 3.3: Exercise Instruction Sheet: Standing Iliotibial Band Stretch

Starting Position	Stand tall with you legs together, arms relaxed and back straight. Step your left leg behind your right leg, toes pointing forward and legs straight. Put your right hand on your hip and reach your left arm up in line with the shoulder.
Action	Breathe slowly and steadily as you push your hips toward the left and reach your left arm overhead and to the right. Hold the stretch for 10-30 seconds. Switch sides..
Special Instructions	For a deeper stretch, keep your feet farther apart, bend the knee of your front leg and keep the back knee straight. Keep your shoulders relaxed.

Table 3.4: Exercise Instruction Sheet: Forward Lunges

Starting Position	Stand with your feet about 6 inches apart from each other toes pointed forward.
Action	INHALE: Step forward with one leg and lower your body to 90 degrees at both knees. Don't step out too far. There should be 2 to 2.5 feet between your feet at this point. Keep your weight on your heels and don't allow your knees to cross the plane of your toes.
	EXHALE: Push up and back to the starting position to complete one rep. Repeat all reps on one leg, then switch to complete one set.
Special Instructions	Keep your back upright. The further you step, the more you work the glutes (buttocks) and hamstrings. The closer you step, the more you work the quadriceps muscles on the top of your thighs. Place your hand on a chair or wall or balance if necessary.

Table 3.5: Exercise Instruction Sheet: Reverse Lunges

Starting Position	Stand with legs hip-distance apart, toes pointed forward, back straight, hands on hips.
Action	INHALE: Take a big step backward with your right leg, keeping upper body as straight as possible. Bend both knees lowering your body and back knee toward the floor.
	EXHALE: Straighten legs and push off back leg to step forward to complete one rep.
Special Instructions	Take care to keep your back straight (without leaning) and front knee in line with ankle during lunge.

customers for the visualization. The human computation approach (HCI) is promising in this regard, since the task involves many aspects that are easy for humans, but difficult for machines [Krause and Smeddnick, 2011]. Since it is known that even human experts in quality of movement judgments share little inter-rater agreement, which is analysed for movements using a dataset of ten stroke patients and ten healthy age-matched volunteers [Pomeroy et al., 2003]. We set out to explore whether it is possible to achieve a level of inter-rater reliability that would even allow for quality of motion assessment and possibly project a later cross-validation, and to explore which type of a motion-visualization would support the best inter-rater reliability. Using HCI approach we can do an interaction between humans who are known as the users and computers. Here we try to determine user comfortability using a human computation approach in order to find specific visualization suits for users as a system output. We hypothesize that the video-based modality would yield the highest inter-annotator reliability. With this work, we contribute to human computation by exploring the novel area of quality of motion assessment where successful human computation could prove beneficial to a large number of application scenarios. We address the relevant related independent variable of motion visualization.

In order to find out the most suitable visualization for text based physical exercise instruction using human computation approach, we have to do some studies or surveys with the help of humans. To do the same, we aim to create a certain interface using which users can provide some feedback and from the same we try to analyze what the best visualization is and other aspects, such as difficulty level of exercises to perform and understanding, how people prefer to do exercise usually and so on. Therefore, for the survey, we choose five different exercises which are carried out on a standing basis without using any external equipment. These five exercises were recorded using Microsoft Kinect [1] which is a motion-sensing device build by Windows. These five exercises were recorded with the help of seven different human participants, three of which were men and the rest were women. All the participants were 15 to 35 years old, and M (Mean)=25 years and the SD (standard deviation)=5. Ten iterations of every exercise were recorded by each participant in random order. The order of the exercises for the participants is done using permutations as mentioned in Figure 3.1. During the recording of those exercises we only provided instruction sheets as shown in Table 3.1,3.2,3.3,3.4,3.5 and asked the participants to perform their interpretation of the exercises without any other input regarding how to perform them.

[1] `https://dev.windows.com/en-us/kinect`

We collected some demographic data about the participants before they started the exercises; the sample data sheet used to collect the demographic responses is shown in Figure A.1. After recording each exercise, we also asked some questions related to a specific exercise. The questionnaire is shown in Figure A.2. Before the recording session, some demographic questions were asked to the participants, they are:

- What is your age ?
- What is your gender ?
- Do you feel fit to perform some slightly to moderately intensive exercises today ?
- Do you have any problems with specific movements at this time ?
- How many times you perform physical exercises ?

Also, some questions related to the specific exercise performed were asked after performing every exercise. The pattern of the questions are as follows:

- Did you find these exercise instructions to be easy or difficult to understand ?
- Was it (physically) comfortable to perform this exercise ?
- Was it easy or difficult to perform this exercise ?
- Was it fun to perform this exercise ?
- Do you want to give any suggestions regarding the instructions for this exercise ?

A sample of the demographic questionnaire and questionnaire asked after each exercise is shown in Appendix A

Also, during the recording session, different observations of the participants' performance and technical details were noted as shown in Figure A.3. While analyzing the answers from the questionnaires, we found that the exercise instruction sheets were difficult for some participants to understand,

Table 3.6: Questions for the survey

Questions	**Answer Options**
Think of a typical week in the last few months. On how many days per week did you perform light exercise for at least 20 minutes? Light exercises such as walking, stretching,slow cycling, etc.	5-7 Days/Week 3-4 Days/Week 1-2 Days/Week Less than weekly Less than once a month
Think of a typical week in the last few months. On how many days per week did you exercise for at least 20 minutes? Medium to high intensity exercises such as running, gym training, tennis, skating, etc.	
Do you perform exercise initially through the following ? Please tick the exercise instruction modalities that you regularly rely on for your exercising:	Instruction Sheet, Video Instruction, With Instructor, Sports Club Gym (Fitness Club), Other
Have you ever performed regular exercises for physiotherapy or rehabilitation with a professional ?	Yes No
Do you judge the movement of others regularly while watching or in the role of providing feedback e.g. Sports, Dancing, Gym etc?	
Which exercise did you find the easiest to follow when reading the instruction sheet?	Squats, Lateral Lunges, Standing IT Band Stretch, Forward Lunges, Reverse Lunges
Which exercise did you find the most difficult to follow when reading the instruction sheet?	
Please rank how difficult you imagine each exercise to be when performing them following the instruction sheet., [Squats] [Lateral Lunges] [Standing It Band Stretch] [Forward Lunges] [Reverse Lunges]	Easy, Moderately easy, Average, Moderately difficult, Difficult
Please rank the visualization of exercise [Squats] : VR Skeleton Depth RGB [Lateral Lunges] : VR Skeleton Depth RGB [Standing It Band Stretch]: VR Skeleton Depth RGB [Forward Lunges]: VR Skeleton Depth RGB [Reverse Lunges]: VR Skeleton Depth RGB	***1- for Bad and 5- for Good*** Bad 1, 2, 3, 4, Good 5
Which visualization category did you find best overall and why ?	Virtual Reality(VR) Skeleton, Depth, RGB

P0	E1	E2	E5	E3	E4	**Right**
P1	E2	E3	E1	E4	E5	**Left**
P2	E3	E4	E2	E5	E1	**Right**
P3	E4	E5	E3	E1	E2	**Left**
P4	E5	E1	E4	E2	E3	**Right**
P5	E4	E3	E5	E2	E1	**Left**
P6	E5	E4	E1	E3	E2	**Right**

Figure 3.1: Permutation table for performing exercises; P0-P6 = Participant ID, E1-E5 = Exercise ID

but seemed easy for others. Furthermore, the same exercise was sometimes performed differently by the participants. Due to these findings, we can state that instruction sheets are not the optimal way to instruct people to perform exercises.

During the recording of the exercises using Kinect, we collected the data in the format of .xed for Kinect 1.8 and .xef for Kinect 2. For this study we used Kinect 1.8 and our data format is .xed. We recorded our exercises using Kinect 1.8 with .xed format. The Kinect data is a combination of color video, depth data, and skeleton tracking of 20 traceable human joints for Kinect 1.8 whereas 24 for Kinect 2. We generated four different kinds of visualizations from the raw data, shown in Figure 3.2. They are:

- ***RGB :*** This is a simple color video that can easily be recorded by anyone. People usually watch this kind of video in their daily lives. We extracted the video from the Kinect data.
- ***Depth :*** This visualization is likely same as RGB videos, but the best part is that one cannot see the real person, so it is best for ensuring the privacy of the data. We have developed an application using C# to extract the depth data from the raw data.
- ***Skeleton :*** This category is fully generated by joint tracking sensors of the human body. Using this visualization we can easily find which joint(s) is(are) moving and how much. To extract the skeleton information here we have developed an application in C# and extracted the skeleton videos as shown in Figure 3.2.
- ***Virtual Reality or Animated Character :*** This type of visualization is also generated from

the skeleton or joint tracking, and we can add a virtual character to the skeleton. This is one of the best options for ensuring the privacy of game-related visualizations. We mapped the data using a virtual character. To create this kind of video we have developed an application in Unity using C#.

Using the four types of visualizations listed above we developed a survey application to find the best visualization; it is shown in Figure 3.3.

We generated four different categories of videos from the Kinect data we collected (i.e., RGB, Depth, Skeleton, Virtual Reality, shown in Figure 3.2), and we developed a survey application using Unity[2], which is a game engine. The application has been designed, with the aim of crowdsourcing the assessments of the quality of exercise executions and to determine the best visualization modality for high inter-rater agreement.

During the survey, the participants watched 140 videos and rated those videos to determine the best visualization. For every exercise there are four different visualizations, and for every visualization there are seven videos. These videos are recorded and data is generated from Kinect for seven different performers. So, for every exercise, there are 7*4=28 videos, and for five different exercises 28*5=140 videos in total. During the survey, exercises were shown in random order to each of the participants. The visualizations of each exercise were also displayed in random order. In this survey, the participants were asked to click the worst performance video until all videos had been clicked. When a participant clicked a video, that specific video disappeared from the screen and the application kept a record of those selections. Later, the application sent all the records to the server when the survey was completed. Therefore, during the study, an active Internet connection was a definite requirement. We did the study online and shared the links through different social media platforms and websites, so participants could do the survey from any part of the world (a link to the application was spread via snowball sampling).

Following the quality assessment survey, we provided a questionnaire to gather comparative re-

[2] https://unity3d.com/

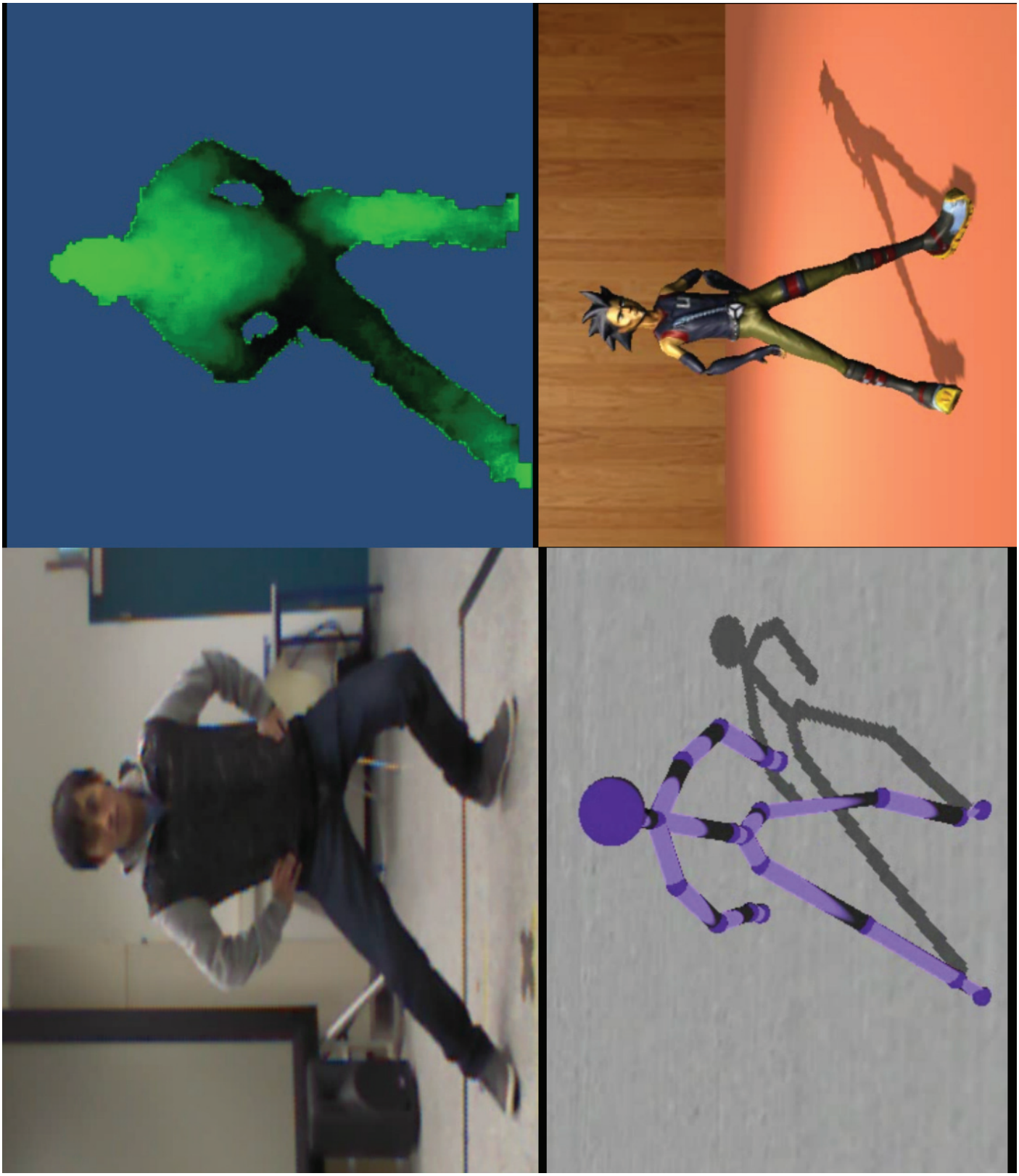


Figure 3.2: Four different visualization.

Figure 3.3: Screenshots of the survey application showing different visualizations

sponses regarding the best visualization type, the movement quality of different body parts during the performance of the exercises, and to acquire additional demographic data. The questionnaire was prepared using LimeSurvey[3], a tool for online surveys. Also, we recorded the IP address of each participant who did the survey. For our analysis we accepted only one record from each Internet Protocol(IP) address. The list of questions asked during the questionnaire is shown in Table 3.6. The answers to the questionnaire were automatically saved in the server. The participants' average age is 26.35 where standard deviation is 4.86 where 65% participants are male and 35% are female.

Results

A total of twenty-four people participated in the survey using the downloadable application. We analyzed the results of our survey along with the questionnaire. We used the kappa coefficient to analyze the data; this is also known as kappa statistics. Kappa statistics are used to measure inter-rater agreement for qualitative items [Carletta, 1996]. Two types of Kappa statistics are usually used to analyze the data, they are:

- *Cohen's kappa:* Cohen's kappa statistics is used to analyze the survey results that from the participants or judges who rated the video in a playable scenario. A sample example of how kappa statistics is calculated is shown in Figure 3.4. Data from two survey participants or judges is shown in Figure 3.4 for an exercise performed by six persons or performers, in which performer 1 to 6 are mentioned as P1,P2 ... P6. Kappa statistics is calculated using the following formula:

$$k = \frac{Pr(a) - Pr(e)}{1 - Pr(e)}$$

.

In order to calculate Pr(a), we have to calculate the total number of videos for which both participants are rating. In the Figure, we have shown for a total of ten videos. The total number of agreements between both participants is eight indicated in the Figure as **A**, which are primarily

[3] https://www.limesurvey.org/

the summation of the diagonal values,indicated in the figure as C6,D7,E8,F9,G10,H11. Pr(a) is calculated as:

$$Pr(a) = \frac{Total\ No}{Agreements\ between\ Judges}$$

.

$$Pr(a) = \frac{8}{10}$$

.

$$Pr(a) = 0.8$$

.

To calculate Pr(e) which is known as random agreement, we have to calculate agreement for each Judges separately for each performer. For calculating P1's probability of random agreement, the following formula is used:

$$P1 = \frac{Total\ Agreements\ for\ Judge\ 1}{Total\ No.} \times \frac{Total\ Agreements\ for\ Judge\ 2}{Total\ No.}$$

.

$$P1 = \frac{4}{10} \times \frac{4}{10}$$

.

$$P1 = 0.16$$

.

After, calculating the values for P1 to P6, the summation of the same(E28:E33) is known as Pr(e) as shown in the Figure 3.4. Thus, Pr(e) = 0.28. Hence, k is calculates as follows:

$$k = \frac{0.8 - 0.28}{1 - 0.28}$$

.

$$k = 0.72$$

.

	A	B	C	D	E	F	G	H	I	J
1										
2										
3					JUDGE 2					
4										
5			**P1**	**P2**	**P3**	**P4**	**P5**	**P6**	Total	
6		**P1**	4	0	0	0	0	0	4	SUM(C6:H7)
7		**P2**	0	0	1	0	0	0	1	SUM(C7:H7)
8		**P3**	0	0	2	0	0	0	2	SUM(C8:H7)
9		**P4**	0	0	0	0	0	0	0	SUM(C9:H7)
10		**P5**	0	0	0	0	2	1	3	SUM(C10:H7)
11	**JUDGE 1**	**P6**	0	0	0	0	0	0	0	SUM(C11:H7)
12		Total	4	0	3	0	2	1	**10**	SUM(C12:H7)
13										
14										
15					**A**	**8**	**SUM(C6,D7,E8,F9,G10,H11)**			
16										
17			**Pr(a)**	**0.8**	**F15/I12**					
18										
19					**Judge 1**				**Judge 2**	
20				P1	0.4	I6/I12		P1	0.4	C12/I12
21				P2	0.1	I7/I12		P2	0	D12/I12
22				P3	0.2	I8/I12		P3	0.3	E12/I12
23				P4	0	I9/I12		P4	0	F12/I12
24				P5	0.3	I10/I12		P5	0.2	G12/I12
25				P6	0	I11/I12		P6	0.1	H12/I12
26										
27										
28				P1	0.16	E20*I20				
29				P2	0	E21*I21				
30				P3	0.06	E22*I22				
31				P4	0	E23*I23				
32				P5	0.06	E24*I24				
33				P6	0	E25*I25				
34				**Pr(e)**	**0.28**	**SUM(E28:E33)**				
35										
36										
37										
38				**K**	**0.72**	**(D17-E34)/(1-E34)**				

Figure 3.4: Calculation of Kappa statistics from the feedback of two participants or judges

- *Fleiss' kappa:* Fleiss' kappa is an extension of Cohen's kappa. In Cohen's kappa the measure of agreement is limited to only two raters, whereas Fleiss' kappa can measure more than two

raters. The kappa statistic for Fleiss' kappa, k can be calculated using the following formula:

$$k = \frac{\overline{P} - \overline{P_e}}{1 - \overline{P_e}}$$

.

In kappa statistics, k varies from 0 to 1. If k = 1 it means there is a complete agreement between all raters, and if k = 0 then there is no any agreement between the raters. To calculate $\overline{P}$ and $\overline{P_e}$ the following equations are used:

$$P_j = \frac{1}{Nn} \sum_{i=1}^{N} n_{ij}$$

$$p_i = \frac{1}{n(n-1)} \sum_{j=1}^{k} n_{ij}(n_{ij} - 1)$$

$$\overline{P} = \frac{1}{N} \sum_{i=1}^{N} p_i$$

$$\overline{P_e} = \sum_{j=1}^{k} P_j^2$$

.

An example of how to calculate kappa statistics is shown below for the data mentioned in Table 3.7; also, p_i and P_j are shown in the same table. In the table "subject" N=4; "no. of raters" n=4, and "no. of categories" k=3.

Following are the values of p_i:

$$p_1 = \frac{1}{4(4-1)}(0^2 + 1^2 + 3^2 - 4) = 0.5$$

$$p_2 = \frac{1}{4(4-1)}(4^2 + 0^2 + 0^2 - 4) = 1$$

$$p_3 = \frac{1}{4(4-1)}(3^2 + 0^2 + 1^2 - 4) = 0.5$$

$$p_4 = \frac{1}{4(4-1)}(1^2 + 1^2 + 2^2 - 4) = 0.167$$

.

And, the values of P_j are:

$$P_1 = \frac{0+4+3+1}{4*4} = 0.5$$

$$P_2 = \frac{1+0+0+1}{4*4} = 0.125$$

$$P_2 = \frac{3+0+1+2}{4*4} = 0.375$$

.

So,

$$\overline{P} = \frac{1}{4}(0.5 + 1 + 0.5 + .167) = 0.542$$

and

$$\overline{P_e} = 0.5^2 + 0.125^2 + 0.375^2 = 0.406$$

Table 3.7: Example: Kappa statistics

	1	**2**	**3**	p_i
1	0	1	3	0.5
2	4	0	0	1
3	3	0	1	0.5
4	1	1	2	0.167
Total	8	2	6	
P_j	0.5	0.125	0.375	

Table 3.8: Kappa value interpretation

k	**Result**
0-0.1	Poor
0.11-0.3	Good
0.31-0.6	Very good
0.61-0.9	Better
0.91-1	Best

.

Therefore,

$$k = \frac{0.542 - 0.406}{1 - 0.406} = 0.229$$

.

The interpretation of our accuracy of kappa statistics is mentioned in Table 3.8.

Table 3.9: Best performer per exercise and the inter-rater agreement on the positioning

Exercise	**Performer**	**Kappa**
Squats	Performer 1	0.51
Lateral Lungs	Performer 2	0.59
Standing IT	Performer 6	0.53
Forward Lungs	Performer 1	0.74
Reverse Lunges	Performer 6	0.57

With the help of Kappa statistics, we calculated the best performer of all five exercises (shown in Table 3.9) and the best visualization type, displayed in Figure 3.5. Also, we display the result of the

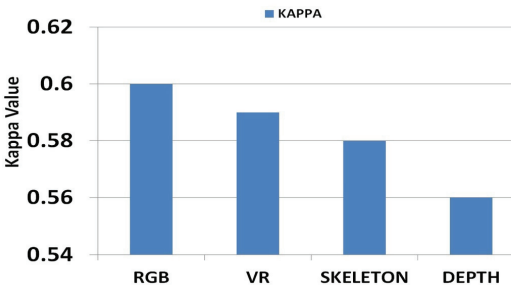


Figure 3.5: Agreement for different visualizations based on Kappa statistics

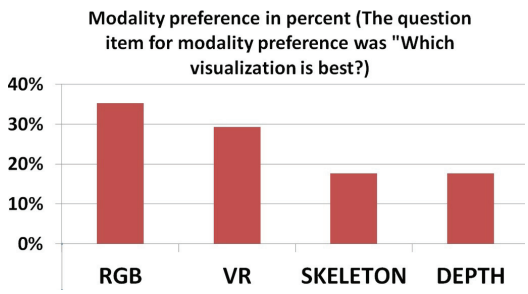


Figure 3.6: Agreement for different visualizations based on questionnaire

best visualization in Figure 3.6 based on the data that we got from crowdsourcing for the question, "Which visualization category did you find best overall and why?" In both the cases, we found that RGB was the best visualization and that depth or infrared was the worst, and virtual character animation was the second method of visualization for both cases. Also, using Kappa statistics we found which exercise is the best, as shown in Table 3.10. From the result we found that *"Lateral Lungs"* is the best exercise for performing.

Table 3.10: Best exercise using Kappa statistics

	Squats	**Lateral Lungs**	**Standing IT**	**Forward Lungs**	**Reverse Lunges**
Performer 1	0.51	0.54	0.40	0.74	0.47
Performer 2	0.40	0.59	0.45	0.45	0.38
Performer 3	0.47	0.50	0.50	0.52	0.53
Performer 4	0.46	0.50	0.42	0.49	0.51
Performer 5	0.47	0.50	0.50	0.36	0.53
Performer 6	0.44	0.48	0.53	0.51	0.57
Average	**0.46**	**0.52**	**0.47**	**0.51**	**0.50**

We also made some interesting observations: only 6% of the respondents rely on professional instruction to do their daily exercise, and more than 70% of the respondents never did their exercise with a professional. Therefore, we can say that the majority of people rely on instructions which also we found from our analysis, where showed that almost 60% of people rely on instructions (either textual or video instructions).

We did a case study in this Chapter to find out which visualization is best of four different visual-

izations, those are RGB(Red Green Blue, Normal video), Animation, Skeleton, and Depth or Infrared. We conducted a study of physical exercise videos and and created the ones using Kinect and developed an application for the survey and tried to find out which one is best with the human calculation approach. Our findings indicate that RGB remains best where animation with certain virtual character remains second and low-ranking visualizations are infrared or depth. The study was carried out on two counts, in both cases we found the same results with one data that we evaluated based on the survey application and the second on the questionnaire that we asked after the survey. In this Chapter, we were also looking to find out different questions concerning the instructions for physical exercise which we reported in the Chapter.

Chapter 4

Prototype System

We aim to design a text-to-animation system based on physical exercises where text-based exercise instructions are in the format of an imperative sentence. The first step in the design of a text-to-animation system is therefore to understand imperative sentences. We aim to understand imperative structures that contain many implicit information even under restrictive conditions, such as a particular focus on exercise. As mentioned, it is nearly impossible to extract and understand pragmatic or implicit information by a semantic parser because it often has to do with contextual or experiential knowledge. Human beings fill the gaps and understand these kind of textual instructions much better than machines or robots [Thomas, 2014, KNIGHT, 2017, Cambria and White, 2014]. No software is known from our study about which any type of pragmatics can be extracted from the phrases of text. An important tool is the understanding of pragmatics in a textual instruction. As people easily understand, an artificial intelligence and human calculation combination can therefore be used to address gaps in automated NLU programs that can not be filled alone in digital computers. If human input is used to support task-specific solutions and improve the underlying models, more general approaches to scalable NLUs can be developed.

To achieve our goal of a text to animation system that understands NLU we propose a system that consists of three parts: (1) a semantic parser, (2) a Bayesian network, and (3) an animation creation system as shown in Figure 4.1. In the first step, semantic information is extracted using embodied

construction grammar [Chang et al., 2002]. The second step attempts a best-guess explication of a complete semantic construct, filling in implicit location information using a Bayesian belief network [Friedman et al., 1997]. The third step is the system generating an animation file using an appropriate XML-based movement markup language, which is then employed to generate a variable number of best candidate animation videos as an output related to the original textual exercise instructions.

Work in this direction makes important contributions since natural language understanding has been a subject of a large number of research efforts. While domain-specific solutions exist, the range of domains is limited; and the understanding within these domains is usually still limited to a predefined selection of constructions.

- *Step 1*: In this step, we extract the basic semantic information to generate an animation from textual instructions. In its current state, the system extracts three different types of information, namely:
  - *Actions* - different types of actions, such as *lift*, *tilt*, etc.;
  - *Body parts* - that are prominently involved in the exercise, e.g. *shoulders*, *legs*, etc.;
  - *Location* - where body parts are to be moved to (focusing primarily on destination locations).
- *Step 2*: If any elements from the three types of information above are missing or underspecified, the system will refer to the Bayesian network to attempt to extract implicit semantic information.
- *Step 3*: After extracting all the required information from *Step 1 and 2*, the system automatically generates an animation file using behavior markup language and generates a video based on the animation execution.

As we mentioned above, understanding language is an important part of the development of an animation system that takes textual exercise instruction sheets as input. The implicit information contained in these exercise instructions is the most difficult part to extract automatically. Using existing systems, it is not possible to extract the implicit information from the exercise instructions. This is

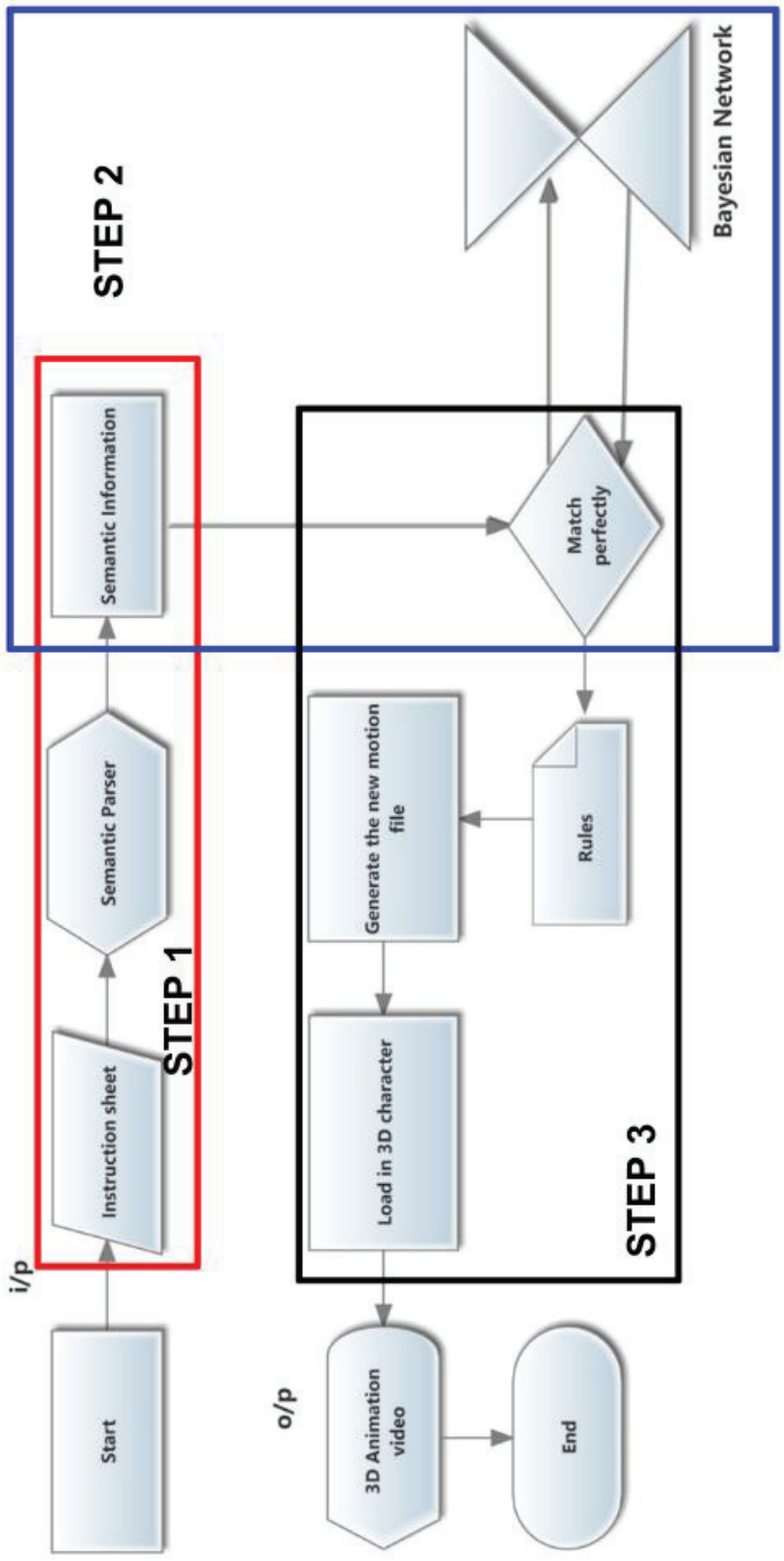


Figure 4.1: Pipeline for Virtual Movement from Textual Instructions

because semantic parsers can only extract information that is present in the sentence. The parsers do not have any cognitive knowledge to use to extract the implicit information. Therefore, we augmented a natural language understanding system in order to extract the required implicit information from the given exercise instruction sheets. Currently, our system is limited to single sentence exercise instructions only, such as *lift your left arm*.

If information on the action, body part, or location is missing the system moves to the next step. A Bayesian network sets the extracted information as evidence in the model and extrapolates the implicit information that is missing from the semantic analysis. In our physical exercise instruction sheet corpus, we found that exercise instructions with single poses frequently contain *body parts* and *actions*, but **locations** are often left implicit, or are explained in the accompanying images only.

4.1 Semantic Parser

A semantic parser is the first part of our system. Semantic parsing is the process of extracting the basic meaning of natural language texts. In this section, we will analyze the semantic parser we developed to extract the physical exercise instruction text. We try to extract the basic meaning or semantic information of physical exercise instruction texts, isolating *action*, *body part*, and *location*.

Our system works properly to extract the all required information from the exercise instruction sheet if the information is present in the text-based exercise instruction sheet. During our study of exercise instruction sheets, specifically the Physical Exercise Instruction Sheet Corpus (PEISC), we found that usually in single instructions *body parts* and *actions* are usually mentioned but **locations** are often left implicit. For this kind of scenario, our semantic parser extracted only *body parts* and *actions*; otherwise the parser is able to isolate **location** also from the exercise instruction texts.

Two different semantic parsers used for extracting this information were appropriated and tested. Both approaches were found to work properly for our needs; they were able to extract *action*, *body part* and **location** if mentioned in the text-based physical exercise instruction.

Using a Constructional Analyzer

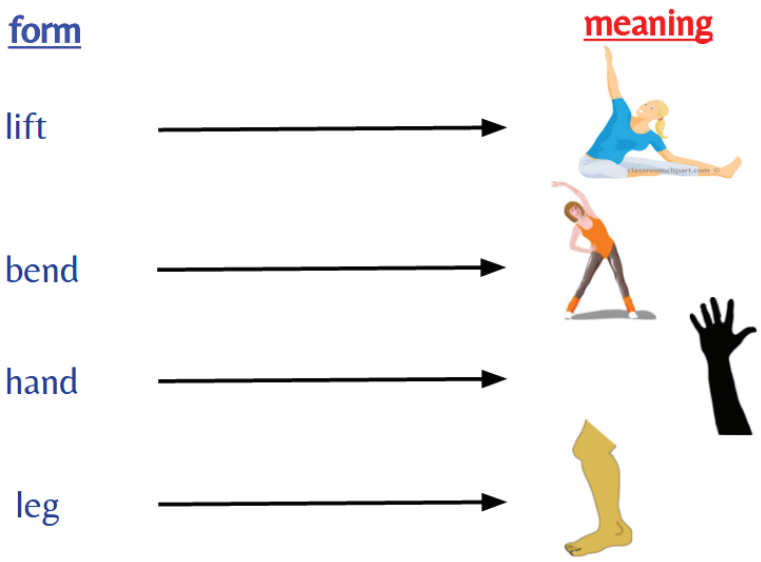


Figure 4.2: Examples of ***form*** and ***meaning; symbolic representations of specific manifestations of the meaning that relates to the lexical forms***

Construction grammar is mainly a linguistic theory which is designed based on the speaker's knowledge [Goldberg, 1996]. Construction grammar basically consists of two parts: ***form*** and ***meaning***. *Form* is used to describe syntactic, morphological or prosodic patterns and *meaning* describes lexical semantics, pragmatics, and discourse structure [Fried, 2014]. In Figure 4.2 we show examples of *form* and *meaning* that are used for exercise instructions. As mentioned earlier, this is mainly used in cognitive linguistics. To understand the natural language text of the exercise instruction sheet for our text to animation system, we employed embodied construction grammar (ECG) because unlike the other construction grammar it doesn't limit itself to mapping between phonological forms and conceptual representations [Bergen and Chang, 2005].

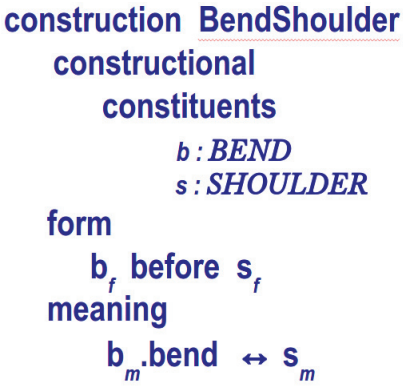


Figure 4.3: Construction Grammar for BendShoulder

We wrote our own construction grammar to extract the semantic information of imperative sentences, mainly for use with physical exercise textual instructions.

In Figure 4.3, we show a specific instance of a construction for ***BendShoulder***. As shown in Figure 4.3, ***BendShoulder*** contains three main elements: *constituents*, *form* and *meaning*. In ***BendShoulder*** there are two different constituents *BEND* and *SHOULDER* each of which is giving some local alias *b* and *s* respectively. The *form* part consists of b_f *before* s_f, which means that BEND will always take place before SHOULDER. The *meaning* part consists of $b_m.bend \leftrightarrow s_m$. This means constituent b has a *bend* role referred to as $b_m.bend$ that is identified with the other constituent referred to as s_m.

In our grammar, we merged similar constructions into one category, e.g, different body parts, different action verbs, etc. as shown in Figure 4.5. There are two different constructions named *BendShoulder* and *BendElbow* with the constituents *Bend, Shoulder* and *Bend, Elbow*. Both shoulder and elbow belong to the human body parts. Therefore, we merged *BendShoulder* and *BendElbow* into one category named *BendBodypart* that consists of the *Bend* and *Bodypart* constituents. For the shoulders and elbows we designed different construction grammar elements, making a subcase of *Bodypart* as shown in Figure 4.5.

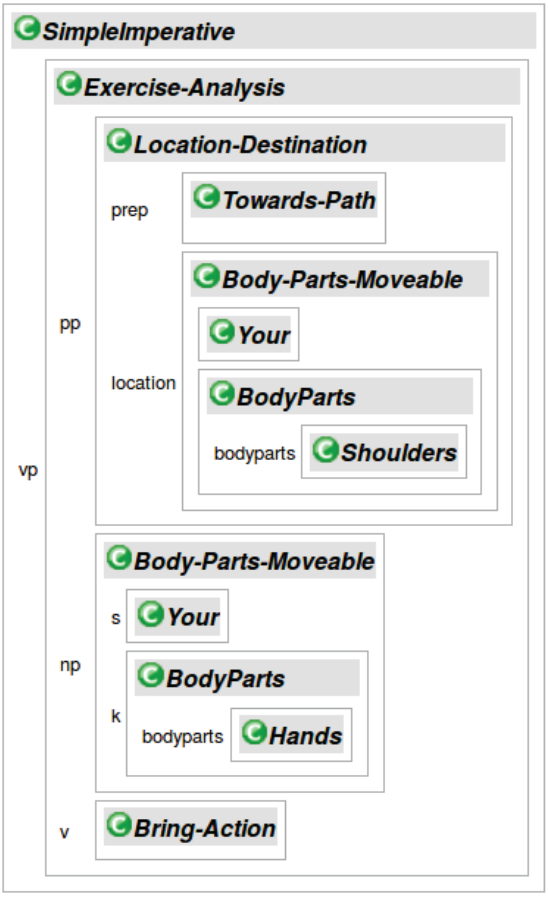


Figure 4.4: Results from the Embodied Construction Grammar

In this approach, we used a constructional analyzer to extract the semantic information. There are different types of formal construction grammars, but in this case we used Embodied Construction Grammar (ECG) to run our own constructional analyzer. We analyzed the sentence structures from our exercise instruction sheet corpus and developed our own construction grammar, covering mainly exercise instructions. Figure 4.4 shows the result of our construction grammar using

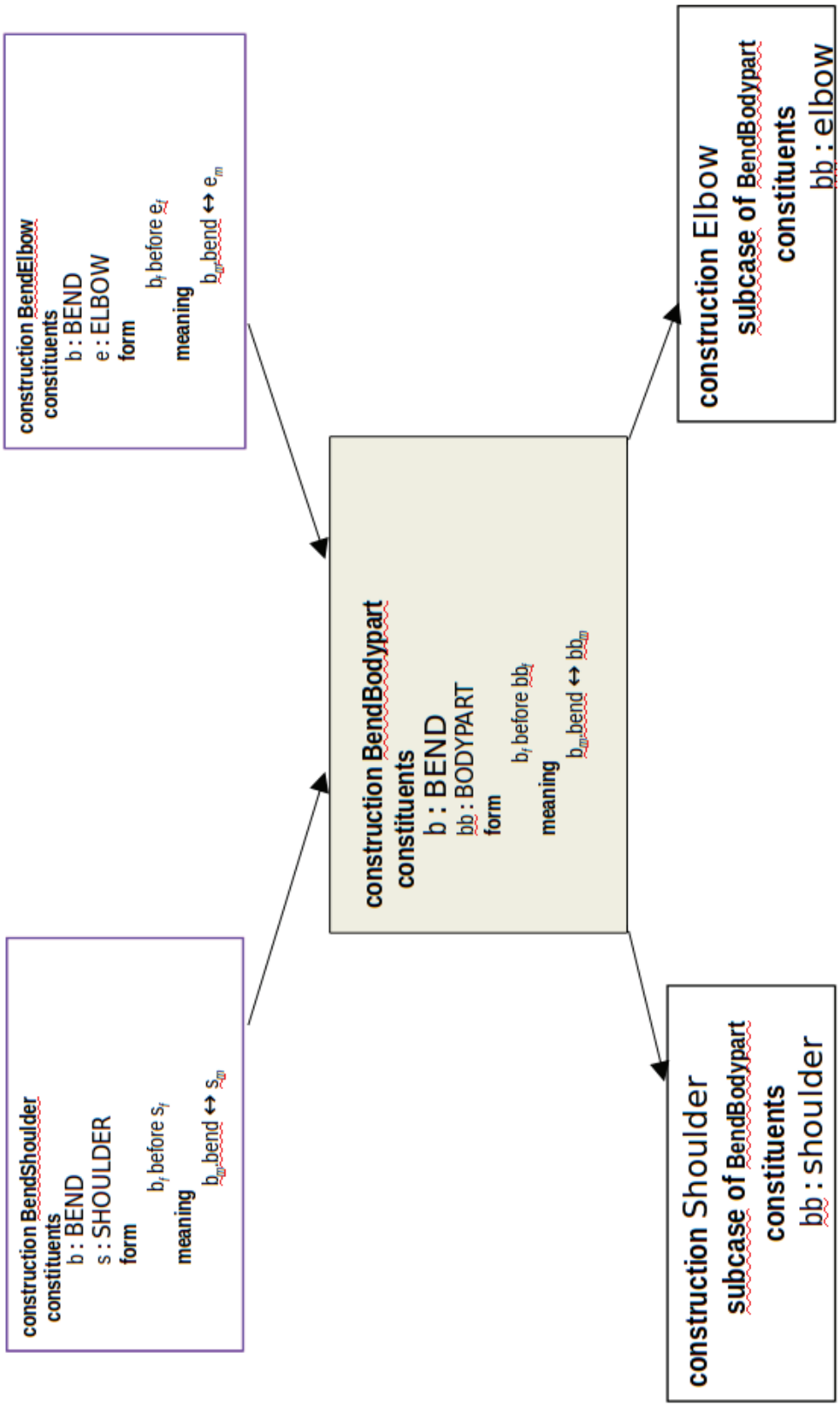


Figure 4.5: Merging BendShoulder and BendElbow

ECG for *Example 1* “Bring your hands towards your shoulders.”. So, from the results, we are able to extract all the require semantic information(*action, bodypart, location*) to create animation from a textual instruction sheet if the information are available in the textual instruction. As shown in the figure, the syntactic structure of the sentence consists of three parts preposition phrase (PP), a noun phrase (NP) and a verb (VB) that is a part of the verb phrase(VP). The required semantic parser also consists of three parts, i.e., ***Location-Destination*** (used for destination location), ***Body-Parts-Moveable*** (the body part involved), and ***Bring-Action*** (an action verb). From the analysis, we can say that construction grammar is working properly according to our necessity for extracting the information present in the exercise instruction. Since Embodied Construction Grammar is already an established tool that can be readily implemented on a robotic platform [Eppe et al., 2016, Feldman et al., 2009], ECG is therefore one of the best candidates to formalize language as per our future pipeline in the framework.

Using a Rule-based Parser

Extracting the semantic information is working nicely by using construction grammar as shown in the previous section. However, it has some limitations when using embodied construction grammar. Basically, when we needed to extract semantic information for multiple sentence instruction it failed because the system is designed mainly for instruction in a single exercise. Therefore, we also designed another semantic parser which we can use for multiple sentence instructions. We can apply this very easily to our framework, where the output from the parsers can automatically proceed to the next step of the framework to design a text to animation system for physical exercise instructions.

As part of this approach, we developed a semantic parser using the *Stanford syntactic parser*. Our system first extracts the syntactic structure of the exercise instruction as shown in Figure 4.6, 4.7 for *‘lift your right arm’*, and *‘bring your hands toward your shoulder then move it down’* respectively.

We set different rules for the semantic parser based on the syntactic structure of the textual instructions of the PEISC corpus. Understanding the syntactic structure and then labeling the words with correct labels or frame names is one of the important aspects of the semantic parser. Therefore, we used different frames or labels to analyze the meaning of different phrases or words of the instructions. We added some frames or labels of our own (e.g. Action, Bodypart, Destination, Direction etc.)

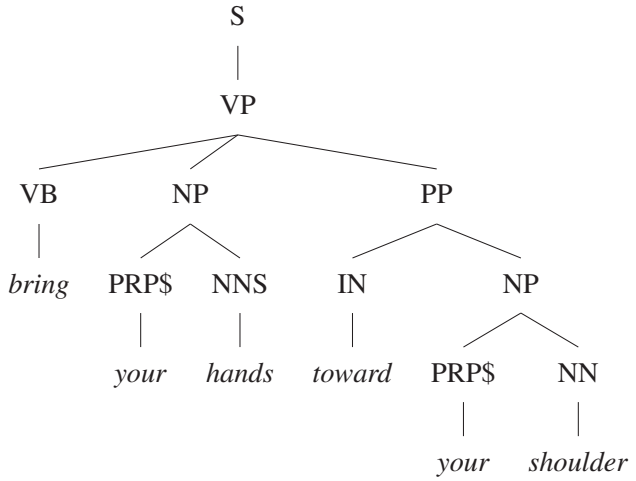


Figure 4.6: Syntactic structure of *'bring your hands toward your shoulder'*

for the analysis of the exercise instructions. The maximum basic frames or labels were adopted from Frame Net [Baker et al., 1998] for our parser.

In our semantic parser, the instructions are initially passed through the Stanford syntactic parser; then the output is matched with different rules in our semantic parser, which had been set before as mentioned earlier. At last, the system gives an output with the most possible matching frames, as shown in Figure 4.9. The ensuing results are shown in Figure 4.9 for two different instructions, *'lift your right arm'* and *'bring your hands toward your shoulder'*. *Action*, *Bodypart*, *Direction* (direction of the location), and *Destination* (destination location) are additional frames that we tested for these two instructions, as displayed in Figure 4.9. Like the construction grammar, we found that by using our proposed semantic parser we were also able to extract all the required information (*action, bodypart, location*) needed to create a correct animation from the textual instruction sheet.

Evaluation & Analysis

The semantic parser's results, which is a list of 13 exercises, are shown in Table 4.1 with their corresponding output from the semantic parser that we developed. The exercise instructions tested consist of both single and multiple sentence structures. The first ten instructions are single sentences and last three instructions consist of multiple sentences; the exercise instructions were taken from the PEISC

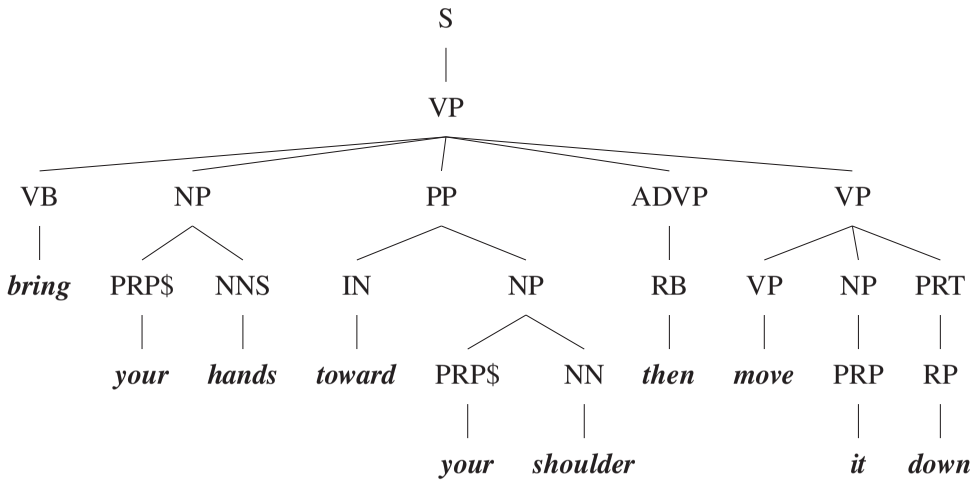


Figure 4.7: Syntactic structure of *'bring your hands toward your shoulder then move it down'*

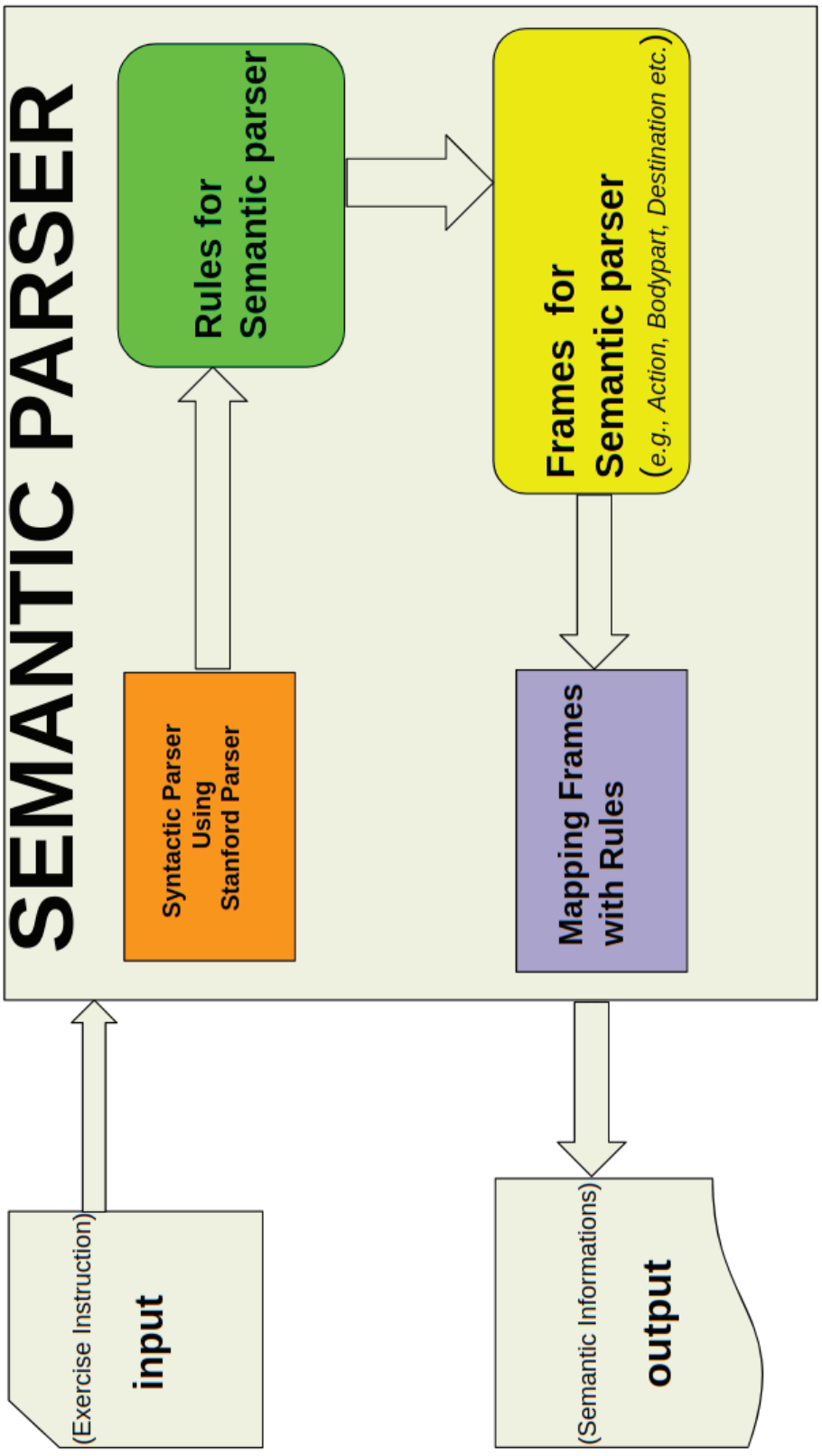


Figure 4.8: Framework of rule-based semantic parser

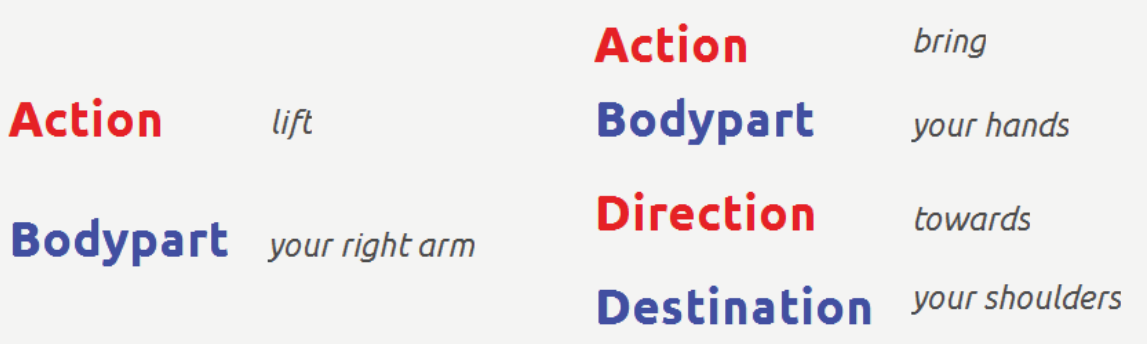


Figure 4.9: Results from our rule-based semantic parser for instruction: left: *lift your right arm*; right: *bring your hands towards your shoulders*

Table 4.1: List of exercises with their corresponding results

S. No.	Exercise	Action	Bodypart	Location	Action2	Location2
1	Bend your left ankle	Bend	your left ankle	—	—	—
2	Tilt your head	Tilt	your head	—	—	—
3	Stretch your leg	Stretch	your leg	—	—	—
4	Bend your right elbow	Bend	your right elbow	—	—	—
5	Raise your left shoulder	Raise	your left shoulder	—	—	—
6	Lower your head	Lower	your head	—	—	—
7	Bring your left arm toward your shoulder	Bring	your left arm	your shoulder	— —	— —
8	Bring your right arm toward your shoulder	Bring	your right arm	your shoulder	—	—
9	Push your right leg toward opposite	Push	your right leg	opposite	—	—
10	Push your left leg toward opposite	Push	your left leg	opposite	—	—
11	Lift your left arm. Move it to your right elbow	Lift	your left arm	—	Move	your right elbow
12	Driving your knee up toward the ceiling	Driving	your knee	ceiling		
13	Lift your right leg and move it to right	Lift	your right leg	—	move	right

corpus. To evaluate the instructions, we used both of the above= mentioned semantic parsers. We mainly focused on extracting action, body part, and location words from the text-based instruction sheets. These sheets are the pre-requirements for developing a text-to-animation system.

From the results, we found that the semantic parser is able to extract all the above-mentioned information that is required to develop a text to animation system. But the parser did not understand

the pragmatics, as we can see in the first six examples. In these examples, the system was not able to extract the implicit location; this is an important criterion needed to create the text to animation system. But from the results, we can say that the semantic parser we developed worked properly to extract the required information that was present in the form of an imperative sentence and is also needed to develop a text to animation system. Both semantic parsers worked properly for first ten sentences, displaying the same output as mentioned in the Table 4.1, However, for last three instructions, which consisted of multiple sentences, we were working only with the rule-based semantic parser. Therefore, for our framework to develop a text to animation system we use our rule-based semantic parser. We can use this parser with more freedom for both single and multiple sentence instructions.

4.2 Bayesian Network

A Bayesian network is the second part of the system. It is a way to obtain implicit information or uncertainty about a subject which in this scenario constitutes physical exercise instructions. This section provides a brief insight into why Bayesian network is a real need for our system and how it will overcome the disadvantages of natural language understanding and obtain the all necessary information from natural language text to generate the animation from the natural language text input for physical exercise instructions.

After acquiring the semantic structure of the instruction, the system must assure adequate actionable information is present that can map body movements. This frequently encompasses information that is implicitly contained in the exercise description and cannot be recovered with the semantic parsing methods which are discussed above. Understanding this type of pragmatics or implicit information is impossible for a semantic parser, as we discussed in the previous section. It is also mentioned in Table 4.1; for first six exercise instructions locations are implicit information which is easily understandable by humans but not by a semantic parser. Also, in Figure 4.10 we have shown missing information during semantic parser processing for two different exercises with respect to single and multiple poses.

In the Figure 4.10 we only mentioned the missing information *action*, *bodypart* and *location*(mainly

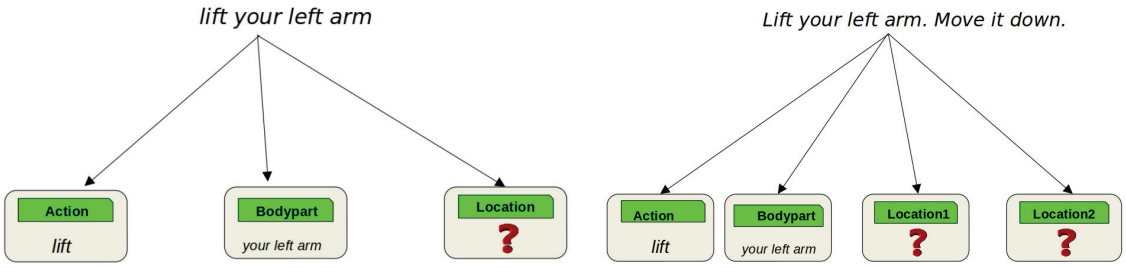


Figure 4.10: Missing implicit information during semantic parsing for *"lift your left arm"* and *"Lift your left arm. Move it down"*

destination location), which are the primary pieces of information needed for developing the text to animation system. There is also a lot of information missing that we can focus on, e.g., source location, what are the positions of other body parts during the exercise, etc. However, to develop our text to animation system we assume other body parts and source locations are static as neutral position (standing with the hands in a resting position), which is known as the N(neutral) pose.

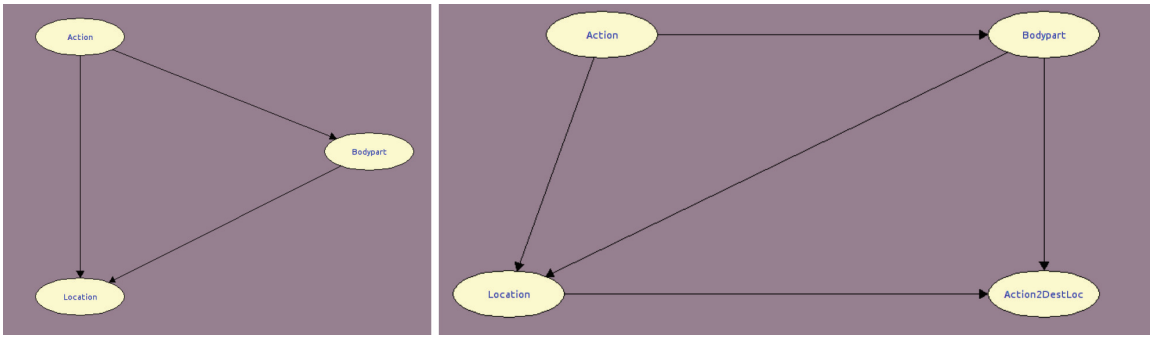


Figure 4.11: Dependency graph for different variables used in Bayesian network

We explored the application of Bayesian networks for explicating the implicit or hidden information in the given exercise instructions. A Bayesian network is very useful for finding the uncertainty of a domain. Therefore, we use the Bayesian network to find the implicit information. The dependency for different variables for use in two different Bayesian networks is shown in Figure 4.11.

For a Bayesian network, the Conditional Probability Table (CPT) is the most important part needed to build an intelligent system. The CPT is the backbone of a Bayesian network. Using the CPT, the network provides the result of the uncertain variable using the probability of other variables which are dependent on that specific variable. So, to increase accuracy, the CPT needs to be more reliable. To build a reliable CPT, researchers have to build a large dataset to analyze the domain in

which he/she wants to build the Bayesian network. In most cases it is very expensive to build a large dataset. However, we built our CPT using a human computation approach with the help of crowdsourcing. Crowdsourcing is cost effective and we get better values by using human brains. To update our CPT, we have developed a survey system consisting of 13 different exercises involving 13 body parts. The survey consists of 44 different videos, and there are three or four videos for every exercise. Using crowdsourcing and this system we build a strong CPT to get accurate implicit information from humans from the physical exercise instruction sheets. We built our Bayesian network using ***SamIam***(Sensitivity analysis, modeling, inference and more) [Darwiche] [Chan and Darwiche, 2001]. We developed two different Bayesian networks. They are:

- ***Bayesian network for single action:*** This belief network, shown in Figure 4.12, is designed for single sentences and is concerned with single actions, e.g., *lift your left arm*. This network consists of three different variables. They are:
  - *Action:* This is an action variable is designed for action verbs in the exercise instructions.In this variable we include 14 values that are listed in Table 4.2 and shown in Figure 4.12.
  - *Bodypart:* This Bayesian model is mainly designed for exercise instructions. Human body parts are always involved in an exercise. Therefore, as a value in this variable, we include all possible moveable human body parts that are required to perform an exercise. Fourteen different body parts are included in this variable and shown in Figure 4.12 and 4.14 and listed in Table 4.2.
  - *Location:* The "Location" variable is designed for target or destination locations of the body parts involved in the exercise, and is basically used to find the implicit target locations of the exercise. We use 22 different locations that are basically the coordinates around a human body, as shown in Figure 4.15.

Table 4.2: The different possible values of the variables

Action	**Bodypart**	**Location**	
lift	l_hip	location1	location2
bring	r_hip	location3	location4

bend	l_knee	location5	location6
push	r_knee	location7	location8
keep	l_ankle	location9	location10
bent	r_ankle	location11	location12
tilt	l_shoulder	location13	location14
rest	r_shoulder	location15	location16
lower	l_elbow	location17	location18
stretch	r_elbow	location19	location20
pull	l_wrist	location21	location22
sit	r_wrist		
reach	HumanoidRoot		
raise	skullbase(head)		

At the beginning, the CPT of the network was built from the data of PEISC corpus. The implicit *location* for the exercise instruction *lift your left arm* is shown in Figure 4.12. From the instruction *lift your left arm* we can extract the action verb *lift* and body part *left arm* using a semantic parser but the location is left implicit. Therefore, as you can see in the figure 4.12, in the Bayesian network we keep the action verb *lift* and the body part *left arm* as evidences in *Action* and *Bodypart* evidence respectively to extract the destination location of the exercise instruction. As shown in the figure 4.12 there are three probable locations, *location2*, *location3* and *location17*. Later we will try to find the most accurate location using the human computation approach.

- ***Bayesian network for multiple actions:*** This belief network is designed mainly for multiple actions with single or multiple sentences. This network has four different variables; three are same and one extra variable named *"Action2DestLoc"*. This variable has the same values as the variable *Location*. In this network, the *Location* variable is the destination of the first action and the source of the second action, whereas *Action2DestLoc* is the destination of the second action.

  An action verb *lift* and body part *left arm* can be extracted using a semantic parser, but the parser is unable to extract the location. As the instruction has two different movements, first

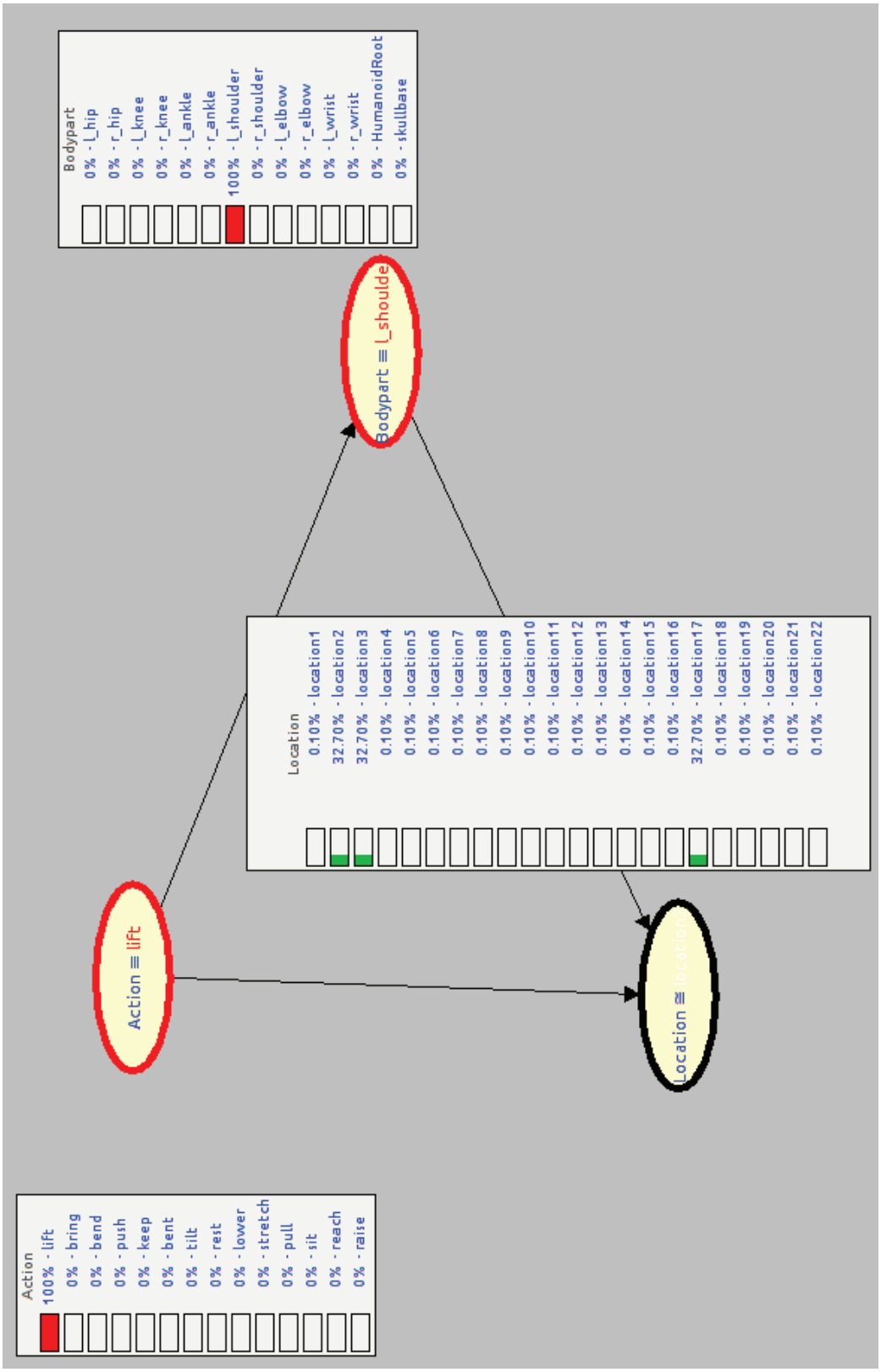


Figure 4.12: Bayes network for *"lift your left arm"*

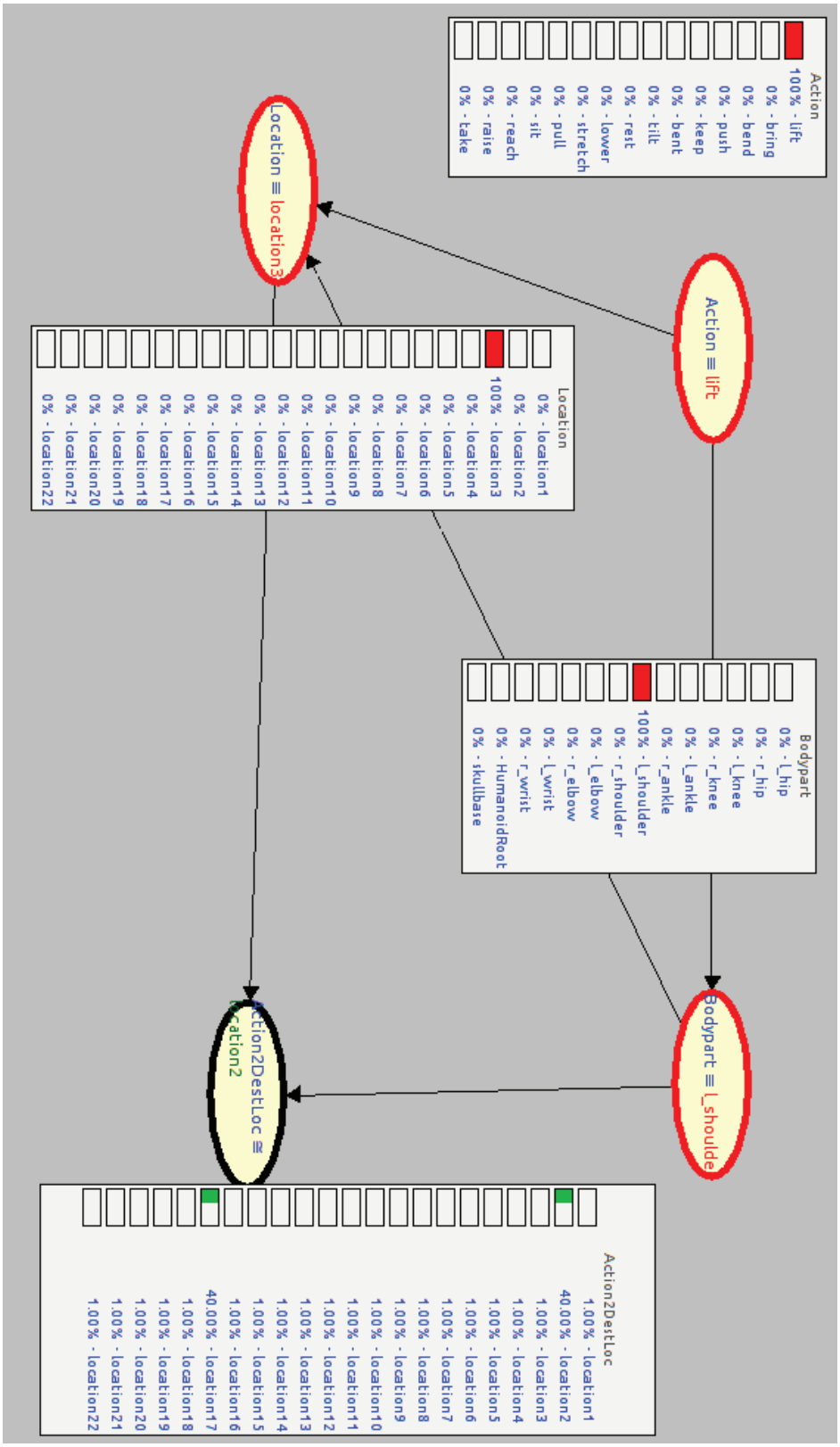


Figure 4.13: Bayes network for *"Lift your left arm. Move it down"*

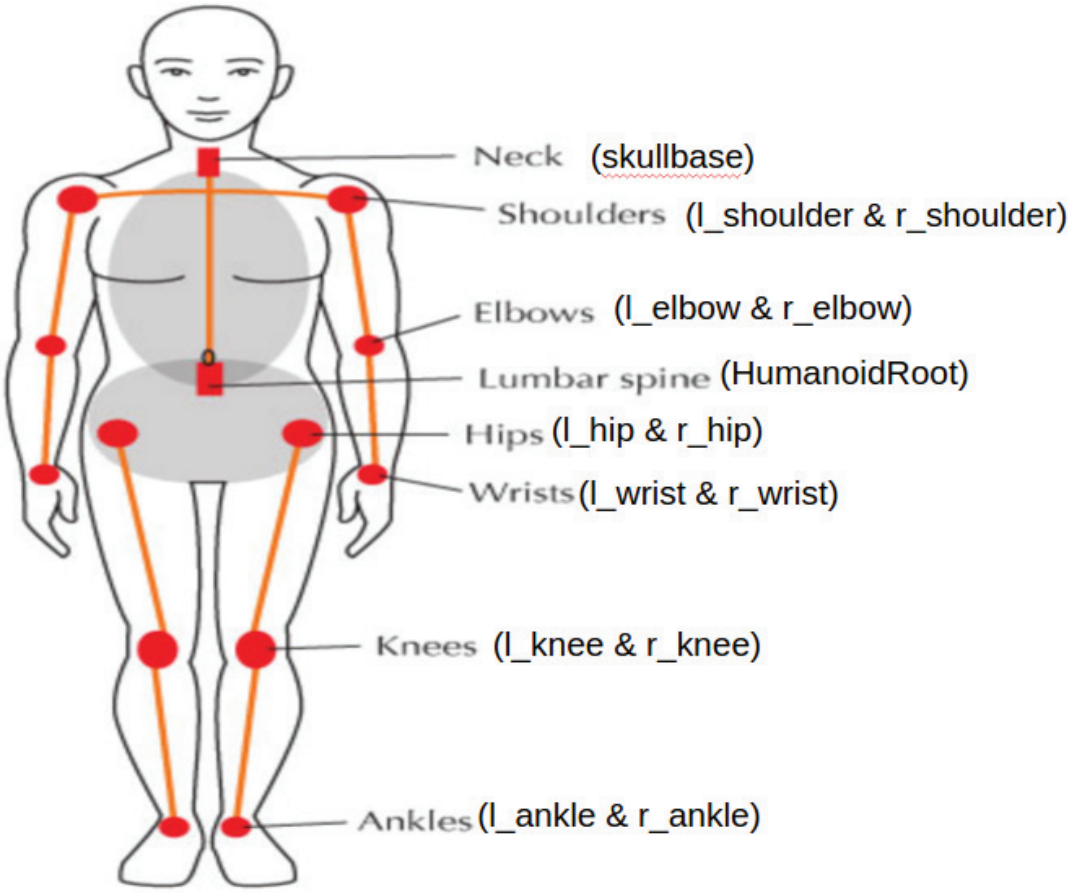


Figure 4.14: Body parts or human joints used in the Bayes network and in our system

the network must extract the implicit location for first movement keeping the action verb *lift* and body part *left arm* as evidence. Later on, the system also keeps the destination location *location3* as evidence to find the final destination for the movement, which has two probable locations, *location2* and *location17*. The result for *Lift your left arm. Move it down,* in our Bayesian network is shown in Figure 4.13.

Variables, values, and the conditional probability table (CPT) compose the parts of the Bayesian network. To build a reliable Bayesian network, the availability of an appropriate as well as reliable CPT is the core challenge. At the start, the model was informed by word frequencies from our PEISC corpus [Sarma et al., 2015] to build the conditional probability table (CPT), as shown in Figure 4.12.

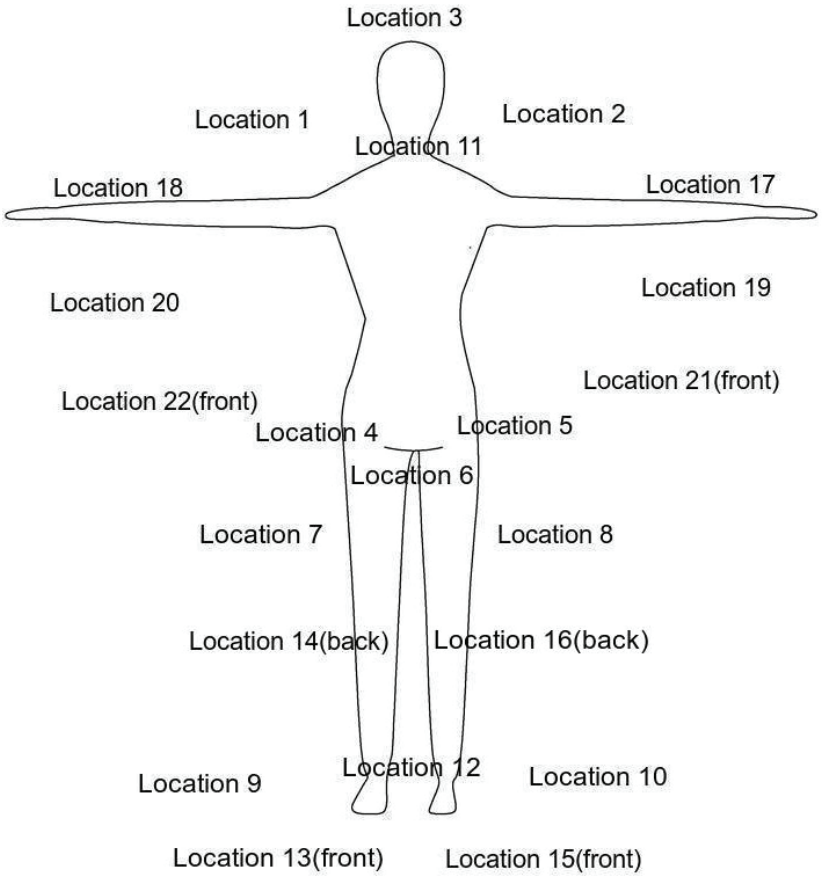


Figure 4.15: Different locations or co-ordinates used in the Bayes network

In this form, if the system is employed to analyze exercise instructions such as *lift your left arm*, the text will first go through the semantic parser and extract the following information:

Action: *lift*

Bodypart: *left arm*

The extracted information *lift* and *left arm* is set as evidence in the Bayesian network as shown in Figure 4.12, leading to the extraction of the probable implicit destination location. However, in this case the result contains three different probable locations, that means there are three different coordinates, location2, location3, and location17 as shown in Table 4.3. Accordingly, we updated the CPT of the Bayesian network using crowdsourcing, which we are going to analyze in Section 4.2.1.

Table 4.3: Results before and after survey

Lift your left arm		**Lift your left arm.** **Move it to your right elbow**	
Before Survey	*After Survey*	*Before Survey*	*After Survey*
location2(32.7%)	location2(22.7%)	location2(40%)	location2(33.33%)
location3(32.7%)	**location3(44.7%)**	location17(40%)	**location17(44.44%)**
location17(32.7%)	location17(30.7%)		

Automatic Updates of the Bayesian Network

The Bayesian network usually has some limitation, such as the system usually work only for those variables and the values using which the network was designed. So, the network will not work or fail if the input is some other values which are not mentioned in the network as it ranges only within the variables and values of the network.

As mentioned earlier, there are three variables in the proposed Bayesian network. The variable *action* contains 14 possible values. This means the system works only if there is one of these 14 values in the exercise instruction. The system will fail if there is another action verb in the exercise instruction, e.g., if there is an exercise instruction *turn your head*, the system will fail because *turn* is not a value in the *action* variable. However, to complete the network it is almost impossible to add all the action verbs of an English dictionary. Furthermore, the system will fail without the required action verb in the instruction, so, we are automatically trying to update the network by keeping the basic 14 action verb at the start. Therefore, the network was designed in a way that the *action* variable will automatically update if it encounters an exercise description with an action verb that is not yet a value of the *action* variable.

Thus, when a textual instruction was caught in the network following the semantic parser the network is trying to map the extracted action verb to *action* variable's list of values. If the action verb is present in the value list, then the system will move to the next step, otherwise the new action

verb will automatically add as a new value to the action variable. Hence, when the system faced with the expression *turn your head*, where *turn* is not a value available as an action variable, the system automatically adds *turn* as a value in the **action** variable and updates the Bayesian network as shown in Figure 4.16.

4.2.1 Automatic CPT Update Using Crowdsourcing

To design a reliable and scalable conditional probability table, we update our CPT using crowdsourcing. To this end, we have developed a system implemented in *Java* and using *SamIam* system [Darwiche] for the Bayesian network. We use SamIam since it is an open source software. Also, it is easy for us to adjust it in our framework. In future, we will implement this subsystem into the fourth stage of our framework, which is a human computation approach.

In this study, we asked people to rate 13 different exercises presented in 44 different exercise videos; every exercise is presented in three or four candidate videos that contain different enactions of potential target destinations. For the visualization of the videos we used animation, which is highly rated compared to depth or infrared and skeleton-based visualizations as analyzed in Chapter 3. We did not use RGB as this kind of visualisation is not possible to generate automatically using machines. The destination locations are basically assumed as per different possible coordinates of the relevant body part. The participants are asked to rate the videos on a scale of 1-5, where 1 stand for the best execution of the exercise instructions and 5 for the worst execution. The original written instructions of all 13 exercises do not explicitly mention the destination location of the body part. Each and every exercise represents precisely one body part or human joint (which are the values of the Bodypart variable in the Bayesian network). The list of the 13 exercises,and their corresponding body parts is provided in Table 4.4.

The scientific goal of the system is to update the CPT of the Bayesian network using human computation assistance. If, for some movement, the there is more than one probability for the destination location, we tried to update the probability of each location using human computation through crowdsourcing. As mentioned earlier, there are three to four candidate videos that are the main probable destination locations of the exercises. With the help of crowdsourcing, we updated the probabilities

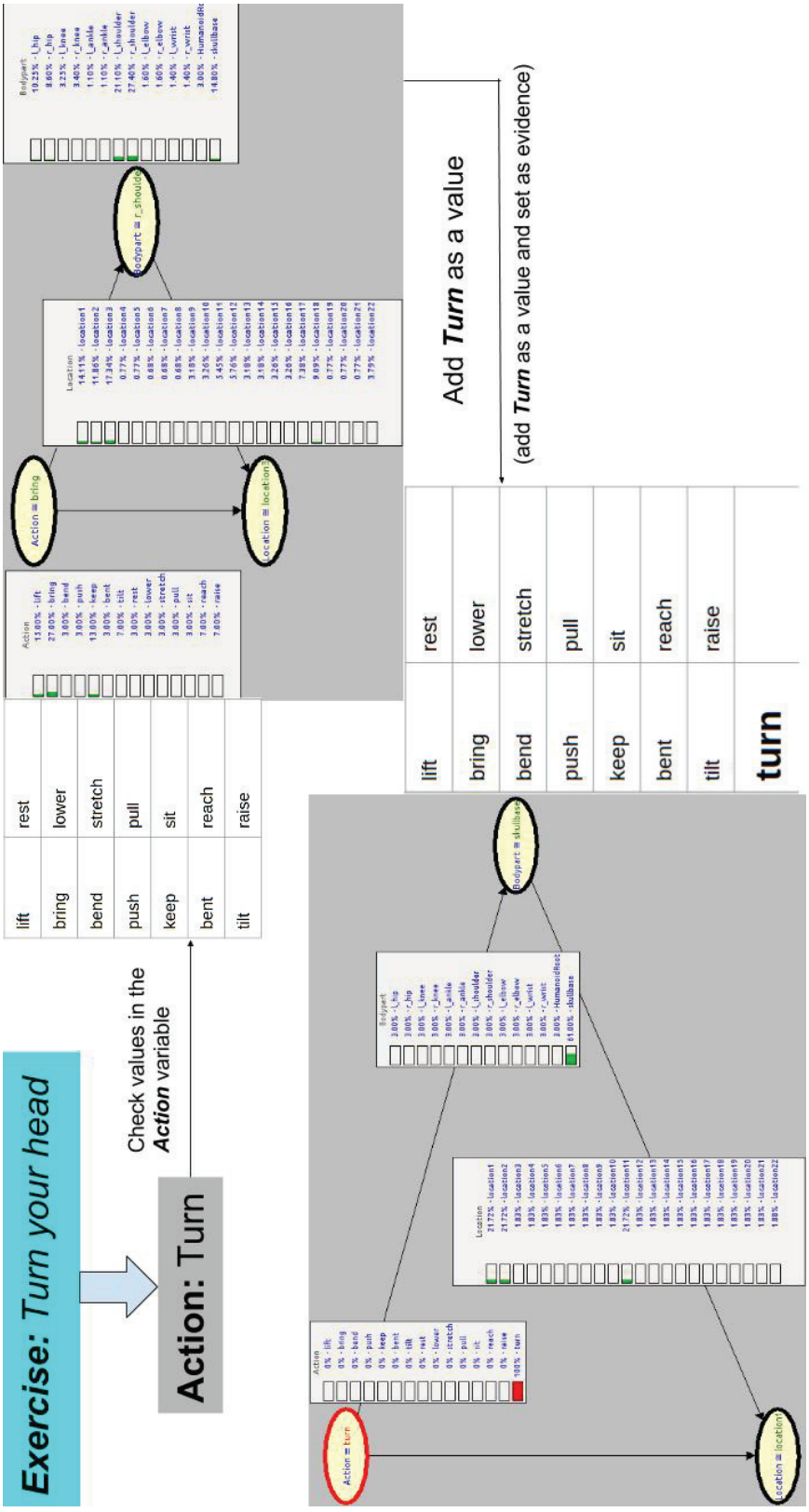


Figure 4.16: Add *turn* as a value in the **action** variable

and tried to find the best candidate video for each exercise as well as the best destination location out of the three or four videos per exercise. There may be more than three or four probable destination locations, but we limited the options to three or four.

Table 4.4: List of exercises with corresponding body parts contained in the Bayesian network

Num.	**Exercise**	**Corresponding body part**	**Num. of videos**
1	Lift your left arm	l_shoulder	3
2	Lift your right arm	r_shoulder	3
3	Bend your left ankle	l_ankle	3
4	Bend your right ankle	r_ankle	3
5	Bend your left knee	l_knee	4
6	Bend your right knee	r_knee	4
7	Bend your left wrist	l_wrist	3
8	Bend your right wrist	r_wrist	3
9	Bend your left leg	l_hip	4
10	Bend your right leg	r_hip	4
11	Bend your left elbow	l_elbow	3
12	Bend your right elbow	r_elbow	3
13	Tilt your head	skullbase	4

The system was updated using the following heuristic. If a participant rated a video as quality level 1, 25% of the probability of the remaining videos (two or three) of the *Location* variable which corresponds to that body part decreases and is added to those with the clicked evidence. If a participant clicks quality level 2, 3, or 4, the CPT decreases the probability by 20%, 15% and 10% respectively; and if a video is rated 5, the CPT remains same without any change. Figure 4.17 shows how the system looks for 3 and 4 videos.

Considering the expression *'lift your left arm'* as an example, the model originally contained

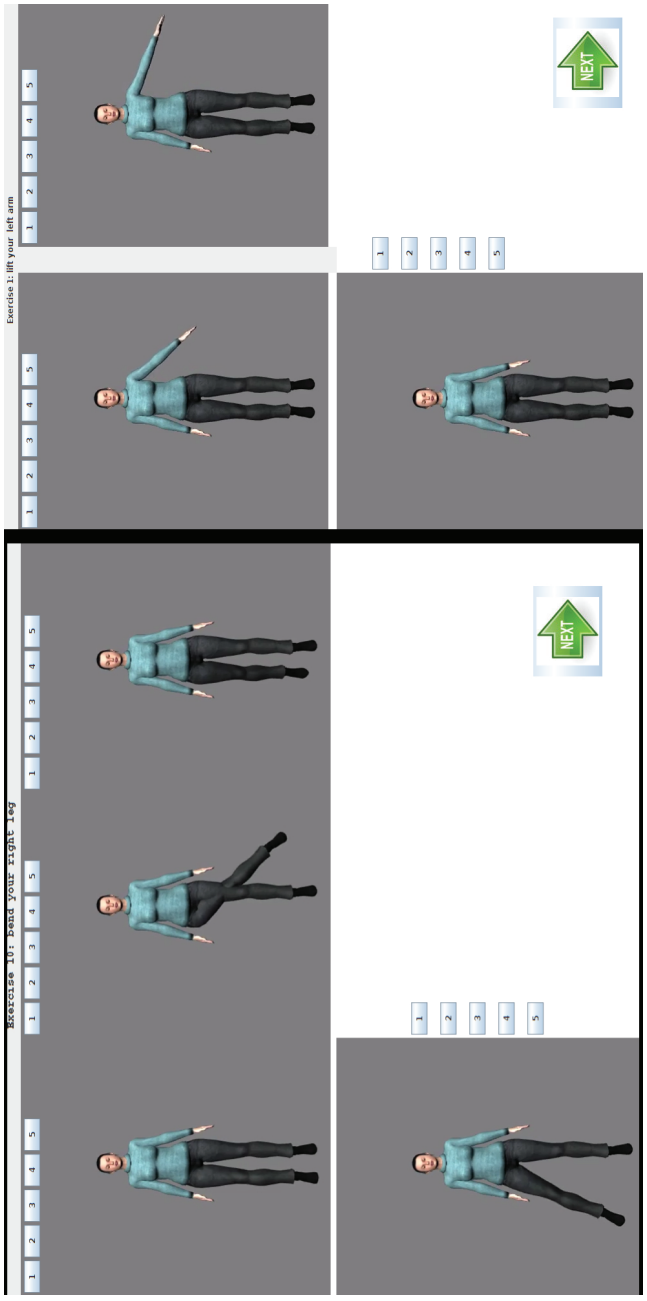


Figure 4.17: Screenshot of automatic CPT update application; left: *bend your right leg(r_hip)*; right: *lift your left arm(l_shoulder)*

three different possible destination locations with the same probability, as shown in Figure 4.18. In Figure 4.17, the screenshot of the survey application is shown and every video is presented together with a rating scale (1-5). If a participant rated a video as 1, 2, or 3, the probability of the other two *location* values of the *Location* variable that are represented in the candidate videos decreases by 25%, 20%, and 15% respectively and those amounts are added in the rated video (location). If a participant rated a candidate position video as a 4 or 5, the CPT was not adjusted.

In cases with four possible locations, the system produces four possible videos as shown in Figure 4.17. In these cases, if a participant rated the video as a 4, then the probability decreases by 10% and there is still no change for a rating of 5. Using this approach, the probabilities encoded in the network change based on the crowd's input. For example, the probability for location3 for *"lift your left arm"* increased from 32.7% to 44.7% after updating the CPT using crowdsourcing as shown in Figure 4.18 and listed in Table 4.3. A total of 32 people participated in our study (the participants are mainly students who said that they regularly perform physical exercises). The study was conducted offline to keep track of the participants' records.

Before the study, the probability for destination locations are same for each and every exercise. But the probability of destination locations changed based on the participants' ratings of the exercise videos. The ratings for all 13 exercises used in this study are shown in Figure 4.19. The top eight exercises, as shown in the figure, consist of three candidate videos or three different probabilities, whereas the five low-rated exercises consist of four candidate videos.

Here, we have shown how our whole system is working on exercise instructions containing implicit information. If we take the example of *lift your left arm*, the system first extracts the semantic information of the instruction using the semantic parser as shown in Figure 4.3 and Figure 4.9. From the first part of our system, the semantic parser, we get the *action* (i.e., *lift*) and the *body part* (i.e., *left arm*). However, the implicit destination (*location*), is missing after the semantic analysis. Therefore, the system moves to the second part, the Bayesian network, to extract the implicit information *location* for the destination of the exercise. The system automatically sets *lift* for the *action* variable and *l_shoulder* (left arm) for the *body part* variable as evidence for the Bayesian network, as shown in Figure 4.12. The most probable level of the implicit variable for *location* (destination) is automatically determined using this system and returns *location3* (overhead) in this example as shown in

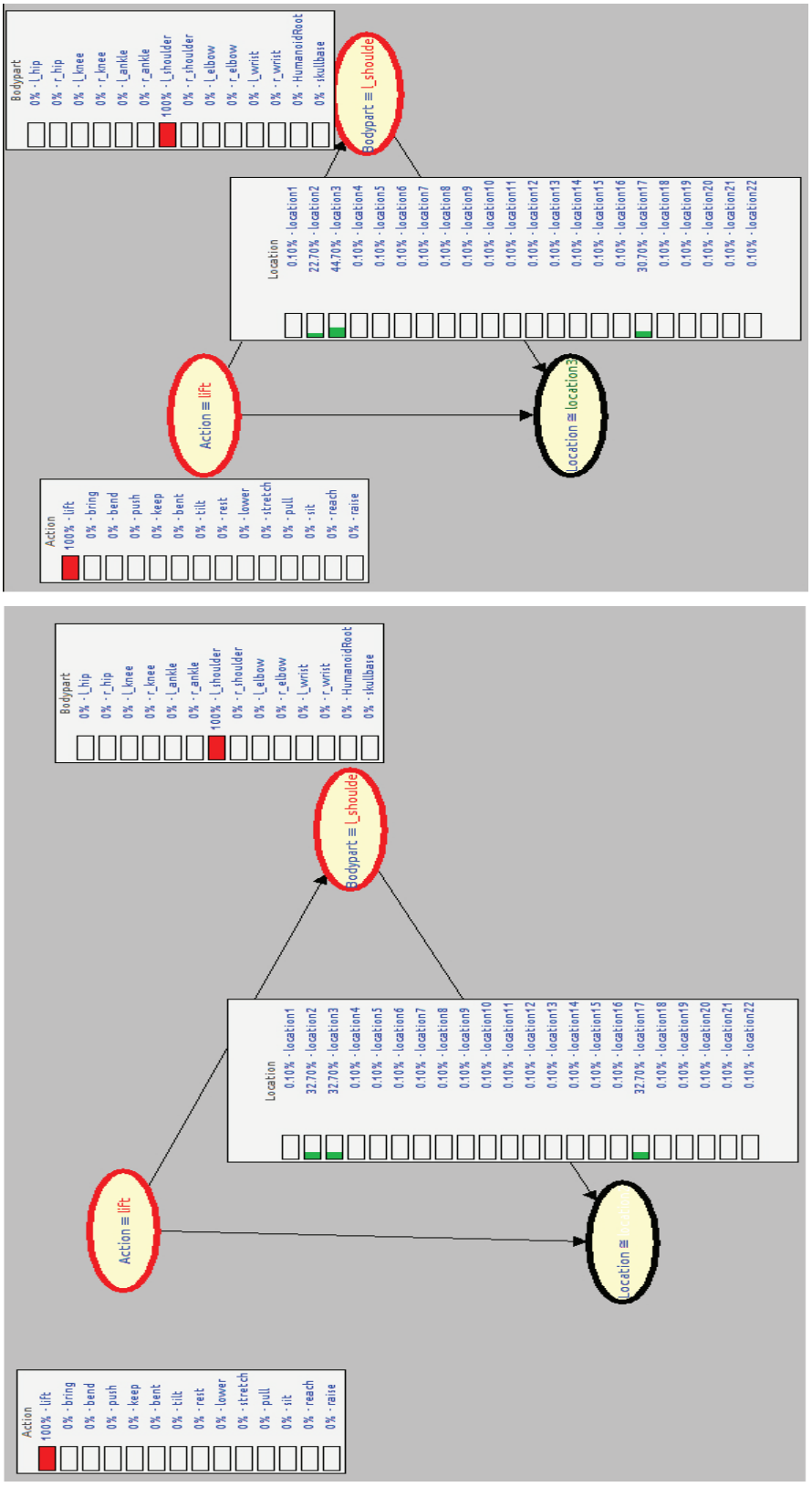


Figure 4.18: Bayesian network for *lift your left arm*

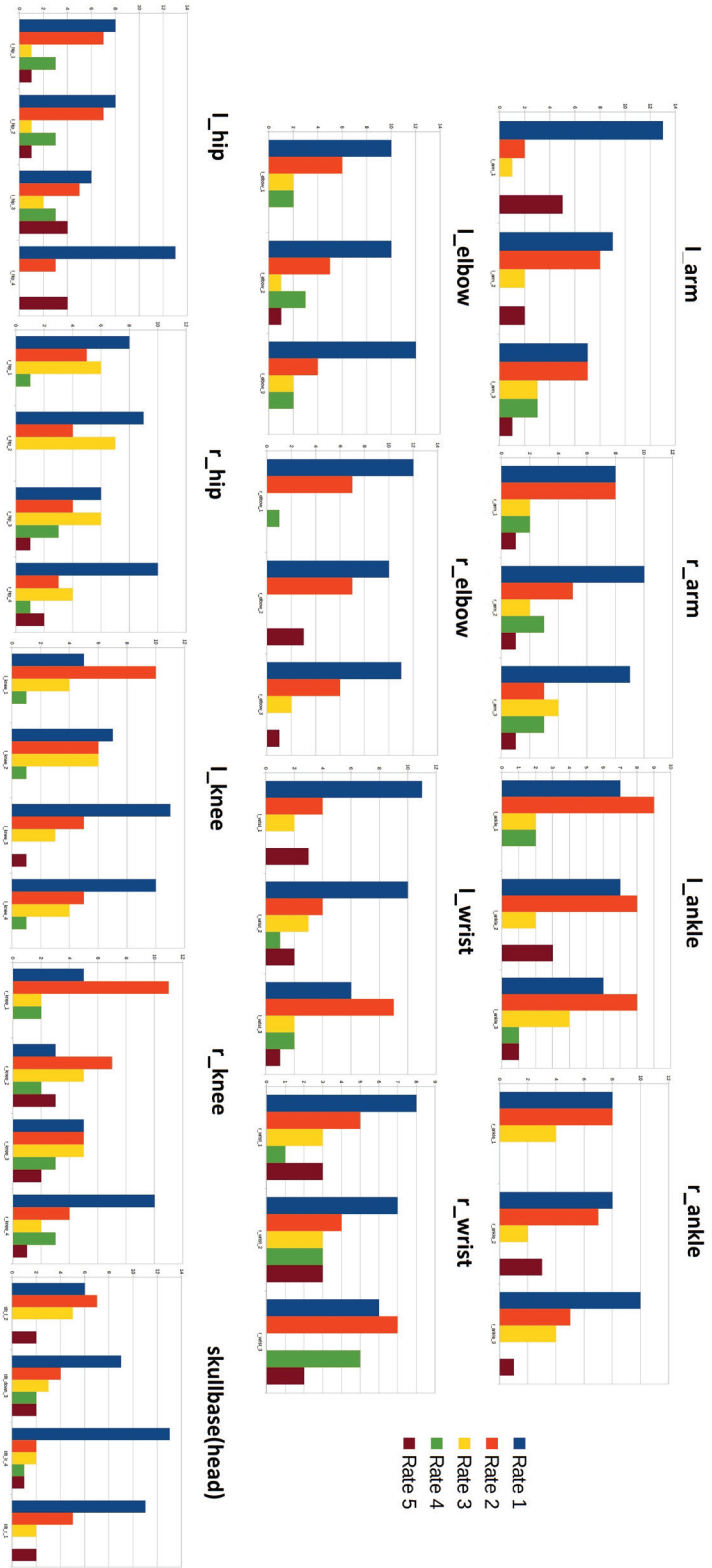


Figure 4.19: Rating of different candidate videos for 13 different exercises

Figure 4.18. The overall result of the instruction *lift your left arm* is shown below:

Action verb: lift

Bodypart: your left arm

Location: overhead

.

Table 4.5: List of exercises with their corresponding results

Exercise	**Action**	**Bodypart**	**Location**
Bend your left ankle	Bend	your left ankle	location 10
Tilt your head	Tilt	your head	location 1, location 2
Stretch your leg	Stretch	your leg	location 10
Bend your right elbow	Bend	your right elbow	location 1
Raise your left shoulder	Raise	your left shoulder	location 3
Lower your head	Lower	your head	location 11
Bring your left arm toward your shoulder	Bring	your left arm	location 1
Bring your right arm toward your shoulder	Bring	your right arm	location 2
Push your right leg toward opposite	Push	your right leg	location 10
Push your left leg toward opposite	Push	your left leg	location 9

As shown in Figure 4.18 and Figure 4.20, the probable location before and after the survey for exercise instruction *"lift your left arm"* and *Lift your left arm. Move it t down* are shown in Table 4.3. In Table 4.1, in a list of ten exercises we found that in natural language textual exercise instruction sometimes locations are missing. Therefore, again for the same ten exercises, we used our Bayesian network to find the missing locations from the textual exercise instructions. The results are shown in Table 4.5. Using the selected fourteen body parts, the system almost ascertained the accurate location which is implicit in each exercise instruction.

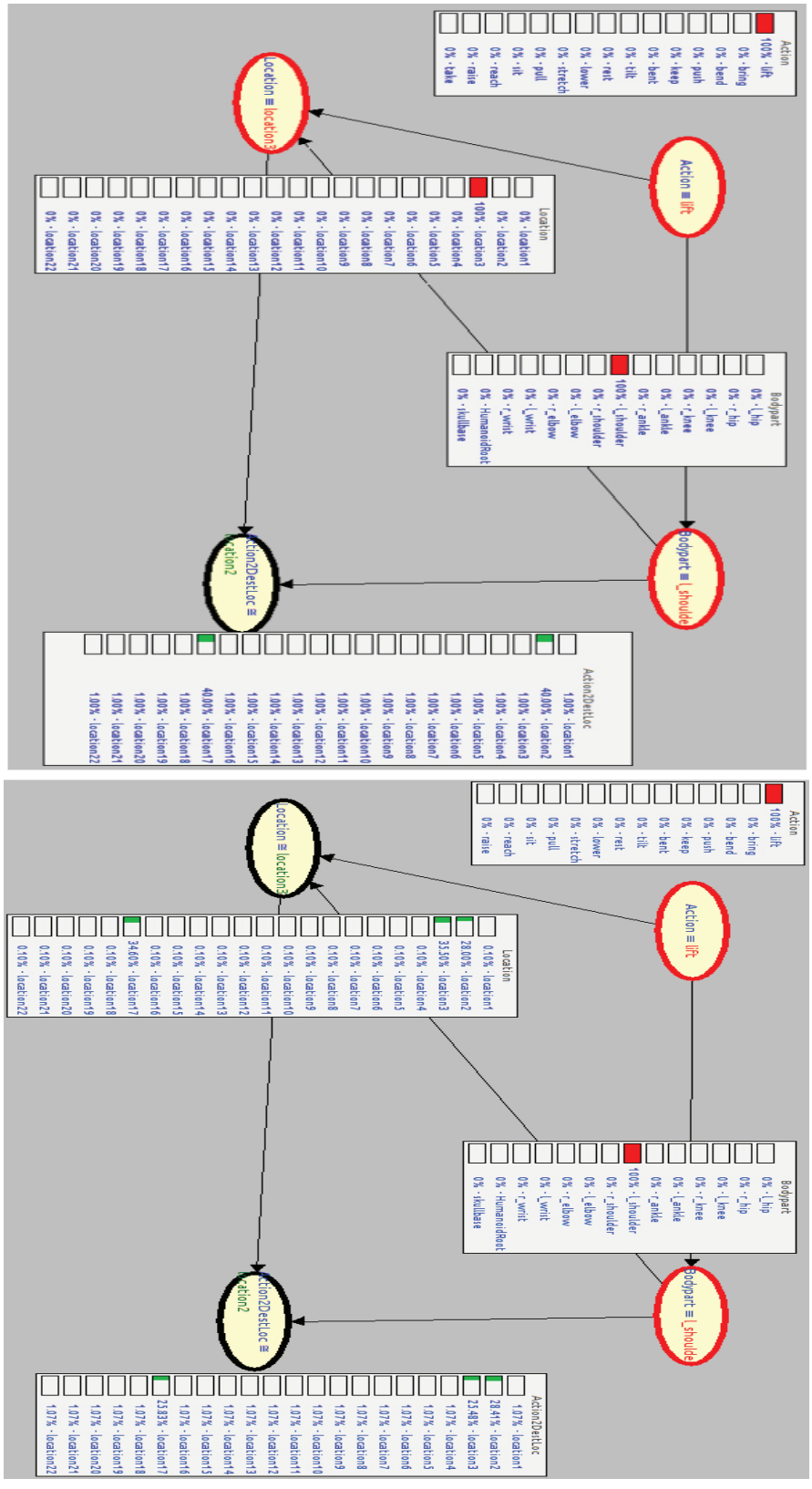


Figure 4.20: Bayesian network for *Lift your left arm. Move it down*

4.3 Animation System

The third stage of our model, as shown in the Figure 4.1, is to make an animation video from the information retrieved from first and second step of the system respectively semantic parser and Bayesian network. The information collected is primarily *action*, *body part* and *location*. It is also challenging to select an appropriate way to create an animation file for animated video. However, as discussed in Chapter 2(State of The Art), the creation of an animation video for physical exercise instruction needs a humanoid kind of animation, which can provide the whole body movement comparing it to other animation categories. The creation of an animated video from physical exercise instruction requires much information as previously discussed, such as which bodyparts are involved, the place where the movement was originated and its destination, the type of action used during the exercise and so on. Already, from the first two parts of the system, the system retrieves all the required information even certain implicit information, while the starting position of the performing actor or humanoid character has always been kept in neutral position or N-position, an example of N-pose is shown in Figure 4.21 which is created using MakeHuman[1].

As mentioned, the creation of the animation video from the source requires three basic information, namely action, bodypart and source and destination location involvement for the exercise. Actions are assumed according to the action verb present in the exercise instruction and already obtained from the semantic parser. The body elements currently included in this exercise are currently confined, as shown in Figure 4.14 and discussed in the previous section, to the fourteen basic bodypart or physical joints. The last requirement is a movement's source and place of destination, which are basically the coordinates around our human body. Twenty two different coordinates around a human body are used to support text-to-animation system as illustrated in Figure 4.15. As mentioned, the starting position is an N-pose so only a destination location is required for single pose exercise instructions that is often left implicit in the exercise instruction and the system has already obtained those with the aid of crowdsourcing using the combination of semantic parser and Bayesian network. The exercise instruction with two composite poses needs a minimum of three coordinates in which the first is identical as the last N-pose, the second is the destination for the first position and the same has been used as the beginning position for the second pose, e.g. *Lift your left arm. Move it down.*

[1] `http://www.makehumancommunity.org/`; Access date: 25 February 2019

Figure 4.21: Example of N-position.

At present the animation files are created using two different methods. We mainly use *Behavior Markup Language* (BML) [Kopp et al., 2006], which runs on the *Artificial Social Agent Platform* (ASAP) framework [Kopp et al., 2014] to generate the targeted exercise animation. Also, we use Humanoid Animation (H-Anim), which is basically a virtual reality markup language (VRML). Using H-Anim we generate the 3D animation of the targeted exercise.

ASAP and BML

ASAP is an environment for embodied agents who are engaged in fluid human discussions with a degree of real time flexibility and a smooth interaction, similar to what humans usually expect of each other. In the Figure 4.22, the entire design of the architecture is shown which consists of two central parts, the left hand part includes the sub-system for behavior generation and the right part the

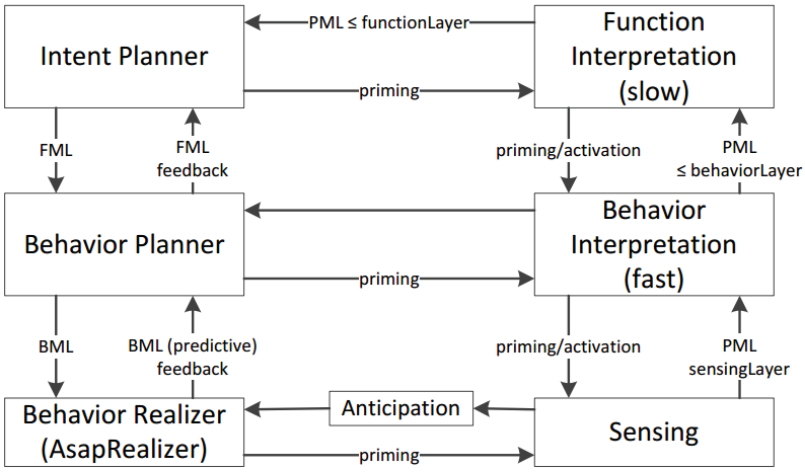


Figure 4.22: ASAP Framework [Kopp et al., 2014].

sub-system for behavior processing. All ASAP modules run simultaneously and can be transmitted through asynchronous, incremental messages. The Intent planner specifies communication objectives, intended messages and interactional goals in Functional Markup Language (FML) during generation and is responsible for monitoring the state of discourse, including the basis status of propositions. The Behavior Planer is used to translate intentions into BML specified surface behaviors. The Behavior Realizer adopts BML behavior requirements, transforming them into an behavior of an embodied conversational agents (ECA) [Kopp et al., 2014].

Behavior markup language or BML is a language based on XML that can be embedded in a bigger XML message or document by simply starting a $< bml >$ block and completing it with behaviors that an agent needs to implement as shown in Figure 4.23.

Behaviours are listed one by one, at the same level in XML hierarchy. In BML, $< posture >$ elements are used for keyframe animations with humanoid characters and correspond to human joints or body parts, such as *left arm* (l_shoulder) as indicated in Figure 4.23. As shown in following Figure 4.24, three different coordinates inside the *posture* element are used for moving a bodypart or human joint, they are:

- *First Coordinate:* This coordinate is used to move a bodypart towards left or right.
- *Second Coordinate:* For forward or backward movements of a bodypart the second or middle

```
<bml xmlns="http://www.bml-initiative.org/bml/bml-1.0"  id="bml1" >
 <murmlgesture id="gesture1" start="2" xmlns="http://www.techfak.uni-bielefeld.de/ags/soa/murml">
    <murml-description>
   <dynamic>
    <keyframing easescale="10" easeturningpoint="0.5" >
 <phase>
<frame ftime="5">
 <posture> Humanoid (l_shoulder 3 170 0 -30) </posture>
</frame>
<frame ftime="10">
 <posture>    Humanoid (l_shoulder 3 0 0 0)    </posture>
</frame>
 </phase>
  </keyframing>
   </dynamic>
 </murml-description>
</murmlgesture>
 </bml>
```

Figure 4.23: A BML example for *lift your left arm*

coordinates are used.

- *Third Coordinate:* Third and last coordinate is used for front or back facing movement of a bodypart.

Coordinates are represented mainly in degrees as shown in Figure 4.23, first coordinate represent 170, which means l_shoulder moves 170 degree towards left.

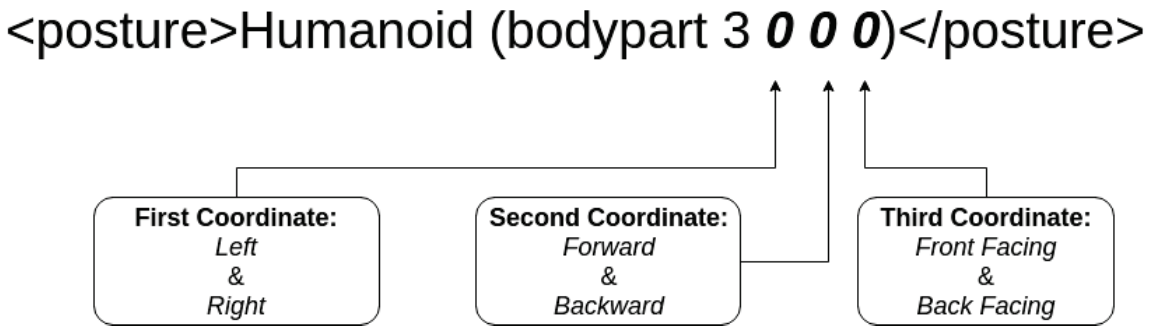


Figure 4.24: Coordinates of a BML file

$< frameftime >$, represents the time required to complete one pose mentioned in the posture element in the above Figure. The BML file shown in the Figure 4.23 is *lift your left arm*, where a pose

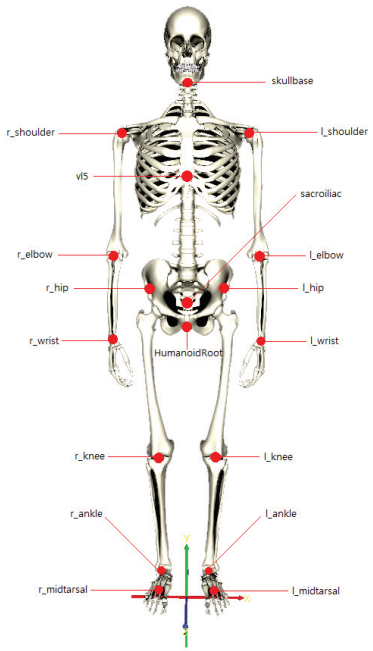
needs three coordinates to complete the task, they are:

- *Starting Coordinate:* As mentioned earlier, starting coordinate of any exercise or pose is neutral position or N-pose, therefore the animation file did not include it as it is default and it always start from 0 second.

- *Destination Coordinate: l_shoulder 3 170 0 -30* is destination location of left arm for instruction *lift your left arm* as shown in Figure 4.23. The posture is included within frame ftime 5 seconds which means, it represents to complete destination location from starting takes five seconds.

- *Final or End Coordinate:* The final and end coordinates or location is the same as the starting position which is N-pose and the coordinates for same is mentioned in the Figure 4.23 as *l_shoulder 3 0 0 0*. This posture is included within frame ftime 10 seconds that means, it takes five seconds from the destination coordinates to final coodinates or starting position.

H-Anim




```
humanoid_root : sacrum
  sacroiliac : pelvis
  | l_hip : l_thigh
  | | l_knee : l_calf
  | |   l_talocrural : l_hindfoot
  | |     l_metatarsophalangeal : l_middistal
  | r_hip : r_thigh
  |   r_knee : r_calf
  |     r_talocrural : r_hindfoot
  |       r_metatarsophalangeal : r_middistal
  vl5 : l5
    skullbase : skull
    l_shoulder : l_upperarm
    | l_elbow : l_forearm
    |   l_radiocarpal : l_hand
    r_shoulder : r_upperarm
      r_elbow : r_forearm
        r_radiocarpal : r_hand
```

Figure 4.25: 18 joints character of H-Anim with respective hierarchy

H-Anim is a virtual reality markup language. In order to enable sharable skeletons, bodies, and animations, the H-Anim supports a broad range of articulated characters including naturally correct human model. Creating an animation using humanoid character, understanding number of human joints used in the character is a very important thing. Number of joints defined for a humanoid character is known as the level of articulation (LOA). According to the requirement of the scenario, the scene is described with different numbers of human joints. The lowest possible level of articulation for a humanoid character is a skeletal hierarchy that only contains a HumanoidRoot Joint. It is said that a humanoid character with fourteen joints as *low level of articulation*, whereas with 72 joints a humanoid character is said as *high level of articulation.*

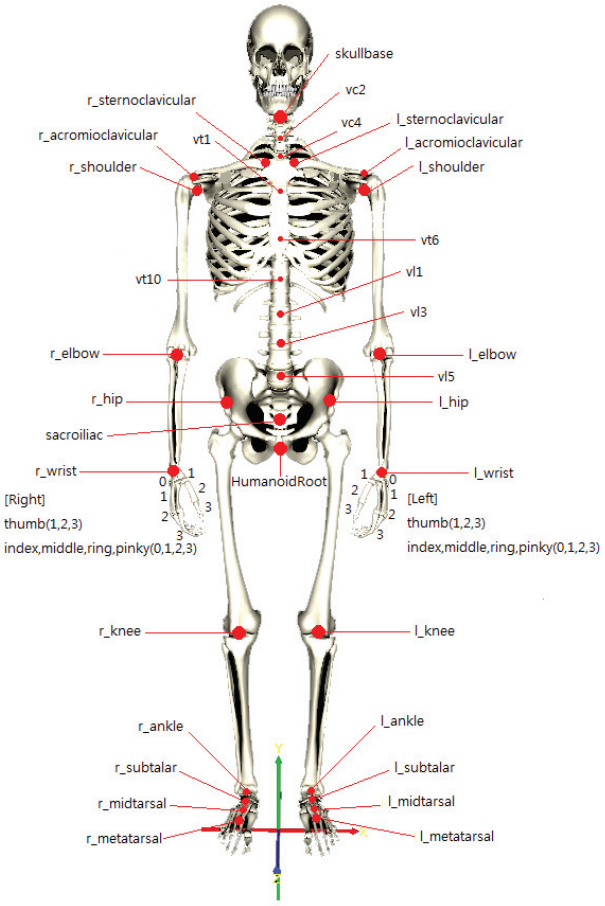


Figure 4.26: 71 joints character of H-Anim

There are different sets of joints or level of articulation use for H-Anim humanoid character[2], they

[2]http://www.web3d.org/documents/specifications/19774-1/V2.0/HAnim/concepts.html

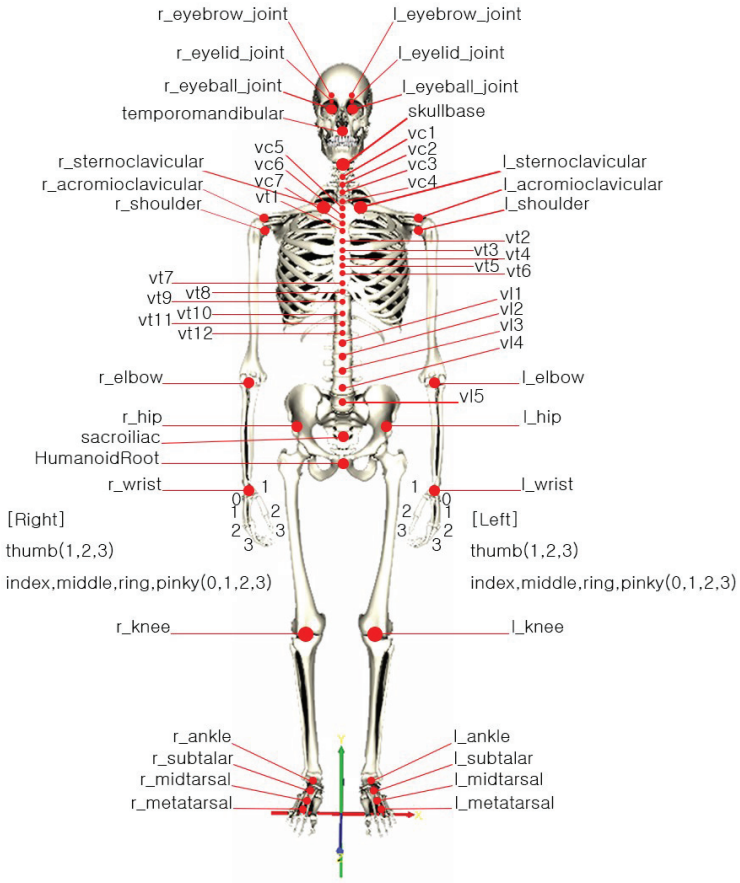


Figure 4.27: 94 joints character of H-Anim

are as follows:

- ***LOA 0:*** It contains only one human one joint which is HumanoidRoot, as mentioned earlier it is known as lowest level of articulation.
- ***LOA 1:*** This is for a humanoid which consists of 18 joints, one of the most simple hierarchy of human joints. The hierarchy and skeleton of the same is shown in Figure 4.25.
- ***LOA 2:*** It consists of 71 human joints as shown in Figure 4.26, hierarchy for LOA 2 is shown in Appendix B.
- ***LOA 3:*** LOA 3 represent humanoid character with 94 joints as shown in Figure 4.27 and hierarchy is shown in Appendix B.

- ***LOA 4:*** It is the recent updated level of articulation known as LOA 4, which consists of 148 human joints.

KoreanCharacter12Sun[3] is used as a humanoid character in the text-to-animation system to develop 3D animation video from the exercise instruction. As mentioned earlier, in our text to animation system we used 14 human joints. The mappings of 14 human joints used in text-to-animation system along with the H-Anim joints(LOA 1) of the humanoid character is shown in Table 4.6.

Table 4.6: Mapping H-Anim joints(LOA 1) with our proposed 14 joints

Sl. No.	**H-Anim joints**	**Our Proposed joints**
1	Humanoid root	HumanoidRoot
2	l_hip	l_hip
3	l_knee	l_knee
4	l_ankle	l_ankle
5	l_midtarsal	**NA**
6	r_hip	r_hip
7	r_knee	r_knee
8	r_ankle	r_ankle
9	r_midtarsal	**NA**
10	sacroiliac	**NA**
11	vl5	**NA**
12	l_shoulder	l_shoulder
13	l_elbow	l_elbow
14	l_wrist	l_wrist
15	r_shoulder	r_shoulder
16	r_elbow	r_elbow
17	r_wrist	r_wrist
18	skullbase	skullbase

[3] `http://www.web3d.org/x3d/content/examples/Basic/HumanoidAnimation/KoreanCharacter12SunIndex.html`

H-Anim consist of the following objects[4]:

- ***Humanoid:*** It is the root of H-Anim which is attached to all other parts of the humanoid character. It stores the information about the skeleton, landmark, and geometry as well as information about joint, segment, site and displacer objects. It also stores the information about the author's name, email, copyright, etc. A sample structure of Humanoid is shown in below:

PROTO Humanoid [

field	SFVec3f	bboxCenter	0 0 0
field	SFVec3f	bboxSize	-1 -1 -1
exposedField	SFVec3f	center	0 0 0
exposedField	MFString	info	[ ]
exposedField	MFNode	joints	[ ]
exposedField	SFString	name	""
exposedField	SFRotation	rotation	0 0 1 0
exposedField	SFVec3f	scale	1 1 1
exposedField	SFRotation	scaleOrientation	0 0 1 0
exposedField	MFNode	segments	[ ]
exposedField	MFNode	sites	[ ]
exposedField	MFNode	skeleton	[ ]
exposedField	MFNode	skin	[ ]
exposedField	SFNode	skinCoord	NULL
exposedField	SFNode	skinNormal	NULL
exposedField	SFVec3f	translation	0 0 0
exposedField	SFString	version	"2.0"
exposedField	MFNode	viewpoints	[ ]

]

- ***Joint:*** The joint object is attached to other joint objects and the humanoid object. It basically represents different human joints that are the main body parts.

[4] http://h-anim.org/Specifications/H-Anim200x/ISO_IEC_FCD_19774/concepts.html

- ***Segment:*** This specifies the attributes of the physical links between the joints of the humanoid character.
- ***Site:*** This specifies locations with which known semantics can be associated.
- ***Displacer:*** This describes information about the range of movement allowed for an object in which it is embedded.

As described in the case of BML, animation section is the backbone of an animation file. A sample of the animation part of the H-Anim file for *lift your right arm* is shown in Figure 4.28. As shown in the Figure, top section describe the coordinates of the r_shoulder which represents right arm of the bodypart. The below section *loop* is used to describes an animation played for a single occasion or in a loop, if a loop is TRUE, animation is played in a loop, if the animation wishes to be played only on a single occasion it is written as FALSE.

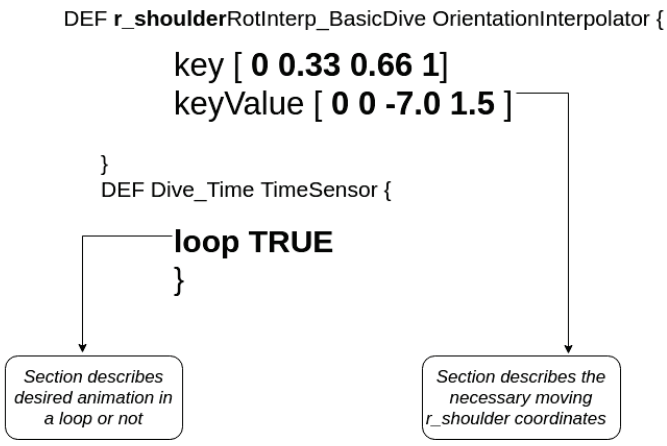


Figure 4.28: Animation section of a H-Anim file for *lift your right arm*

After the first and second steps, which are the semantic parser and the Bayesian network, the system creates the animation file. For example. when we try to generate an animation of an exercise instruction such as *lift your right arm* we need mainly *action*, *bodypart* and *destination location*. We only need the destination location because we assume the source location is same for all exercises, which is a neutral position. Using the semantic parser's results the system will extract *action* and *bodypart*, and the Bayesian network will extract the implicit destination *location* as shown in Figure 4.29 keeping the extracted *action* and *bodypart* as evidences in the *Action* and *Bodypart* variables

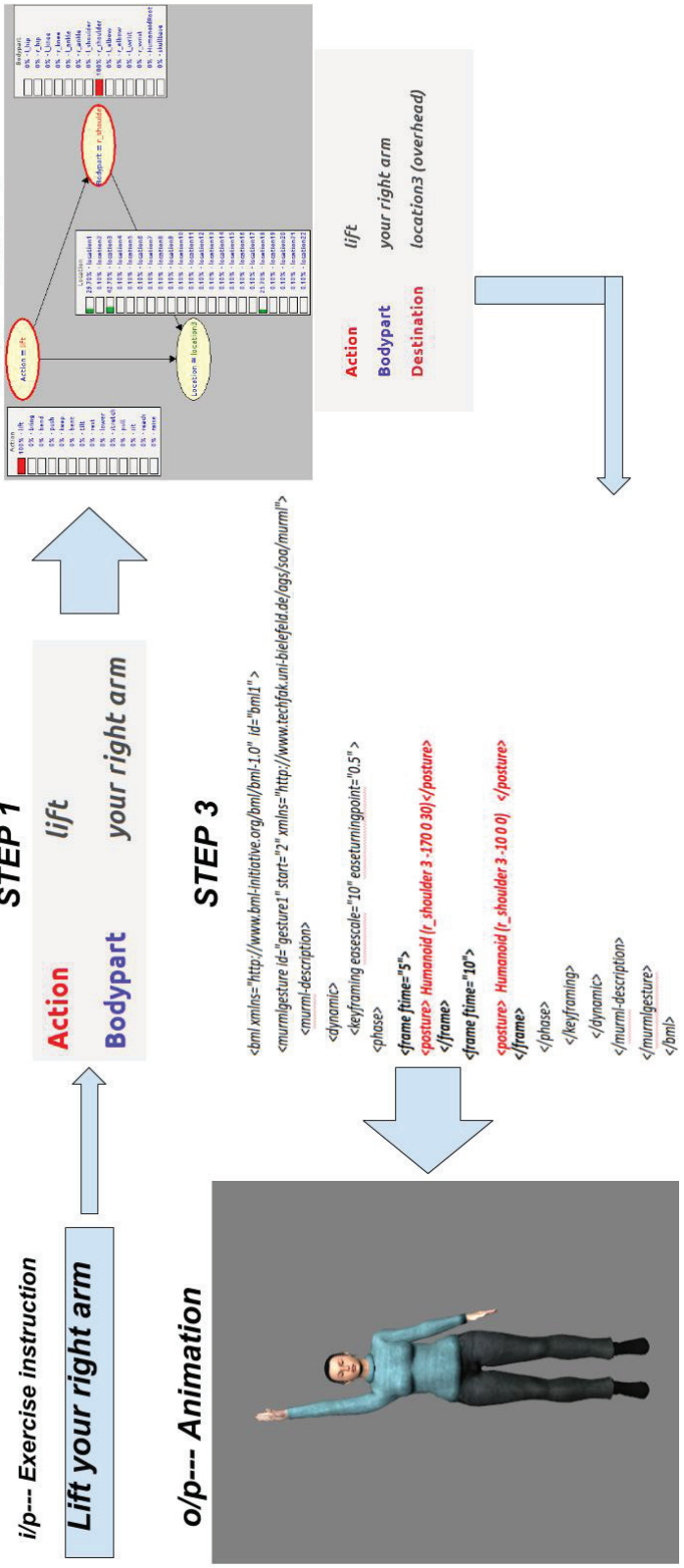


Figure 4.29: Result of text to animation system for exercise instruction *lift your right arm*

respectively. Then the system moves to the third and final step, which is to create the animation and generate the animation video for exercise instruction, such as *lift your right arm*. In Figure 4.29 we have shown the BML for creating the animation file, in which the system automatically generates an animation of the exercise instruction with the help of XML language BML, as shown in Figure 4.23. Step by step results for the whole system for the exercise instruction *lift your right arm* are shown in Figure 4.29.

Figure 4.30: Automated 3D animation for *lift your right arm*

The result of *lift your right arm* using H-Anim is shown in Figure 4.30; the format of the H-Anim animation file for the same is shown in Appendix D. For playing the output 3D animation we use the Instant Reality player[5]. The animation video which is created using BML, is played directly in the interface developed for BML using Java.

Our text to animation system for physical exercise is able to produce results for both single and multiple sentence exercise instructions consisting of single and multiple (currently two) actions or

[5] `http://www.instantreality.org`; Access date: 25 June 2018

motions or poses. The output animation of the exercise instruction is very satisfying regarding the movement of the specified body part and coordinates of the destination location. Step by step results showing how the system works as per our proposed framework for exercise instruction *lift your right arm* with the single sentence and pose as shown in Figure 4.29.

Also, an automatically generated animation file for ten different exercises as listed in Table 4.5 is shown in Appendix C.1, C.2, C.3, C.4 and C.5. We have shown only the animation file created using BML as H-Anim files are too long; therefore we have only shown one file for *lift your right arm* in Appendix D.

Chapter 5

Conclusion & Future Work

5.1 Summary

A text to animation system was developed where input is a text based physical exercise instruction and generate an animation video as an output. It is not an easy task for machines to analyze textual instructions. Thus, in Chapter 1, we have sought to analyze various problems or research issues relating to machine understanding of the natural language text. From the analysis, we found that there are many problems concerning the understanding of textual instructions for machines. In addition, the text-based exercise instruction was chosen for a case study because it gives a whole body movement. Hence, developing a text-to-animation system will be helpful in various aspects, such as imitation learning or demonstration training in robotics, health, exercise learning by people from home and so on. Chapter 2 describes the various currently available text-to-animation techniques and their limitations. It provides a brief discussion about various natural language understanding techniques such as syntactic parsers, semantic parsers and pragmatics, and attempted to identify the limitation on the natural language understanding using state-of-the-art applications. In order to overcome limitations on typical NLU techniques to analyze textual instructions, a Bayesian network was also introduced and analyzed. Finally, the animation techniques for generating the animation video from natural language text were described in this Chapter, describing a list of mark-up languages by which a user can create different types of animation. In Chapter 3, we have shown a case study for five different exercises

conducted by seven different people using typical text based exercise instruction sheets as an first step towards an automated pipeline that goes from text to virtual motions. Based on the crowdsourcing results, we found that assessing the quality of the workouts performed is not an easy task for human beings. Furthermore, the RGB-type (regular video) of visualization yields the most reliable ratings out of four different visualizations such as RGB, Depth, Skeleton, and with Virtual character, which might be expected, since it was the visualization modality that subjects were likely the most familiar with. The main prototype of text-to-animation is described in Chapter 4, consisting of three parts: the semantic parser, the Bayesian network, and the system to create animation. At the first stage, the design of two different types of semantic parser is described using the embodied construction grammar and Stanford's syntactic parser. The Bayesian network was described followed by a semantic parser used for extracting implicit information from textual instructions, which can not be extracted using the typical semantic parser. An animated video generation constitutes the final or the last phase of a text-to-animation system for which automatic animation files have been created using two different approaches for 2D and 3D animation. For both types of animation XML-based movement markup language was used, where for 2D animation behavior markup language(BML) and H-Anim was used for 3D animation videos. A human computation step was also included as the fourth and last step that validated the animation, which is basically the future step of our system.

In future work, we will aim to extend our system to handle multiple actions and sentence instructions. We will ultimately try to facilitate filling a broader array of types of potential implicit information. We are also working on a further prototype that is capable of extending the CPT with new variable levels if actions are encountered that are not yet supported by the network.

5.2 Conclusion

The main purpose of our thesis, as discussed in Chapter 1, is the development of a text-to-animation system for text based physical exercise instruction. In order to find a best visualization for the output of the text-to-animation system, we tried to find out the best visualizations using a human computation approach. In this study five exercises were recorded using Kinect and four different visualizations were developed, i.e. RGB, Depth, Skeleton and Virtual character. With these four different visualiza-

tions, a survey application designed in Unity game engine, aiming to crowdsource the assessment of the quality of exercise executions and to determine the best visualization modality for high inter-rater agreement. Following the quality assessment survey, a questionnaire was given to the participants to gather comparative responses on items expressing preferences regarding the visualization type, movement quality of different exercises. It has been found from both the survey and the questionnaire that RGB provides the best visualization, followed by Virtual, Skeleton and Depth visualization. In Chapter 2, currently available various text-to-animation and text-to-scene systems such as CONFUCIUS [Ma, 2006], Stanford's text to scene [Chang et al., 2014], CarSim [Åkerberg et al., 2003] and WordsEye [Coyne and Sproat, 2001] were discussed where the system generated animation videos or 3D images by inputing natural language text. The analysis shows all these system were designed based on declarative sentence and animation was generated from the content extracted from text using semantic analyzer. Thus, we can say that Natural language understanding is an important and primary task in developing a text-to-animation system. To advanced the current state-of-the-art, we are trying to develop a text-to-animation system for text based physical exercise instruction which are imperative sentence structure. After the analysis, we found that it takes primarily three parts like *Action, Bodypart and Location* to be extracted from textual instructions to generate an animation video from textual instructions. However, physical exercise instructions usually contain some implicit information, such as location for exercise instruction *lift your left arm* which is easily understandable for human. But, as shown in Chapter 2, these kinds of implicit information or pragmatics can not be extracted using typical semantic parser such as SEMAFOR [Chen et al., 2010], shallow semantic parsers [Pradhan et al., 2004], semantic role labelers [Björkelund et al., 2010], cognitive grammar [Langacker, 2008], radical construction grammar [Croft, 2001], Embodied construction grammar (ECG) [Bergen and Chang, 2005] and fluid construction grammar (FCG) [Steels, 2011] and so on. Furthermore, these semantic parsers or NLU tools do not give results as per our requirement as mentioned in Chapter 2. Hence, two different types of semantic parser was designed based on embodied construction grammar and Stanford's syntactic parser. These two parsers can extract all the information such as *Action, Bodypart and Location* as per per system's requirement if it present in the exercise instruction. For implicit information or pragmatics these parsers also failed, therefore, Bayesian network was included as the second step to the text-to-animation system to extract the pragmatics or implicit information from the natural language text. Two Bayesian networks have been designed to extract the implicit information from text-based physical exercise instruction sheets. These two networks are composed

respectively of three and four variables. These networks are used for single sentences with a single pose and multiple sentence instructions with two poses. The network consists of 50 unique values for variables *Action*, *Bodypart*, *Location*, and *Action2DestLoc*. In developing a strong and accurate Bayesian network, conditional probability table (CPT) plays the major role to predict the uncertainty of the domain. Therefore, a novel way of preparing the CPT of the Bayesian network is introduced, where human computation approach is used tp prepare the CPT with the help of crowds. For the same purpose, a survey application was designed consisting of 13 exercises that correspond to 13 human joints that are the values of the *Bodypart* variable. The participants rated 44 different videos on a scale of 1 to 5, with the possible destination location of those exercises. From the results, we found that it is easy to extract the most probable destination location from the network, even though that was unclear before the survey. The system also automatically updated the values of the *Action* variable using crowdsourcing, if the value was not an existing value of the *Action* variable. The animation video was generated using XML based MarkUp language from the natural language text after the necessary information was obtained from the semantic parsers and the Bayesian network. As mentioned in the Chapter 2, humanoid animation is required to generate the animation video for physical exercise instruction, as it has whole body movement. In order to create the animation two different types of MarkUp language were used, where BML was used for 2D animation and H-Anim for 3D animation. As mentioned, our text-to-animation system overcomes the present state of the art and produces 2D and 3D animation videos from imperative exercise instructions consisting of implicit information for single and multiple actions.

5.3 Future Work

In future work, we will endeavor to extend the set of variables. This will help machines to process more complex expressions. We will also focus on including further values of existing variables by automatically extending our Bayesian models. New categories will be introduced based on an expressed need being returned during the human computation step. This would also move the system further towards understanding more complex and multiple sentence instructions with the possibility of using movements of different body parts at the same time. Hence, human computation will be explored for making the Bayesian models more expressive where a user can provide feedback during or after the

creation of animation. According to the feedbacks, during the next animation generation, the system automatically updates and produces improved results. Knowledge of the action verb is also one important issues with respective to the the exercise instruction, which we have not explored right now. Hence, in future we will try to explore the same for better results. Discourse is also a very important part of natural language understanding, but this is very difficult for machines to understand. To design an animation system for multiple system instructions we will also need to focus on discourse in order to make the machines able to understand the instructions. Thus, in the future we will also try to explore discourse and add it to our model for multiple sentence instructions. The present state allows text-to-animation system, using BML and H-Anim humanoid formats, to create 2D and 3D animation videos. We will be trying to produce different formats in the future, including Wavefront (.obj), X3D Extensible 3D (.x3d), 3D Studo (.3ds), Filmbox (.fbx), Biovision Hierarchy (.bvh) and others. We are also going to try to give different characters rather than the same character perform exercise. Also, we can then use the results of our system and animations in different application areas, such as robotics. For the same, we will try to link the text-to-animation system with established robotic animation platforms that can generate a more accurate physical outcome generation, candidate validation, and for eventually transferring the outcomes to physical robots in real world situations. Using this approach, we can extend our advanced human computation-based text-to-animation scheme to the robotic training platform as well as attempt to instruct the same robot using natural language text.

References and Bibliography

P. Aguilera, A. Fernández, F. Reche, and R. Rumí. Hybrid bayesian network classifiers: application to species distribution models. *Environmental Modelling & Software*, 25(12):1630–1639, 2010.

O. Åkerberg, H. Svensson, B. Schulz, and P. Nugues. Carsim: an automatic 3d text-to-scene conversion system applied to road accident reports. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 2*, pages 191–194. Association for Computational Linguistics, 2003.

S. K. Andersen, K. G. Olesen, F. V. Jensen, and F. Jensen. HUGIN - A Shell for Building Bayesian Belief Universes for Expert Systems. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence.*, pages 1080–1085, Detroit, MI, USA, August 1989. Available from: `https://www.ijcai.org/Proceedings/89-2/Papers/037.pdf`.

Y. Arafa and A. Mamdani. Scripting embodied agents behaviour with cml: character markup language. In *Proceedings of the 8th international conference on Intelligent user interfaces*, pages 313–316. ACM, 2003.

C. F. Baker, C. J. Fillmore, and J. B. Lowe. The berkeley framenet project. In *Proceedings of the 17th international conference on Computational linguistics-Volume 1*, pages 86–90. Association for Computational Linguistics, 1998.

M. Beetz, U. Klank, I. Kresse, A. Maldonado, L. Mosenlechner, D. Pangercic, T. Ruhr, and M. Tenorth. Robotic roommates making pancakes. In *Humanoid Robots (Humanoids), 2011 11th IEEE-RAS International Conference on*, pages 529–536. IEEE, 2011.

I. A. Beinlich, H. J. Suermondt, R. M. Chavez, and G. F. Cooper. *The ALARM monitoring system: A case study with two probabilistic inference techniques for belief networks*. Springer, 1989.

B. Bergen and N. Chang. Embodied construction grammar in simulation-based language understanding. *Construction grammars: Cognitive grounding and theoretical extensions*, 3:147–190, 2005.

N. Birtles, N. Fenton, M. Neil, and E. Tranham. AgenaRisk manual Computer software, 2014. Available from: `http://www.agenarisk.com/resources/AgenaRisk_User_Manual.pdf`.

A. Björkelund, B. Bohnet, L. Hafdell, and P. Nugues. A high-performance syntactic and semantic dependency parser. In *Proceedings of the 23rd International Conference on Computational Linguistics: Demonstrations*, COLING '10, pages 33–36, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. URL `http://dl.acm.org/citation.cfm?id=1944284.1944293`.

R. Brooks and K. Cagle. The web services component model and humanml. *Proceedings of OASIS/HumanML technical comitee*, 2002.

J. Bruce, N. Sünderhauf, P. Mirowski, R. Hadsell, and M. Milford. One-shot reinforcement learning for robot navigation with interactive replay. *arXiv preprint arXiv:1711.10137*, 2017.

D. Brutzman. The virtual reality modeling language and java. *Communications of the ACM*, 41(6): 57–64, 1998.

E. Cambria and B. White. Jumping nlp curves: A review of natural language processing research. *IEEE Computational intelligence magazine*, 9(2):48–57, 2014.

J. Carletta. Assessing agreement on classification tasks: The kappa statistic. *Comput. Linguist.*, 22(2): 249–254, June 1996. ISSN 0891-2017. URL `http://dl.acm.org/citation.cfm?id=230386.230390`.

J. Cassell, H. H. Vilhjálmsson, and T. Bickmore. Beat: the behavior expression animation toolkit. In *Life-Like Characters*, pages 163–185. Springer, 2004.

H. Chan and A. Darwiche. When do numbers really matter? In *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*, pages 65–74, Seattle, Washington, August 2001.

Morgan Kaufmann Publishers Inc. Available from: https://www.jair.org/media/967/live-967-2041-jair.pdf.

A. Chang, W. Monroe, M. Savva, C. Potts, and C. D. Manning. Text to 3d scene generation with rich lexical grounding. *arXiv preprint arXiv:1505.06289*, 2015.

A. X. Chang, M. Savva, and C. D. Manning. Semantic parsing for text to 3d scene generation. *ACL 2014*, page 17, 2014.

N. Chang, J. Feldman, R. Porzel, and K. Sanders. Scaling cognitive linguistics: Formalisms for language understanding. In *Proc. 1st International Workshop on Scalable Natural Language Understanding*, 2002.

D. Chen, N. Schneider, D. Das, and N. A. Smith. Semafor: Frame argument resolution with log-linear models. In *Proceedings of the 5th international workshop on semantic evaluation*, pages 264–267. Association for Computational Linguistics, 2010.

S. H. Chen and C. A. Pollino. Good practice in bayesian network modelling. *Environmental Modelling & Software*, 37:134–145, 2012.

D. Chi, M. Costa, L. Zhao, and N. Badler. The emote model for effort and shape. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 173–182. ACM Press/Addison-Wesley Publishing Co., 2000.

L. Chrisman, M. Henrion, R. Morgan, et al. Analytica 4.2 User Guide. Lumina Decision System. *Inc., Los Gatos, CA, USA*, 2010. Available from: http://analyticaonline.com/ana/AdeUsersGuide4_2_3.pdf.

M. Cobo and H. Bieri. A web3d toolbox for creating h-anim compatible actors. In *Computer Animation, 2002. Proceedings of*, pages 120–125. IEEE, 2002.

S. Conrady and L. Jouffe. Introduction to Bayesian Networks & Bayesialab. *September*, 3(201): 3, 2013. Available from: http://library.bayesia.com/download/attachments/10092794/Bayesian_Networks_Intro_v16.pdf.

G. F. Cooper. Nestor: A Computer-Based Medical Diagnostic Aid That Integrates Causal and Probabilistic Knowledge. Technical report, DTIC Document, 1984. Available from: http://www.dtic.mil/get-tr-doc/pdf?AD=ADA152046.

B. Coyne and R. Sproat. Wordseye: an automatic text-to-scene conversion system. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 487–496. ACM, 2001.

F. G. Cozman. Javabayes-bayesian networks in java. 2001. Available from: http://www.cs.cmu.edu/~javabayes/.

W. Croft. *Radical construction grammar: Syntactic theory in typological perspective*. Oxford University Press on Demand, 2001.

A. Darwiche. Samiam. *Software available from http://reasoning. cs. ucla. edu/samiam*.

B. De Carolis, C. Pelachaud, I. Poggi, and M. Steedman. Apml, a markup language for believable behavior generation. In *Life-Like Characters*, pages 65–85. Springer, 2004.

M.-C. De Marneffe, B. MacCartney, C. D. Manning, et al. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC*, volume 6, pages 449–454, 2006.

M. J. Druzdzel. GeNIe: A development environment for graphical decision-analytic models. In *Proceedings of the AMIA Symposium*, page 1206, Washington, DC, USA, November 1999a. American Medical Informatics Association. Available from: www.pitt.edu/~druzdzel/psfiles/amia99.pdf.

M. J. Druzdzel. SMILE: Structural Modeling, Inference, and Learning Engine and GeNIe: A Development Environment for Graphical Decision-theoretic Models. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence and the Eleventh Innovative Applications of Artificial Intelligence Conference Innovative Applications of Artificial Intelligence*, AAAI '99/IAAI '99, pages 902–903, Orlando, Florida, USA, 1999b. American Association for Artificial Intelligence. ISBN 0-262-51106-1. Available from: http://www.aaai.org/Papers/AAAI/1999/AAAI99-129.pdf.

R. Elliott, J. Glauert, V. Jennings, and J. Kennaway. An overview of the sigml notation and sigmlsigning software system. In *Sign Language Processing Satellite Workshop of the Fourth International Conference on Language Resources and Evaluation, LREC 2004*, pages 98–104, 2004.

M. Eppe, S. Trott, and J. Feldman. Exploiting deep semantics and compositionality of natural language for human-robot-interaction. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 731–738. IEEE, 2016.

V. Evans and M. Green. *Cognitive linguistics: An introduction.* Lawrence Erlbaum Associates Publishers, 2006.

J. Feldman, E. Dodge, and J. Bryant. Embodied construction grammar. In *The Oxford handbook of linguistic analysis*. 2009.

M. Fried. *Construction Grammar*. I: A. Alexiadou & T. Kiss (red.), Handbook of syntax (2nd ed.) Berlin: de Gruyter, 2014.

N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine learning*, 29 (2-3):131–163, 1997.

C. Geiger, W. Mueller, and W. Rosenbach. Sam-an animated 3d programming language. In *Visual Languages, 1998. Proceedings. 1998 IEEE Symposium on*, pages 228–235. IEEE, 1998.

A. Goldberg. Construction grammar. *Concise encyclopedia of syntactic theories*, 6871, 1996.

T. Hanke. Hamnosys-representing sign language data in language resources and language processing contexts. In *LREC*, volume 4, 2004.

D. Heylen, S. Kopp, S. C. Marsella, C. Pelachaud, and H. Vilhjálmsson. The next step towards a function markup language. In *International Workshop on Intelligent Virtual Agents*, pages 270–280. Springer, 2008.

J. Hirschberg and C. D. Manning. Advances in natural language processing. *Science*, 349(6245): 261–266, 2015.

W. Hua, Z. Wang, H. Wang, K. Zheng, and X. Zhou. Short text understanding through lexical-semantic analysis. In *Data Engineering (ICDE), 2015 IEEE 31st International Conference on*, pages 495–506. IEEE, 2015.

Z. Huang, A. Eliëns, and C. Visser. Implementation of a scripting language for vrml/x3d-based embodied agents. In *Proceedings of the eighth international conference on 3D Web technology*, pages 91–100. ACM, 2003.

F. V. Jensen. *Introduction to Bayesian Networks*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1st edition, 1996. ISBN 0387915028.

F. V. Jensen. *Bayesian Networks and Decision Graphs*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2001. ISBN 0387952594.

Z. Ju, C. Yang, and H. Ma. Kinematics modeling and experimental verification of baxter robot. In *Control Conference (CCC), 2014 33rd Chinese*, pages 8518–8523. IEEE, 2014.

C. M. Kadie, D. Hovel, and E. Horvitz. MSBNx: A component-centric toolkit for modeling and inference with Bayesian networks. *Microsoft Research, Richmond, WA, Technical Report MSR-TR-2001-67*, 28, 2001. Available from: `https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/tr-2001-67.pdf`.

R. Kadlec, M. Schmid, O. Bajgar, and J. Kleindienst. Text understanding with the attention sum reader network. *arXiv preprint arXiv:1603.01547*, 2016.

W. KNIGHT. The dark secret at the heart of ai: no one really knows how the most advanced algorithms do what they do-that could be a problem, 2017.

S. Kopp, B. Krenn, S. Marsella, A. N. Marshall, C. Pelachaud, H. Pirker, K. R. Thórisson, and H. Vilhjálmsson. Towards a common framework for multimodal generation: The behavior markup language. In *Intelligent virtual agents*, pages 205–217. Springer, 2006.

S. Kopp, H. van Welbergen, R. Yaghoubzadeh, and H. Buschmeier. An architecture for fluid real-time conversational agents: integrating incremental output generation and input processing. *Journal on Multimodal User Interfaces*, 8(1):97–108, 2014.

A. Kranstedt, S. Kopp, and I. Wachsmuth. Murml: A multimodal utterance representation markup language for conversational agents. In *Proceedings of the AAMAS02 Workshop on Embodied Conversational Agents-let's specify and evaluate them*, 2002.

M. Krause and J. Smeddnick. Human computation–a new aspect of serious games. *Handbook of Research on Serious Games as Educational, Business and Research Tools: Development and Design*, 2011.

S. Kshirsagar, N. Magnenat-Thalmann, A. Guye-Vuillème, D. Thalmann, K. Kamyab, and E. Mamdani. Avatar markup language. In *ACM International Conference Proceeding Series*, volume 23, pages 169–177, 2002.

R. W. Langacker. *Cognitive grammar: A basic introduction*. OUP USA, 2008.

S. L. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 157–224, 1988.

P. Linell. *The written language bias in linguistics: Its nature, origins and transformations*. Routledge, 2004.

M. Ma. *Automatic conversion of natural language to 3D animation*. PhD thesis, University of Ulster, 2006.

N. Manual. Netica v1. 05. *Norsys Software Corp*, 1997. Available from: `https://www.norsys.com/downloads/old_versions/NeticaMan_Win_105.pdf`.

A. Marriott. Vhml–virtual human markup language. In *Talking Head Technology Workshop, at OzCHI Conference*, pages 252–264, 2001.

F. Michard, M. R. Pinsky, and J.-L. Vincent. Intensive care medicine in 2050: News for hemodynamic monitoring. *Intensive care medicine*, 43(3):440–442, 2017.

K. P. Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

S. S. Narayanan. *Knowledge-based Action Representations for Metaphor and Aspect RKARMAU*. PhD thesis, UNIVERSITY of CALIFORNIA at BERKELEY, 1997.

P. Norvig. Inference in text understanding. In *AAAI*, pages 561–565, 1987.

N. Okazaki, S. Aya, S. Saeyor, and M. Ishizuka. A multimodal presentation markup language mpml-vr for a 3d virtual space. In *Proceedings (CD-ROM) of Workshop on Virtual Conversational Characters: Applications, Methods, and Research Challenges (in conj. with HF2002 and OZCHI2002)*. Citeseer, 2002.

K. Perlin and A. Goldberg. Improv: A system for scripting interactive actors in virtual worlds. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 205–216. ACM, 1996.

V. Pomeroy, A. Pramanik, L. Sykes, J. Richards, and E. Hill. Agreement between physiotherapists on quality of movement rated via videotape. *Clinical rehabilitation*, 17(3):264–272, 2003.

R. Porzel. *Contextual computing: models and applications*. Springer Science & Business Media, 2010.

S. Pradhan, W. Ward, K. Hacioglu, and J. H. Martin. Shallow semantic parsing using support vector machines. 2004.

H. Prendinger, S. Descamps, and M. Ishizuka. Mpml: A markup language for controlling the behavior of life-like characters. *Journal of Visual Languages & Computing*, 15(2):183–203, 2004.

J. Richter. Brain rules (updated and expanded): 12 principles for surviving and thriving at work, home, and school. *International Sport Coaching Journal*, 2(1):83–84, 2015.

F. Ruggeri, R. S. Kenett, and F. W. Faltin. *Encyclopedia of statistics in quality and reliability*. Wiley Chichester, England, 2007.

A. Saffiotti and E. Umkehrer. PULCINELLA: A General Tool for Propagating Uncertainty in Valuation Networks. In *Proceedings of the Seventh conference on Uncertainty in Artificial Intelligence*, pages 323–331, Los Angeles, CA, USA, July 1991. Morgan Kaufmann Publishers Inc. Available from: `https://pdfs.semanticscholar.org/767a/a03075337bf74e79defb24f61ce4407044c9.pdf`.

H. Sarma, R. Porzel, J. Smeddnick, and R. Malaka. Towards generating virtual movement from textual instructions: A case study in quality assessment. In *Proceedings of The Third AAAI Conference on Human Computation and Crowdsourcing (HCOMP-2015)*. AAAI, 2015.

J. D. Smeddinck. Games for health. In *Entertainment Computing and Serious Games*, pages 212–264. Springer, 2016.

J. D. Smeddinck, J. Voges, M. Herrlich, and R. Malaka. Comparing modalities for kinesiatric exercise instruction. In *CHI'14 Extended Abstracts on Human Factors in Computing Systems*, pages 2377–2382. ACM, 2014.

L. Steels. *Design patterns in fluid construction grammar*, volume 11. John Benjamins Publishing, 2011.

A. Taylor, M. Marcus, and B. Santorini. The penn treebank: an overview. In *Treebanks*, pages 5–22. Springer, 2003.

M. Tenorth, G. Bartels, and M. Beetz. Knowledge-based specification of robot motions. In *ECAI*, pages 873–878, 2014.

J. A. Thomas. *Meaning in interaction: An introduction to pragmatics*. Routledge, 2014.

S. Uzor and L. Baillie. Exploring & designing tools to enhance falls rehabilitation in the home. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1233–1242. ACM, 2013.

S. Uzor and L. Baillie. Investigating the long-term use of exergames in the home with elderly fallers. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2813–2822. ACM, 2014.

Y. Wang, Y. Rao, and L. Wu. A review of sentiment semantic analysis technology and progress. In *Computational Intelligence and Security (CIS), 2017 13th International Conference on*, pages 452–455. IEEE, 2017.

R. V. Yampolskiy. Turing test as a defining feature of ai-completeness. In *Artificial intelligence, evolutionary computing and metaheuristics*, pages 3–17. Springer, 2013.

List of Publications

1. Sarma, H., Porzel, R., Smeddnick, J., and Malaka, R. (2015). Towards Generating Virtual Movement from Textual Instructions: A Case Study in Quality Assessment. In Proceedings of The Third AAAI Conference on Human Computation and Crowdsourcing (HCOMP-2015), San Diego, USA. AAAI.

2. Sarma, H. (2015). Virtual Movement from Textual Instructions. Doctoral Consortium of The Third AAAI Conference on Human Computation and Crowdsourcing (HCOMP-2015), San Diego, USA. AAAI.

3. Sarma, H. (2016). "From Textual Instructions to Virtual Actions–A Case Study." International Graduate School for Dynamics in Logistics: 59., University of Bremen, Germany

4. Sarma, H., Porzel, R., and Malaka, R. (2016). A Step Toward Automated Simulation in Industry. In Dynamics in Logistics: Proceedings of the 5th International Conference LDIC, 2016 Bremen, Germany, pages 99–105, Bremen, Germany.

5. Sarma, H., Porzel, R., Malaka, R., and Samaddar, A. B. (2017). A Step towards Textual Instructions to Virtual Actions. In 2017 IEEE 7th International Advance Computing Conference (IACC), pages 239–243, Hyderabad, India. IEEE.

6. Sarma, H., Samaddar, A. B. , Porzel, R., Smeddnick, J. and Malaka, R., and (2017). UPDATING BAYESIAN NETWORKS USING CROWDS. Neural Network World, Czech Technical University in Prague, Faculty of Transportation Sciences.

7. Sarma, H., Porzel, R., Smeddnick, J. and Malaka, R., and Samaddar, A. B. (2018). A Text to Animation system for Physical Exercises. , The Computer Journal, Oxford University Press.

Appendices

Appendix A

Questionnaire: Physical Exercise Recording Session

Demographic Questionnaire

1. What is your age?

Less than 15	15 - 25	25 - 35	35 - 45	45 - 55	Older than 55
☐	☐	☐	☐	☐	☐

2. What is your gender?

Male	Female
☐	☐

3. Do you feel fit to perform some slightly to moderately intensive exercises today ?

Yes	No
☐	☐

If, No

.....
.....
.....

4. Do you have any problems with specific movements at this time ?

Yes	No
☐	☐

If, Yes.... What and Why ?

.....
.....
.....

5. How many times you perform physical exercises ?

Daily	More than once in a Week	Once in a Week	Once in 2 Week	Once in a Month	Once in a Year	Never
1	2	3	4	5	6	7
☐	☐	☐	☐	☐	☐	☐

Figure A.1: Demographic Questionnaire

Questionnaire

Exercise 1

1. Did you find these exercise instructions to be easy or difficult to understand ?

Very Difficult	Moderately Difficult	Slightly Difficult	Neutral	Slightly Easy	Moderately Easy	Very Easy
1	2	3	4	5	6	7
☐	☐	☐	☐	☐	☐	☐

2. Was it (physically) comfortable to perform this exercise ?

Very Uncomfortable	Moderately Uncomfortable	Slightly Uncomfortable	Neutral	Slightly Comfortable	Moderately Comfortable	Very Comfortable
1	2	3	4	5	6	7
☐	☐	☐	☐	☐	☐	☐

3. Was it easy or difficult to perform this exercise ?

Very Difficult	Moderately Difficult	Slightly Difficult	Neutral	Slightly Easy	Moderately Easy	Very Easy
1	2	3	4	5	6	7
☐	☐	☐	☐	☐	☐	☐

4. Was it fun to perform this exercise ?

Very Boring	Moderately Boring	Slightly Boring	Neutral	Slightly Fun	Moderately Fun	Very Fun
1	2	3	4	5	6	7
☐	☐	☐	☐	☐	☐	☐

5. Do you want to give any suggestions regarding the instructions for this exercise ?

Figure A.2: Questionnaire asked after performing each exercise

OBSERVATIONS

DATE: _____ RESEARCHER: Himangshu Sarma

PARTICIPANT# : _____

Movement Start : _____

N = no feedback E1,E2,.....En = Exercise Number

1. PLAYER UTTERANCES

Please record when the player makes notable utterances about the his/her experience, or when completing the surveys.

2. PLAYER BEHAVIOR

Please record when the player shows distinct behaviors or has trouble when doing those exercises .

3. TECHNICAL NOTES

Please record any technical problems.

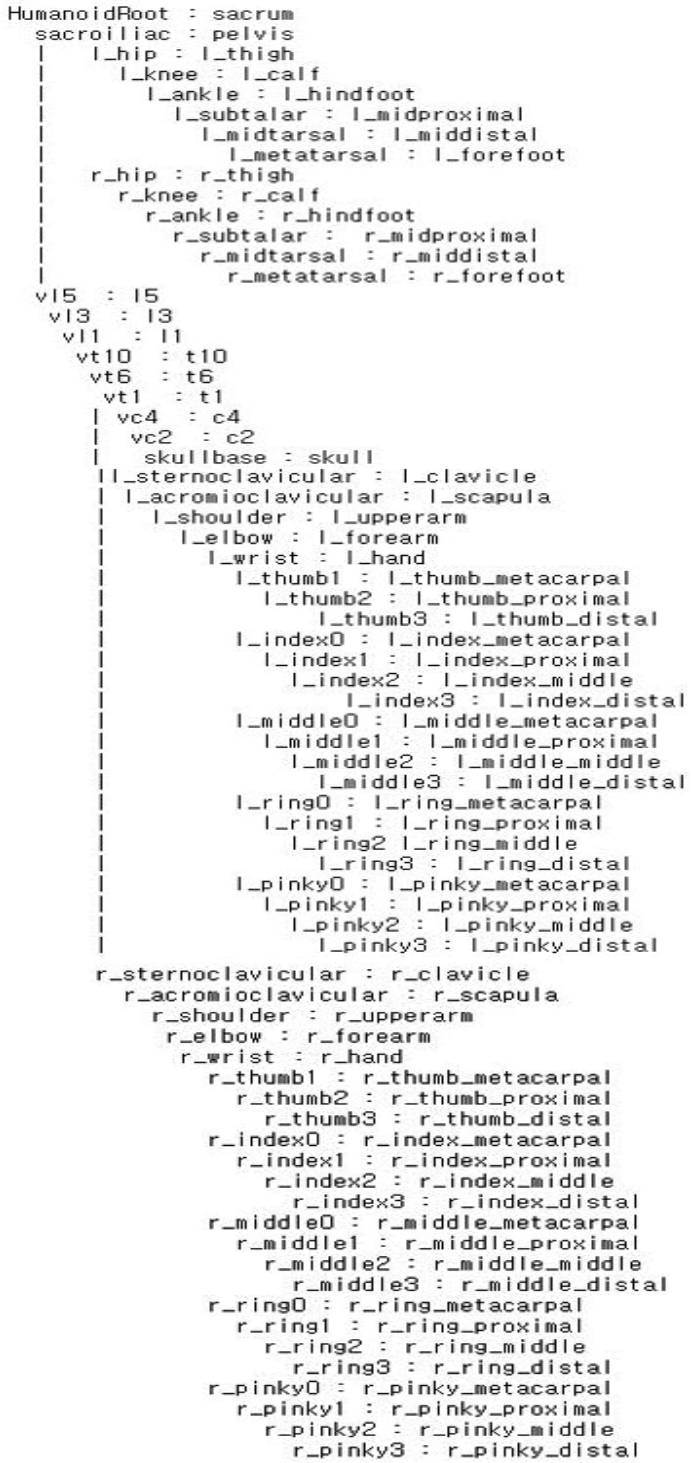
4. OTHERNOTES & INTERVIEW NOTES

Figure A.3: Observation during performance/recording of the exercises

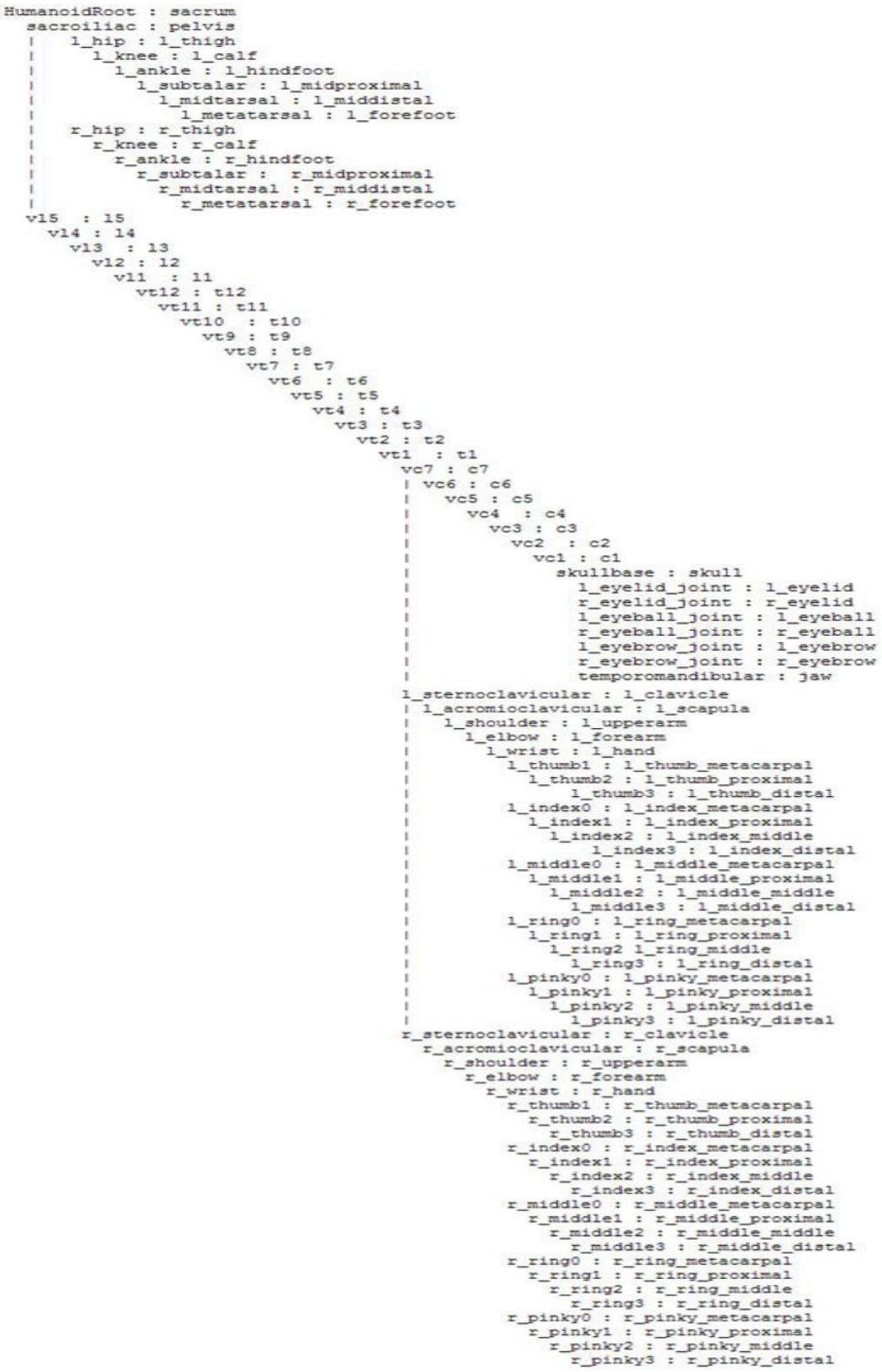
Appendix B

Hierarchy structure of H-Anim joints

B.1 Hierarchy structure of H-Anim For 71 Joints



B.2 Hierarchy structure of H-Anim For 94 Joints



Appendix C

Auto Generated Animation files using BML

Bend your left ankle

```
<bml xmlns="http://www.bml-initiative.org/bml/bml-1.0"  id="bml1" >
<murmlgesture id="gesture1" start="2"
xmlns="http://www.techfak.uni-bielefeld.de/ags/soa/murml">
<murml-description>
 <dynamic>
      <keyframing easescale="10" easeturningpoint="0.5" >
           <phase>
<frame ftime="3">
<posture>Humanoid (l_ankle 3 0 0 30)</posture>
</frame>
<frame ftime="6">
<posture>Humanoid (l_ankle 3 -10 0 0)</posture>
</frame>
</phase>
      </keyframing>
    </dynamic>
</murml-description>
</murmlgesture>
</bml>
```

Tilt your head

```
<bml xmlns="http://www.bml-initiative.org/bml/bml-1.0"  id="bml1" >
<murmlgesture id="gesture1" start="2"
xmlns="http://www.techfak.uni-bielefeld.de/ags/soa/murml">
<murml-description>
 <dynamic>
      <keyframing easescale="10" easeturningpoint="0.5" >
           <phase>
<frame ftime="3">
<posture>Humanoid (skullbase 3 20 0 0)</posture>
</frame>
<frame ftime="6">
<posture>Humanoid (skullbase 3 -20 0 0)</posture>
</frame>
<frame ftime="9">
<posture>Humanoid (skullbase 3 0 0 0)</posture>
</frame>
</phase>
      </keyframing>
    </dynamic>
</murml-description>
</murmlgesture>
</bml>
```

Figure C.1: Animation file for *Bend your left ankle* and *Tilt your head*

Stretch your leg

```
<bml xmlns="http://www.bml-initiative.org/bml/bml-1.0"  id="bml1" >
<murmlgesture id="gesture1" start="2"
xmlns="http://www.techfak.uni-bielefeld.de/ags/soa/murml">
<murml-description>
 <dynamic>
      <keyframing easescale="10" easeturningpoint="0.5" >
           <phase>
<frame ftime="3">
<posture>Humanoid (r_hip 3 20 -10 0)</posture>
</frame>
<frame ftime="6">
<posture>Humanoid (r_hip 3 0 0 0)</posture>
</frame>
</phase>
      </keyframing>
    </dynamic>
</murml-description>
</murmlgesture>
</bml>
```

Bend your right elbow

```
<bml xmlns="http://www.bml-initiative.org/bml/bml-1.0"  id="bml1" >
<murmlgesture id="gesture1" start="2"
xmlns="http://www.techfak.uni-bielefeld.de/ags/soa/murml">
<murml-description>
 <dynamic>
      <keyframing easescale="10" easeturningpoint="0.5" >
           <phase>
<frame ftime="3">
<posture>Humanoid (r_elbow 3 0 -150 0)</posture>
</frame>
<frame ftime="6">
<posture>Humanoid (r_elbow 3 0 0 0)</posture>
</frame>
</phase>
      </keyframing>
    </dynamic>
</murml-description>
</murmlgesture>
</bml>
```

Figure C.2: Animation file for *Stretch your leg* and *Bend your right elbow*

Raise your left shoulder

```
<bml xmlns="http://www.bml-initiative.org/bml/bml-1.0"  id="bml1" >
<murmlgesture id="gesture1" start="2"
xmlns="http://www.techfak.uni-bielefeld.de/ags/soa/murml">
<murml-description>
 <dynamic>
      <keyframing easescale="10" easeturningpoint="0.5" >
            <phase>
<frame ftime="3">
<posture>Humanoid (l_shoulder 3 -20 -130 0)</posture>
</frame>
<frame ftime="6">
<posture>Humanoid (l_shoulder 3 10 0 0)</posture>
</frame>
</phase>
       </keyframing>
    </dynamic>
</murml-description>
</murmlgesture>
</bml>
```

Lower your head

```
<bml xmlns="http://www.bml-initiative.org/bml/bml-1.0"  id="bml1" >
<murmlgesture id="gesture1" start="2"
xmlns="http://www.techfak.uni-bielefeld.de/ags/soa/murml">
<murml-description>
 <dynamic>
      <keyframing easescale="10" easeturningpoint="0.5" >
            <phase>
<frame ftime="3">
<posture>Humanoid (skullbase 3 0 30 0)</posture>
</frame>
<frame ftime="6">
<posture>Humanoid (skullbase 3 0 0 0)</posture>
</frame>
</phase>
       </keyframing>
    </dynamic>
</murml-description>
</murmlgesture>
</bml>
```

Figure C.3: Animation file for *Raise your left shoulder* and *Lower your head*

Bring your left arm toward your shoulder

```
<bml xmlns="http://www.bml-initiative.org/bml/bml-1.0"  id="bml1" >
<murmlgesture id="gesture1" start="2"
xmlns="http://www.techfak.uni-bielefeld.de/ags/soa/murml">
<murml-description>
 <dynamic>
      <keyframing easescale="10" easeturningpoint="0.5" >
           <phase>
<frame ftime="3">
<posture>Humanoid (l_shoulder 3 -40 -90 0)</posture>
</frame>
<frame ftime="6">
<posture>Humanoid (l_shoulder 3 10 0 0)</posture>
</frame>
</phase>
      </keyframing>
    </dynamic>
</murml-description>
</murmlgesture>
</bml>
```

Bring your right arm toward your shoulder

```
<bml xmlns="http://www.bml-initiative.org/bml/bml-1.0"  id="bml1" >
<murmlgesture id="gesture1" start="2"
xmlns="http://www.techfak.uni-bielefeld.de/ags/soa/murml">
<murml-description>
 <dynamic>
      <keyframing easescale="10" easeturningpoint="0.5" >
           <phase>
<frame ftime="3">
<posture>Humanoid (r_shoulder 3 40 -90 0)</posture>
</frame>
<frame ftime="6">
<posture>Humanoid (r_shoulder 3 -10 0 0)</posture>
</frame>
</phase>
      </keyframing>
    </dynamic>
</murml-description>
</murmlgesture>
</bml>
```

Figure C.4: Animation file for *Bring your left arm toward your shoulder* and *Bring your right arm toward your shoulder*

Push your right leg toward opposite

```
<bml xmlns="http://www.bml-initiative.org/bml/bml-1.0"  id="bml1" >
<murmlgesture id="gesture1" start="2"
xmlns="http://www.techfak.uni-bielefeld.de/ags/soa/murml">
<murml-description>
 <dynamic>
      <keyframing easescale="10" easeturningpoint="0.5" >
           <phase>
<frame ftime="3">
<posture>Humanoid (r_hip 3 40 -10 0)</posture>
</frame>
<frame ftime="6">
<posture>Humanoid (r_hip 3 0 0 0)</posture>
</frame>
</phase>
      </keyframing>
    </dynamic>
</murml-description>
</murmlgesture>
</bml>
```

Push your left leg toward opposite

```
<bml xmlns="http://www.bml-initiative.org/bml/bml-1.0"  id="bml1" >
<murmlgesture id="gesture1" start="2"
xmlns="http://www.techfak.uni-bielefeld.de/ags/soa/murml">
<murml-description>
 <dynamic>
      <keyframing easescale="10" easeturningpoint="0.5" >
           <phase>
<frame ftime="3">
<posture>Humanoid (l_hip 3 -40 -10 0)</posture>
</frame>
<frame ftime="6">
<posture>Humanoid (l_hip 3 0 0 0)</posture>
</frame>
</phase>
      </keyframing>
    </dynamic>
</murml-description>
</murmlgesture>
</bml>
```

Figure C.5: Animation file for *Push your right leg toward opposite* and *Push your left leg toward opposite*

Appendix D

H-Anim animation file: *"lift your right arm"*

```
# Animation start ========== ========== ==========

DEF Animations Group {
  children [
     DEF Dive_Animation Group {
       children [

DEF r_shoulderRotInterp_BasicDive OrientationInterpolator {
key [ 0.0 0.33 0.66 1.0 ]
keyValue [ 0.0 0.0 0.0 0.0 -1.2687 2.1525 -7.9261 2.201 0.0 0.0 0.0 0.0 ]
}



          DEF Dive_Time TimeSensor {
               cycleInterval 7.0
            loop TRUE
            startTime -1.0
          }
          DEF TriggerProximitySensor ProximitySensor {
            size 150 150 150
          }
      ]
     }
  ]
}




PROTO HAnimHumanoid [
   exposedField    SFString   name                ""
   exposedField    SFString   version             "2.0"
   exposedField    SFString   humanoidVersion      ""
   exposedField    MFString   info                [ ]
   exposedField    SFVec3f    translation          0 0 0
   exposedField    SFRotation rotation             0 0 1 0
   exposedField    SFVec3f    scale               1 1 1
   exposedField    SFRotation scaleOrientation     0 0 1 0
   exposedField    SFVec3f    center              0 0 0
   field           SFVec3f    bboxCenter          0 0 0
   field           SFVec3f    bboxSize            -1 -1 -1
   exposedField    MFNode     skeleton            [ ]
   exposedField    MFNode     skin                [ ]
   exposedField    MFNode     joints              [ ]
   exposedField    MFNode     segments            [ ]
   exposedField    MFNode     sites               [ ]
```

```
  exposedField    MFNode     viewpoints           [ ]
  exposedField    SFNode     skinCoord            NULL
  exposedField    SFNode     skinNormal           NULL
]
{
  Transform {
    translation      IS translation
    rotation         IS rotation
    scale            IS scale
    scaleOrientation IS scaleOrientation
    center           IS center
    bboxCenter       IS bboxCenter
    bboxSize         IS bboxSize
    children [
      Group {
        children IS skeleton
      }
      Group {
        children IS skin
      }
      Group {
        children IS viewpoints
      }
    ]
  }
}



PROTO HAnimJoint [
  exposedField    SFString     name               ""
  exposedField    MFFloat      ulimit             [ ]
  exposedField    MFFloat      llimit             [ ]
  exposedField    SFRotation   limitOrientation    0 0 1 0
  exposedField    MFInt32      skinCoordIndex     [ ]
  exposedField    MFFloat      skinCoordWeight    [ ]
  exposedField    MFFloat      stiffness          [ 0 0 0 ]
  exposedField    SFVec3f      translation        0 0 0
  exposedField    SFRotation   rotation           0 0 1 0
  exposedField    SFVec3f      scale              1 1 1
  exposedField    SFRotation   scaleOrientation    0 0 1 0
  exposedField    SFVec3f      center             0 0 0
  field           SFVec3f      bboxCenter         0 0 0
  field           SFVec3f      bboxSize           -1 -1 -1
  exposedField    MFNode       children           [ ]
  eventIn         MFNode       addChildren
  eventIn         MFNode       removeChildren
]
```

```
{
        Transform {
                translation      IS translation
                rotation         IS rotation
                scale            IS scale
                scaleOrientation IS scaleOrientation
                center           IS center
                bboxCenter       IS bboxCenter
                bboxSize         IS bboxSize
                children         IS children
                addChildren      IS addChildren
                removeChildren   IS removeChildren
  }
}



PROTO HAnimSegment [
  exposedField   SFString name            ""
  exposedField   SFFloat  mass            0
  exposedField   SFVec3f  centerOfMass    0 0 0
  exposedField   MFFloat  momentsOfInertia [ 0 0 0 0 0 0 0 0 0 ]
  field          SFVec3f  bboxCenter      0 0 0
  field          SFVec3f  bboxSize        -1 -1 -1
  exposedField   MFNode   children        [ ]
  eventIn        MFNode   addChildren
  eventIn        MFNode   removeChildren
  exposedField   SFNode   coord           NULL
  exposedField   MFNode   displacers      [ ]
]
{
        Group {
                bboxCenter     IS bboxCenter
                bboxSize       IS bboxSize
                children       IS children
                addChildren    IS addChildren
                removeChildren IS removeChildren
        }
}


# [Scene] ========== ========== ==========


# Background Color start ========== ========== ==========

Background {
  skyColor [ 0 1 1 ]
```

```
}

# Background Color end ========== ========== ==========



# Viewpoint position of the avatar start ========== ========== ==========

Viewpoint {
  centerOfRotation 0 1 0
  description "Sun"
  position 0 1 3
}

# Viewpoint position of the avatar end ========== ========== ==========


DEF hanim_Sun HAnimHumanoid {
  info [ "authorName=Himangshu Sarma" "authorEmail=himangshu.tezu@gmail.com" "creationDate=16
October 2017" "humanoidVersion=2.0" "gender=female" "height=1.5" ]
  name "himsExercise"
  scale 0.0325 0.0325 0.0325
  version "2.0"
  info [ ]


skeleton [
  DEF hanim_HumanoidRoot HAnimJoint {
    center 0 29.860001 -0.456700
    name "HumanoidRoot"
    children [

            Transform {
              translation 0 29.860001 -0.456700
              children [
                  Shape {
                    appearance Appearance {

                      texture DEF SunTextureAtlas ImageTexture {
                        url [ "Sun.png" ]
                      }
                    }
                    geometry IndexedFaceSet {

                      creaseAngle 1.57

                      coord Coordinate {
```

```
            }
            texCoord TextureCoordinate {

            }
          }
        }
      ]
    }

DEF hanim_l_hip HAnimJoint {
  center 2.955000 28.940001 -0.521800
  name "l_hip"
  children [

        Transform {
          translation 2.955000 28.940001 -0.521800
          children [
            Shape {
              appearance Appearance {
                material Material {
                  diffuseColor 0.588000 0.588000 0.588000
                }
                texture USE SunTextureAtlas
              }
              geometry IndexedFaceSet {

                creaseAngle 1.57

                coord Coordinate {

                }
                texCoord TextureCoordinate {

                }
              }
            }
          ]
        }

    DEF hanim_l_knee HAnimJoint {
      center 2.948000 16.459999 -0.527500
      name "l_knee"
      children [

            Transform {
              translation 2.948000 16.459999 -0.527500
              children [
                Shape {
```

```
            appearance Appearance {
              material Material {
                diffuseColor 0.588000 0.588000 0.588000
              }
              texture USE SunTextureAtlas
            }
            geometry IndexedFaceSet {

              creaseAngle 1.57

              coord Coordinate {

              }
              texCoord TextureCoordinate {

              }
            }
          }
        ]
      }

DEF hanim_l_ankle HAnimJoint {
  center 2.839000 3.899000 -0.411600
  name "l_ankle"
  children [

          Transform {
            translation 2.839000 3.899000 -0.411600
            children [
              Shape {
                appearance Appearance {
                  material Material {
                    diffuseColor 0.588000 0.588000 0.588000
                  }
                  texture USE SunTextureAtlas
                }
                geometry IndexedFaceSet {

                  creaseAngle 1.57

                  coord Coordinate {

                  }
                  texCoord TextureCoordinate {

                  }
                }
              }
```

```
                ]
               }

            DEF hanim_l_midtarsal HAnimJoint {
              center 2.839000 3.312000 1.078000
              name "l_midtarsal"
              children [

                    Transform {
                      translation 2.839000 3.312000 1.078000
                      children [
                        Shape {
                          appearance Appearance {
                            material Material {
                              diffuseColor 0.588000 0.588000 0.588000
                            }
                            texture USE SunTextureAtlas
                          }
                          geometry IndexedFaceSet {

                            creaseAngle 1.57

                            coord Coordinate {

                            }
                            texCoord TextureCoordinate {

                            }
                          }
                        }
                      ]
                    }

                ]
               }
            ]
           }
       ]
      }
  ]
}
DEF hanim_r_hip HAnimJoint {
  center -2.955000 28.940001 -0.521800
  name "r_hip"
  children [

          Transform {
            translation -2.955000 28.940001 -0.521800
```

```
      children [
        Shape {
          appearance Appearance {
            material Material {
              diffuseColor 0.588000 0.588000 0.588000
            }
            texture USE SunTextureAtlas
          }
          geometry IndexedFaceSet {

            creaseAngle 1.57

            coord Coordinate {

            }
            texCoord TextureCoordinate {

            }
          }

    }
  ]
}
DEF hanim_r_knee HAnimJoint {
  center -2.948000 16.459999 -0.527500
  name "r_knee"
  children [

        Transform {
          translation -2.948000 16.459999 -0.527500
          children [
            Shape {
              appearance Appearance {
                material Material {
                  diffuseColor 0.588000 0.588000 0.588000
                }
                texture USE SunTextureAtlas
              }
              geometry IndexedFaceSet {

                creaseAngle 1.57

                coord Coordinate {

                }
                texCoord TextureCoordinate {

                }
```

```
          }

    }
  ]
}
DEF hanim_r_ankle HAnimJoint {
  center -2.839000 3.899000 -0.411600
  name "r_ankle"
  children [

        Transform {
          translation -2.839000 3.899000 -0.411600
          children [
            Shape {
              appearance Appearance {
                material Material {
                  diffuseColor 0.588000 0.588000 0.588000
                }
                texture USE SunTextureAtlas
              }
              geometry IndexedFaceSet {

                creaseAngle 1.57

                coord Coordinate {

                }
                texCoord TextureCoordinate {

                }
              }

        }
      ]
    }
    DEF hanim_r_midtarsal HAnimJoint {
      center -2.839000 3.312000 1.078000
      name "r_midtarsal"
      children [

            Transform {
              translation -2.839000 3.312000 1.078000
              children [
                Shape {
                  appearance Appearance {
                    material Material {
                      diffuseColor 0.588000 0.588000 0.588000
                    }
```

```
                        texture USE SunTextureAtlas
                      }
                      geometry IndexedFaceSet {

                        creaseAngle 1.57

                        coord Coordinate {

                        }
                        texCoord TextureCoordinate {

                        }
                      }

                  }
                ]
              }
            ]
          }
        ]
      }
    ]
  }
}
DEF hanim_sacroiliac HAnimJoint {
  center 0 29.860001 -0.456700
  name "sacroiliac"
  children [

        Transform {
          translation 0 29.860001 -0.456700
          children [
            Shape {
              appearance Appearance {
                material Material {
                  diffuseColor 0.588000 0.588000 0.588000
                }
                texture USE SunTextureAtlas
              }
              geometry IndexedFaceSet {

                creaseAngle 1.57

                coord Coordinate {

                }
                texCoord TextureCoordinate {
```

```
            }
           }

     }
  ]
}
DEF hanim_l_shoulder HAnimJoint {
  center 6.077000 45.880001 -1.319000
  name "l_shoulder"
  children [
     DEF hanim_l_upperarm HAnimSegment {
       name "l_upperarm"
       children [
          Transform {
            translation 6.077000 45.880001 -1.319000
            children [
               Shape {
                 appearance Appearance {
                   material Material {
                     diffuseColor 0.588000 0.588000 0.588000
                   }
                   texture USE SunTextureAtlas
                 }
                 geometry IndexedFaceSet {

                   creaseAngle 1.57

                   coord Coordinate {

                   }
                   texCoord TextureCoordinate {

                   }
                 }
               }
            ]
          }
       ]
     }
     DEF hanim_l_elbow HAnimJoint {
       center 7.076000 38.529999 -1.385000
       name "l_elbow"
       children [

              Transform {
                translation 7.076000 38.529999 -1.385000
                children [
```

```
          Shape {
            appearance Appearance {
              material Material {
                diffuseColor 0.588000 0.588000 0.588000
              }
              texture USE SunTextureAtlas
            }
            geometry IndexedFaceSet {

              creaseAngle 1.57

              coord Coordinate {

              }
              texCoord TextureCoordinate {

              }
            }

      }
    ]
  }
  DEF hanim_l_wrist HAnimJoint {
    center 6.946000 30.889999 -1.308000
    name "l_wrist"
    children [

          Transform {
            translation 6.946000 30.889999 -1.308000
            children [
              Shape {
                appearance Appearance {
                  material Material {
                    diffuseColor 0.588000 0.588000 0.588000
                  }
                  texture USE SunTextureAtlas
                }
                geometry IndexedFaceSet {

                  creaseAngle 1.57

                  coord Coordinate {

                  }
                  texCoord TextureCoordinate {

                  }
                }
```

```
                }
              ]
            }
          ]
        }
      ]
    }
  ]
}
DEF hanim_r_shoulder HAnimJoint {
  center -6.077000 45.880001 -1.319000
  name "r_shoulder"
  children [

          Transform {
            translation -6.077000 45.880001 -1.319000
            children [
              Shape {
                appearance Appearance {
                  material Material {
                    diffuseColor 0.588000 0.588000 0.588000
                  }
                  texture USE SunTextureAtlas
                }
                geometry IndexedFaceSet {

                  creaseAngle 1.57

                  coord Coordinate {

                  }
                  texCoord TextureCoordinate {

                  }
                }

              }
            ]
          }
          DEF hanim_r_elbow HAnimJoint {
            center -7.076000 38.529999 -1.385000
            name "r_elbow"
            children [

                    Transform {
                      translation -7.076000 38.529999 -1.385000
                      children [
```

```
            Shape {
              appearance Appearance {
                material Material {
                  diffuseColor 0.588000 0.588000 0.588000
                }
                texture USE SunTextureAtlas
              }
              geometry IndexedFaceSet {

                creaseAngle 1.57


              }
              texCoord TextureCoordinate {

              }
            }

        }
      ]
    }
    DEF hanim_r_wrist HAnimJoint {
      center -6.946000 30.889999 -1.308000
      name "r_wrist"
      children [

            Transform {
              translation -6.946000 30.889999 -1.308000
              children [
                  Shape {
                    appearance Appearance {
                      material Material {
                        diffuseColor 0.588000 0.588000 0.588000
                      }
                      texture USE SunTextureAtlas
                    }
                    geometry IndexedFaceSet {

                      creaseAngle 1.57


                    }
                    texCoord TextureCoordinate {

                    }
                  }

            }
```

```
              ]
             }
           ]
          }
       ]
      }
  ]
}
DEF hanim_vl5 HAnimJoint {
  center 0.052490 34.130001 -0.468000
  name "vl5"
  children [

          Transform {
            translation 0.052490 34.130001 -0.468000
            children [
               Shape {
                 appearance Appearance {
                   material Material {
                     diffuseColor 0.588000 0.588000 0.588000
                   }
                   texture USE SunTextureAtlas
                 }
                 geometry IndexedFaceSet {

                   creaseAngle 1.57

                   coord Coordinate {

                   }
                   texCoord TextureCoordinate {

                   }
                 }

               }
       ]
      }
      DEF hanim_skullbase HAnimJoint {
        center 0 51.250000 -1.124000
        name "skullbase"
        children [

                Transform {
                  translation 0 51.250000 -1.124000
                  children [
                     Shape {
                       appearance Appearance {
```

```
                material Material {
                 diffuseColor 0.588000 0.588000 0.588000
                }
                texture USE SunTextureAtlas
               }
               geometry IndexedFaceSet {

                creaseAngle 1.57

                coord Coordinate {

                }
                texCoord TextureCoordinate {

                }
               }

             }
            ]
           }
          ]
         }
        ]
       }
      ]
     }
    ]
   }
]
}

ROUTE TriggerProximitySensor.enterTime TO Dive_Time.startTime
ROUTE Dive_Time.fraction_changed TO r_ankleRotInterp_BasicDive.set_fraction
ROUTE Dive_Time.fraction_changed TO r_kneeRotInterp_BasicDive.set_fraction
ROUTE Dive_Time.fraction_changed TO r_hipRotInterp_BasicDive.set_fraction
ROUTE Dive_Time.fraction_changed TO l_ankleRotInterp_BasicDive.set_fraction
ROUTE Dive_Time.fraction_changed TO l_kneeRotInterp_BasicDive.set_fraction
ROUTE Dive_Time.fraction_changed TO l_hipRotInterp_BasicDive.set_fraction
ROUTE Dive_Time.fraction_changed TO lower_bodyRotInterp_BasicDive.set_fraction
ROUTE Dive_Time.fraction_changed TO headRotInterp_BasicDive.set_fraction
ROUTE Dive_Time.fraction_changed TO neckRotInterp_BasicDive.set_fraction
ROUTE Dive_Time.fraction_changed TO upper_bodyRotInterp_BasicDive.set_fraction
ROUTE Dive_Time.fraction_changed TO whole_bodyRotInterp_BasicDive.set_fraction
ROUTE Dive_Time.fraction_changed TO whole_bodyTranInterp_BasicDive.set_fraction
ROUTE r_ankleRotInterp_BasicDive.value_changed TO hanim_r_ankle.set_rotation
ROUTE r_kneeRotInterp_BasicDive.value_changed TO hanim_r_knee.set_rotation
ROUTE r_hipRotInterp_BasicDive.value_changed TO hanim_r_hip.set_rotation
ROUTE l_ankleRotInterp_BasicDive.value_changed TO hanim_l_ankle.set_rotation
```

ROUTE l_kneeRotInterp_BasicDive.value_changed TO hanim_l_knee.set_rotation
ROUTE l_hipRotInterp_BasicDive.value_changed TO finWarpScript.set_rotationLeft
ROUTE l_hipRotInterp_BasicDive.value_changed TO finWarpScript.set_rotationRight
ROUTE l_hipRotInterp_BasicDive.value_changed TO hanim_l_hip.set_rotation
ROUTE lower_bodyRotInterp_BasicDive.value_changed TO hanim_sacroiliac.set_rotation
ROUTE headRotInterp_BasicDive.value_changed TO hanim_skullbase.set_rotation
ROUTE neckRotInterp_BasicDive.value_changed TO hanim_vc4.set_rotation
ROUTE upper_bodyRotInterp_BasicDive.value_changed TO hanim_vl1.set_rotation
ROUTE whole_bodyRotInterp_BasicDive.value_changed TO hanim_HumanoidRoot.set_rotation
ROUTE whole_bodyTranInterp_BasicDive.value_changed TO hanim_HumanoidRoot.set_translation
ROUTE Dive_Time.fraction_changed TO r_wristRotInterp_BasicDive.set_fraction
ROUTE Dive_Time.fraction_changed TO r_elbowRotInterp_BasicDive.set_fraction
ROUTE Dive_Time.fraction_changed TO r_shoulderRotInterp_BasicDive.set_fraction
ROUTE Dive_Time.fraction_changed TO l_wristRotInterp_BasicDive.set_fraction
ROUTE Dive_Time.fraction_changed TO l_elbowRotInterp_BasicDive.set_fraction
ROUTE Dive_Time.fraction_changed TO l_shoulderRotInterp_BasicDive.set_fraction
ROUTE r_wristRotInterp_BasicDive.value_changed TO hanim_r_wrist.set_rotation
ROUTE r_elbowRotInterp_BasicDive.value_changed TO hanim_r_elbow.set_rotation
ROUTE r_shoulderRotInterp_BasicDive.value_changed TO hanim_r_shoulder.set_rotation
ROUTE l_wristRotInterp_BasicDive.value_changed TO hanim_l_wrist.set_rotation
ROUTE l_elbowRotInterp_BasicDive.value_changed TO hanim_l_elbow.set_rotation
ROUTE l_shoulderRotInterp_BasicDive.value_changed TO hanim_l_shoulder.set_rotation