TECHNISCHE UNIVERSITÄT MÜNCHEN
Lehrstuhl für Echtzeitsysteme und Robotik

Comparing Classical and Embodied Multimodal Fusion for Human-Robot Interaction

Manuel Giuliani

Vollständiger Abdruck der von der Fakultät der Informatik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzender:	Univ. Prof. Dr. Darius Burschka
Prüfer der Dissertation:	1. Univ. Prof. Dr. Alois Knoll
	2. Univ. Prof. Dr. Jan de Ruiter (Universität Bielefeld)

Die Dissertation wurde am 17.01.2011 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 22.08.2011 angenommen.

Abstract

A robot that interacts with a human has to be able to interpret information from various input channels: it needs to understand and analyse the utterances by the human, it has to keep track of its own environment using sensors, and it needs to incorporate background knowledge about the task it was built for. Typically, a human-robot interaction system has various specialised system components that implement these abilities. Thus, the robot also needs to merge the information from its input channels so that it is able to complete its assigned task. This integration of information from input channels is called *multimodal fusion*.

This thesis presents two approaches for multimodal fusion for a robot that jointly cooperates with a human partner. The first approach, which is called *classical multimodal fusion*, focusses on processing human utterances. For that, the robot processes speech and gestures of its human partner using methods from classical artificial intelligence to yield logical representations of the utterances. Following that, these representations are enhanced with further information from other input modalities of the robot. In contrast to that, in the second approach the robot generates representations for its own actions in relation to objects in its environment, so-called *embodied multimodal fusion*. Here, the system uses the data from its input channels to evaluate the relevance of its own actions for a given context.

After a literature review, this thesis discusses the theoretical basis of both multimodal fusion approaches and presents how these methods can be implemented on a robot that is able to work together with a human on a common construction task, for which it processes multimodal input. These implementations were used in three human-robot interaction studies, in which naïve subjects worked together with the robot. The experiments were executed to study different aspects of joint action between human and robot.

The results of the experiments reveal several interesting facts: the first experiment studies how the robot can explain building plans to the human. The results of the

study show that the users preferred a plan explanation strategy in which the robot first names the target object and after that explains the single building steps. The first as well as the second experiment study the generation of referring expression in two different contexts. The results of the studies suggest that experiment participants rate the robot as a better dialogue partner when the robot makes full use of context information to generate referring expressions. Finally, the third experiment studies how humans perceive different roles of the robot in the interaction. The study shows that the users equally accept the robot as an instructor or as an equal partner and simply adjust their own behaviour to the robot's role.

Zusammenfassung

Ein Roboter, der mit einem Menschen interagieren soll, muss in der Lage sein Daten aus unterschiedlichen Eingabekanälen zu verarbeiten: Er muss die Äußerungen des Menschen verstehen und verarbeiten können, mit Sensoren seine Umgebung überwachen und er muss mit Kontextinformationen über die Aufgabe, für die er programmiert wurde, umgehen können. Üblicherweise werden diese unterschiedlichen Fähigkeiten in einem Mensch-Roboter-Interaktionssystem durch spezialisierte Einzelteile realisiert. Daher muss der Roboter auch in der Lage sein die Informationen aus seinen Eingabekanälen zu integrieren. Diese Integration von Informationen aus Eingabekanälen wird *multimodale Fusion* genannt.

In dieser Arbeit werden zwei Ansätze für multimodale Fusion für einen Roboter, der mit einem Menschen zusammenarbeitet, vorgestellt. Der erste Ansatz, die sogenannte *classical multimodal fusion*, ist auf die Verarbeitung von menschlichen Äußerungen fokussiert. Hier verarbeitet der Roboter die Sprache und Gesten seines menschlichen Partners mit klassischen Methoden der künstlichen Intelligenz um eine logische Repräsentation der Äußerungen zu erstellen. Anschließend wird diese Repräsentation mit Kontextinformationen von anderen Eingabemodalitäten des Roboter angereichert. Im Gegensatz dazu generiert der Roboter bei dem zweiten Ansatz, der sogennanten *embodied multimodal fusion*, Repräsentationen die seine eigenen Handlungen in Bezug zu Objekten in seiner Umgebung stellen. Die Informationen aus den Eingabekanälen des Roboters, zu denen auch die menschlichen Äußerungen gehören, verwendet der Roboter dazu, die Relevanz seiner eigenen Aktionen für einen gegebenen Kontext zu bewerten.

Nach einer Literaturrecherche werden in dieser Arbeit zunächst die theoretischen Grundlagen für die beiden vorgestellten Ansätze zur multimodalen Fusion diskutiert und eine Implementierung auf einem Roboter vorgestellt, der in der Lage ist mit einem Menschen zusammen an einer gemeinsamen Aufgabe zu arbeiten und dabei multimodale Eingaben verarbeitet und auch multimodale Äußerungen generiert. Die

vorgestellten Implementierungen werden dazu verwendet um drei Mensch-Roboter-Interaktionsexperimente durchzuführen, in denen unbefangene Versuchspersonen mit dem Roboter zusammenarbeiten. Diese Experimente dienen dazu verschiedene Aspekte der Zusammenarbeit zwischen Mensch und Roboter zu erforschen.

Die Experimente zeigen mehrere interessante Ergebnisse: Das erste Experiment zeigt, dass die Benutzer es bevorzugen, wenn der Roboter beim Erklären von Bauplänen zuerst das zu bauende Zielobjekt benennt und erst danach die einzelnen Bauschritte erklärt und nicht umgekehrt. Sowohl das erste als auch das zweite Experiment zeigen, dass die Menschen den Roboter als besseren Dialogpartner wahrnehmen, wenn dieser beim Benennen von Objekten in seiner Umgebung Ausdrücke verwendet, die Kontextinformation mit einbeziehen. Dies konnte in zwei verschiedenen Kontexten gezeigt werden. Das dritte Experiment zeigt, dass die Versuchspersonen keine klare Präferenz haben, welche Rolle der Roboter in der Interaktion einnimmt (sei es als Instrukteur oder als gleichberechtigter Partner), sondern einfach das eigene Verhalten an das des Roboters anpassen.

Acknowledgements

First of all, I would like to thank Professor Alois Knoll for giving my the opportunity to work at his lab and on the JAST project, and for supervising my thesis. I am also thankful to Professor Jan de Ruiter for being my external reviewer and for supporting me with advise and constructive criticism.

I want to express my gratitude to all the colleagues who worked with me in the JAST project: Mary Ellen Foster, Markus Rickert, Thomas Müller, Amy Isard, Colin Matheson, Jon Oberlander, Estela Bicho, Luís Luoro, and Nzoji Hipólito. This thesis would not have been realisable without the work of these colleagues on the JAST robot and on the experiments.

Many thanks go to my colleagues at the group for robotics and embedded systems. I had a great time working here and will always remember it well.

I want to express special thanks to Clemens Schefels, Claus Lenz, and Thorsten Röder for proof-reading my thesis and for their good comments that helped me improving it.

I want to thank my family for their continuous support, not only during this thesis, but also my whole life, which gave me the chance to come this far in my education. I especially want to thank my father, I had to think a lot about a particular advise he gave me when I started studying computational linguistics during the whole time of working on this thesis.

Finally, I deeply thank Steffi, who gave my advice for the layout of my thesis. But more importantly she gave me the love and emotional support that I needed to be able to finish this work.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The research area of *human-robot interaction* has the central goal to build robots that are able to communicate with humans. For that, a robot must have many abilities: it needs to recognise human utterances and the environment in which it is situated, it needs to have knowledge about the context of the interaction it is involved in, and it needs to be able to manipulate objects in its environment and to communicate to its human partner. Furthermore, the robot needs to integrate all of this information in order to successfully interact with a human, a process that is called *multimodal fusion*. The goal of multimodal fusion is to enable a robot to understand the information from various input modalities and to combine this information into an integrated representation so that the robot is able to perform its designated task.

For humans, multimodal fusion seems natural and easy to do, but consider the following example to see how much context knowledge and various abilities you need for multimodal fusion: you are sitting in a café. A woman at the table next to you points at your table and asks "Can I borrow this from you?". In this simple interaction you already need to be able to understand the language the other person is speaking and you need to see that she is using a pointing gesture to refer to an object on your table. Furthermore, you need to have context knowledge: for example, if there is a salt shaker and a sugar sprinkler on your table, you probably will have a look on the asker's table to see if she wants to drink a coffee or to eat soup. Thus, in multimodal fusion there are some challenges to master, which can be roughly separated into three categories:

- *Varying input data.* Multimodal fusion for human-robot interaction needs to handle data from very different input modalities. For example, the robot could have input channels with which it recognises human utterances, such as speech and gestures, but at the same

time it needs to handle information about its environment and about its assigned task, such as data from object recognition and task planning.

- *Unreliable input.* Typically, the data from input modalities is erroneous and unreliable. This is due to the fact that robots are situated in the real world, which is constantly changing. Thus, reliable signal processing for human-robot interaction is a complicated task that is hard to achieve.

- *The human factor.* Humans are very efficient when they interact with each other. The utterances they use to communicate are very fast and hard to detect. Also, in spoken language, people tend to use grammatically incorrect sentences or to leave out parts of sentences. However, humans are still able to successfully communicate their intentions.

Researchers in human-robot interaction attempt to solve these challenges with different approaches: on the one hand, they use methods from artificial intelligence, which use logical calculus or rule-based approaches to reason about facts. On the other hand, they use statistical methods that need data to train models, which can then be used to recognise unknown data. Both of these approaches have been successfully applied in other fields, for example in expert systems that use methods from artificial intelligence and in search engines that use statistical methods to index and search the internet. However, we argue that these approaches do not perform well in human-robot interaction, because methods from artificial intelligence need a well-defined environment to work with, which is not true in the dynamically changing real world, and statistical methods need huge amounts of annotated data, which is not only cumbersome to collect but there is also the problem that humans all react differently when they interact with a robot.

In the last years, there has been a new research direction in robotics, which is called embodiment (or sometimes "nouvelle artificial intelligence"). The central idea of embodiment is that an artificial agent that should show cognitive skills needs to have a physical body in the real world. Embodiment defines intelligent behaviour as a reasonable reaction by the agent to stimuli of its environment, which involves using sensors (eyes, ears, ...) and manipulators (arms, legs, ...) to explore the environment. Embodiment has been mainly applied successfully to sensorimotor coordination, for example for bipedal walking, where the clever use of the environment can be used to reach more stability. With these successes in mind, in this thesis we follow a central question: can we combine embodiment with methods from artificial intelligence to yield a more robust approach for multimodal fusion in human-robot interaction?

1.1 Thesis Goals

The main goal of this work is to develop a new approach for multimodal fusion that is tailored towards human-robot interaction. This approach should combine methods from artificial intelligence with ideas from embodiment to yield a multimodal fusion method that is robust enough to handle the uncertainties of the real world but still uses logical reasoning, for example to process speech or to plan system actions.

To realise this goal, we are using a robot that is able to work together with a human on a common task. On this robot, we first show an approach for multimodal fusion that is based on methods from artificial intelligence. After that, we develop the theoretical basis for a new multimodal fusion approach to show how data from low-level and high-level input channels can be integrated in a general way. We implement this new approach for multimodal fusion on the same robot as well so that we can compare the approaches to each other to analyse their strengths and weaknesses.

Concretely, the new approach for multimodal fusion should have the following properties:

- *Robustness.* The environment of a robot is marked by uncertainty: the input sensors by the robot can be disturbed or the recognition modules of the robot can report erroneous results. Additionally, unexpected events can occur in the robot's environment. The robot needs to have strategies so that it can complete its assigned task robustly, even if it encounters the problems mentioned above.

- *Integration of environment-related and task-related data.* A method for multimodal fusion for human-robot interaction does not only need to integrate the information from human utterances, for example speech and gestures. It also needs to be able to relate this information to the current state of the robot's environment and its assigned task. For this, the method needs to make use of context information as much as possible.

- *As fast as possible processing.* Humans will only accept robots if the robot is reacting fast to what they say or do. Most confusion in the interaction between human and robot arises when the robot is either not reacting at all or if it needs too much time to react on what the user says. Therefore, an approach for multimodal fusion for human-robot interaction has to work not only in theory, it must also be possible to implement it on a robot and to run with a reasonable speed.

- *Reusability.* Setting up a robot to work in a given context is a labour-intensive and time-consuming task. Therefore, it would be desirable to have robots that can work in different contexts without completely changing them.

1.2 Thesis Constraints

The field of multimodal fusion is broad and in this work we do not have unlimited time. Therefore, we have to make some constraints that define, how far we can go in this thesis:

- *Collaborative robot.* We develop approaches for multimodal fusion for a robot that works together with a human. That means that the robot has pre-installed knowledge about the common task it should achieve together with the human and it needs to be able to talk about the task and to understand utterances by the human that relate to the common goal. Human and robot will not have a free interaction about arbitrarily chosen topics.
- *Goal-oriented robot.* Our approaches for multimodal fusion should enable a robot to reach defined goals. In our opinion, any intelligent agents needs to have a goal in order to show a meaningful behaviour. That means that we do not develop methods that show how multimodal fusion can be done in general, but we produce multimodal fusion approaches with keeping in mind that they should work on a goal-oriented robot.
- *Joint-Action Scenario.* The main scenario for the thesis will be the scenario by the JAST project, which mainly funded this work. The scenario is described in the background section in Chapter 2.
- *Input processing, not input recognition.* The work in this thesis is oriented towards enabling a robot to process the utterances by a human and to integrate them with information from the robot's environment, but we will not develop new methods for input recognition, for example speech or gesture recognition.
- *Scenario language is English.* We will focus on English language processing because of the international direction of the project from which this thesis was funded.

1.3 Thesis Structure

Chapter 2 introduces the JAST project from which we use the robot to show implementations of our multimodal fusion approaches. Furthermore, this chapter contains references to related work

from multimodal dialogue systems, cognitive architectures and robot architectures, human-robot interaction, spoken language processing, and embodiment, which influenced our work.

Chapter 3 describes the theoretical framework of the two approaches for multimodal fusion that we developed for this thesis. The first approach uses methods from artificial intelligence and is focussed on integrating human utterances, including speech and gestures. The second approach combines ideas from artificial intelligence and embodiment. This approach is centred around the idea that the robot as a cognitive agent should evaluate its own actions at any given moment in an interaction to select the action it should execute next.

Chapter 4 gives an overview of the implementation of the two approaches for multimodal fusion on the JAST robot. The chapter describes the software architectures of the two implementations and shows processing examples for an interaction between a human and the JAST robot, which clarify how the different multimodal fusion approaches handle the data of the robot's input channels.

Chapter 5 presents three evaluation studies in which we used our two multimodal fusion approaches. The results of these experiments show how both methods can be applied to realise a successful human-robot interaction between naïve subjects and the JAST robot. Furthermore, we use these studies to research various aspects of joint action.

Finally, Chapter 6 concludes this thesis with a list of the main contributions of this thesis. It also gives an outlook on future research directions of the new multimodal fusion approach.

1.4 Terms and Abbreviations

Throughout this thesis we will use a set of terms, which are explained in this section. The abbreviations that we used in the text are listed in Table 1.1.

- *Input channels* and *modalities*. We call each source that provides information about the robot's environment *input channel* or *modality*. An input channel or modality can be a recognition module for human utterances (e.g. speech or gesture recognition), a component for image processing (e.g. object recognition), or a module that provides information about the robot's task (e.g. task planner or goal inference).

- *Objects*, *target objects*, and *pieces*. For the evaluation of our approaches we are using a scenario in which a human and a robot together build objects from a wooden toy construction set. Hence, we have to clarify the meaning of several terms that relate to

objects as concepts and objects in the case of the scenario. We use the term *object* when we talk about objects as entities in a general way. The robot we are using to demonstrate our implementation is able to work together with a human and to build assemblies from a wooden toy construction set. We use the term *target object* when we talk about these assemblies, and we use the term *piece* to talk about the single parts of the toy construction set.

AI	Artificial Intelligence
CCG	Combinatory Categorial Grammar
CMF	Classical Multimodal Fusion
EMF	Embodied Multimodal Fusion
GOFAI	Good Old-Fashioned Artificial Intelligence
HCI	Human-Computer Interaction
HRI	Human-Robot Interaction
Ice	Internet Communication Engine
JAST	Joint-Action Science and Technology
MF	Multimodal Fusion
OAC	Object Action Complex

Table 1.1: Abbreviations used in this thesis.

Chapter 2

Background and Related Work

In this chapter, we set the background for our work. In Section 2.1, we introduce the JAST project, in which most of this work was developed and implemented. After that, In Section 2.2 we review literature from various research directions that influenced our work.

2.1 The JAST project

JAST[1] was a European project that was funded in the FP6 call for cognitive systems. The acronym JAST stands for *Joint-Action Science and Technology*. The main goal of JAST was to develop inanimate cognitive agents that are able to collaborate with a human on a common task. For that, JAST united researchers from the fields of cognitive psychology, computational linguistics, neurology, psycholinguistics, and robotics.

2.1.1 JAST Robot

In Munich at the Technische Universität München, a robot was built on which the research results of JAST were implemented. Figure 2.1 shows the JAST robot, which has a pair of manipulator arms with grippers, mounted in a position to resemble human arms, and an animatronic talking head [78] capable of producing facial expressions, rigid head motion, and lip-synchronised synthesised speech. The robot can recognise and manipulate pieces of a wooden toy construction set called Baufix, which are placed on a table in front of the robot. A human and the robot work together to assemble target objects from Baufix pieces, coordinating their actions through speech (English or German), gestures, and facial expressions.

[1] For videos and pictures of JAST please refer to `http://www6.in.tum.de/Main/ResearchJast`

Figure 2.1: The JAST robot.

2.1.2 JAST Robot Architecture

The architecture of the JAST robot is composed of a set of modules that implement its abilities. The communication between the system parts is realised with the Internet Communication Engine [45]. Figure 2.2 shows the last version of the JAST architecture that was used for the final system evaluation [36]. The architecture is separated into four layers: the *input* layer holds the input channels of the robot, modules in the *interpretation* layer take the information from the input modules and process them so that the system components of the *reasoning* layer can plan robot actions, which are then realised by the modules in the *output* layer. In the following, we will give a short overview of the JAST robot and cite publications for system parts when they are available. In Chapter 3 we will introduce system parts in more detail were it is needed.

The JAST robot observes its environment by using input channels for *speech recognition*, *object recognition* [62, 61], *gesture recognition* [89, 90], and *head tracking*. The *broker* collects the information from these channels centrally and distributes it to other parts of the robot. For

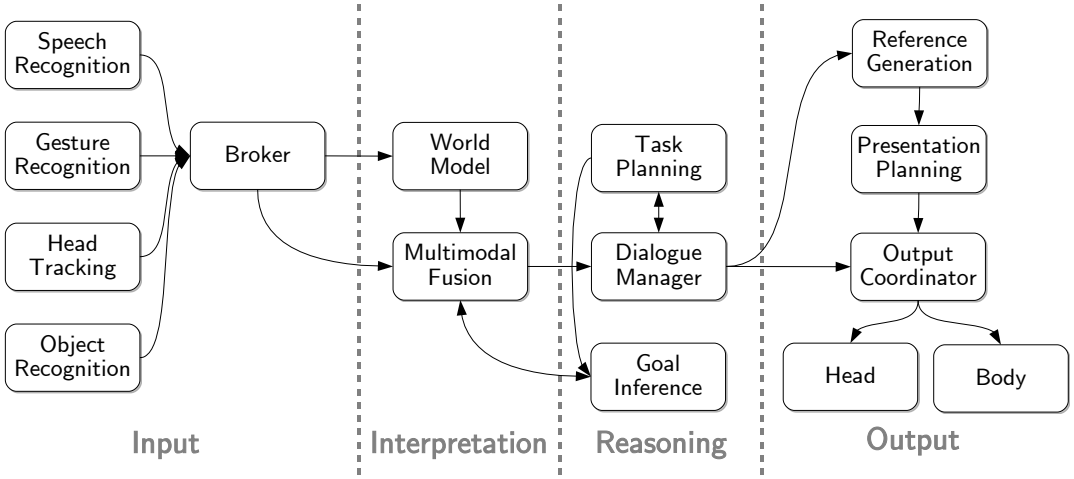


Figure 2.2: JAST robot architecture.

example, the *world model* gets information from object recognition, keeps track of objects in the robot's environment, and provides interfaces to query this information. The second module that gets information from the broker is *multimodal fusion* (MF)—the module that was developed in this thesis. The task of multimodal fusion is to integrate the information from speech and gesture recognition with information from the world model to build an integrated representation for further processing. In the last version of the JAST robot architecture, multimodal fusion was also used as a communicator between *goal inference* and *dialogue manager*. This architecture layout was chosen because goal inference takes information from similar input channels as multimodal fusion and computes the next robot actions based on this information. However, the dialogue manager needs representations of human utterances, which are generated by multimodal fusion. Hence, dialogue manager and goal inference have no direct connection, but multimodal fusion combines the information from the input channels with computations from goal inference and sends it to dialogue manager. Goal inference is a module which is based on the dynamic neural fields approach [30, 11] and its task was to select the robot's actions based on the actions by its human partner. For that, goal inference used information from speech, gesture and object recognition and a *task planning* component that contains building plans and keeps track of the current status of a chosen plan. Multimodal fusion combines information from the input channels and goal inference and sends it to the *dialogue manager*, which implements the information-state based approach to dialogue planning [55]. The dialogue manager controls the interaction between human and robot and calculates the next robot movements and dialogue acts. For that, on the one hand the dialogue manager uses *reference generation*

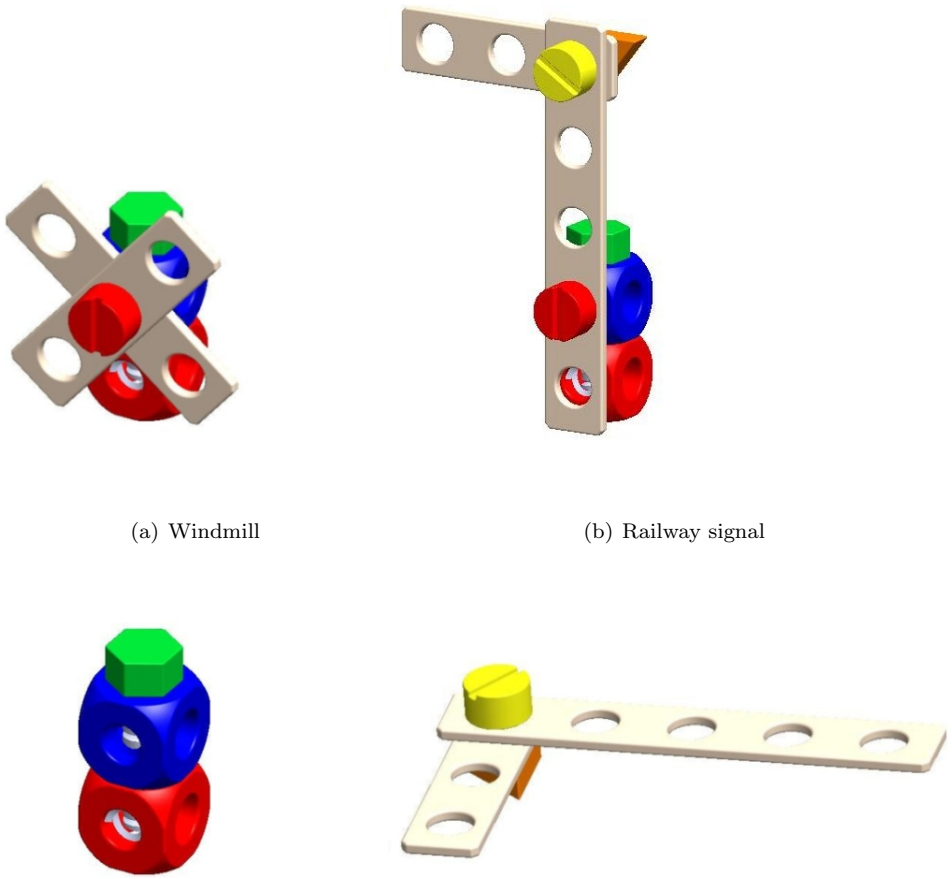
[41] to compute the referring expressions the robot should use when it talks to the human; these expressions are embedded into sentences in the *presentation planning* module. On the other hand, the dialogue manager sends instructions to the *output coordinator* that synchronises the robots actions with the sentences the robot should say and sends instructions to the robot's *head* [32] and *body* separately. In literature, this generation of multimodal output is called *fission*.

2.1.3 JAST Construction Task

For the evaluations of the JAST robot system, a task was defined in which a human and the robot jointly construct target objects from Baufix pieces. Baufix pieces can either be cubes, bolts, nuts, or slats. Cubes and bolts can be blue, green, red, or yellow, nuts are always orange, and bolts and slats come in the three sizes small, medium, and large. Two target objects were defined for the joint construction task, which are shown in Figure 2.3: a *windmill* (Figure 2.3(a)) and a *railway signal* (Figure 2.3(b)). The base of both target objects is the *tower* (Figure 2.3(c)) that is combined with two slats and a red bolt to form the windmill or with the so-called *l shape* (Figure 2.3(d)) to form the railway signal.

During construction, the workspace in front of the robot was divided into a working area for the human and a working area for the robot, from which only human or robot were allowed to pick up Baufix pieces, respectively. The JAST robot is not able to build windmill or railway signal by itself, but it can hand over pieces from its workspace to the human. For the JAST construction task, two initial table layouts exist that define which Baufix pieces have to be placed in the human's and robot's workspaces at the beginning of the interaction. The table layouts make sure that there are enough Baufix pieces that are similar to each other on both sides of the table so that human and robot have to talk to each other about the pieces. Furthermore, the layouts guarantee that there are Baufix pieces on the robot's side of the table, which are needed for the building plan, so that the robot has to hand over pieces to the human. We show the table layouts in Appendix A.2.

For the three user evaluations that we present in this thesis, the JAST robot adopted different roles in the interaction: in the first evaluation [34, 35], only the robot knew the building plan of the two target objects. Thus, it had to instruct the human how to build the windmill and the railway signal. In the second evaluation [41], human and robot both knew the building plans. In this study, the robot was able to detect errors that were made by the human and it was able to explain the error to the human and how to solve the problem. In the



(a) Windmill (b) Railway signal

(c) Tower (d) L shape

Figure 2.3: Target objects of JAST construction task.

third evaluation, we compared two different robot behaviours: a proactive behaviour, in which the robot assisted the human and preferably handed over Baufix pieces to the human, and an instructive behaviour, in which the robot gave instructions to the user first and then handed over pieces from its workspace to the human. For more details on these evaluations see Chapter 5.

2.2 Related Work

In this section, we review related work from diverse research areas. We are using the modularity from *multimodal systems* (Section 2.2.1) for an approach for classical multimodal fusion that we present in Section 3. From *cognitive architectures* and *robot architectures*, we take ideas about

the setup of a cognitive robot, and get influences about data processing inside the system (Section 2.2.2). *Human-robot interaction* (HRI) is based on multimodal systems and cognitive architectures; thus, we review some of the influential HRI systems (Section 2.2.3). Finally, we review approaches for *spoken language processing* (Section 2.2.4) and explain some of the principles of *embodiment* which we incorporated in our work (Section 2.2.5).

2.2.1 Multimodal Dialogue Systems

Much of the work that is done now in human-robot interaction is based on work for multimodal dialogue systems. The first system that is considered as a multimodal system was reported by Bolt in the article "Put-That-There" [12]. The described system was the MIT media room, which allowed a user to draw basic shapes, for example cubes and triangles, on a screen by giving commands such as "put a triangle there" and pointing on the screen at the same time. The system was able to recognise these commands and drew the corresponding shapes on the screen. Speech and gesture recognition was basic at that time due to the constraints in computing power. Bolt saw the application domain for this technology mainly in organising ships in harbours or for military purposes.

"Put-That-There" was an example for a system that uses the so-called *late fusion* or *semantic fusion* to integrate several modalities. These systems classify the input from several channels with separate recognition modules and fuse the interpretations of the modules in a central interpretation module. Some newer examples of multimodal dialogue systems that use late fusion are SmartKom [80] and COMIC [13], and the classical MF approach we show in this thesis also implements the late fusion approach. In contrast to the late fusion approach, multimodal systems that use *early fusion*, which is also called *feature level fusion*, mostly integrate input channels that are closely bound to each other, for example speech and lip movements or speech and pointing gestures. These modalities can be fused by combining features from both modalities into one feature vector. This vector is used for training of statistical models, which are then used for recognition of multimodal events. Examples for multimodal systems that use early fusion are Quickset [26] and MATCH [47]. Quickset fused speech recognition and pen input on a touch screen and was mainly used for military applications. For example, in Quickset the user drew a circle on a map that was presented on a touch screen and said "this is a hot spot zone" at the same time [87]. The system marked the corresponding area as a dangerous zone afterwards. Quickset was developed at the Center for Human-Computer Communication by several researchers including Sharon Oviatt, whose often cited article "Ten

Myths of Multimodal Interaction" [64] reveals common misinterpretations of researchers in the area of multimodal fusion.

The list of multimodal dialogue systems that were developed since "Put-That-There" is of course much longer than the few systems we review here. For a more extensive list of multimodal dialogue systems please refer to Sharma et al. [74], who discuss in their article why multimodal systems are needed in human-computer interaction. They also provide an overview over different input modalities, several fusion techniques, and applications for multimodal systems. Oviatt provides a similar review of multimodal system in the article "*Multimodal Interfaces*" [65] that focuses on the development and status of existing multimodal systems. Another excellent review can be found in the dissertation by Pfleger [69].

There are a few properties of multimodal systems that we want to mention because they influenced the ideas of this thesis: in these systems, only human utterances are regarded as input modalities. Therefore, they are driven by the input of the human user, which means that they are only reacting to what the human says or does. They do not include information from other channels, which is probably due to the facts that computing power was low and that the first multimodal systems were developed for non-embodied agents. Also, many of these systems have in common that they are developed for a certain domain and can only be ported to applications with changing substantial parts of the system. There are some exceptions, for example Johnston et al. [47] present a framework for rapid prototyping of information systems with speech and pen input and the researchers of the Talk project [58] describe a grammatical framework for development of multimodal grammars. The work from Landragin et al. [54] is also very interesting, as they are showing how they port MMIL, the MultiModal Interface Language that is used for multimodal meaning representation, to a new domain.

2.2.2 Cognitive Architectures and Robot Architectures

Many concepts in our work are influenced by cognitive architectures and robot architectures, which are related to each other but still differ in certain parts.

Cognitive architectures are templates for intelligent agents. They define processes that act like intelligent systems or simulate certain aspects of intelligent systems. They were developed by cognitive psychologists to implement models of human cognition. In the following, we will review ACT-R as an example for such a cognitive architecture as it was originally intended. ACT-R (adaptive control of thought-rational) [4] is one of the best-known cognitive architectures. Researchers working on ACT-R strive to understand how humans organise knowledge

and produce intelligent behaviour. Scientists can model aspects of human cognition with the ACT-R framework. These aspects deal with very different tasks, for example memorisation of texts, recognition of speech, or human communication. The models that evolve from the framework are affected by the assumptions that the modeller has about human cognition. Classical tests in cognitive psychology can prove if these assumptions were right. The system architecture of ACT-R consists of several modules that process different kinds of information. For example, the architecture contains modules that simulate the visual perception or the hand-eye coordination of humans. The central component of the ACT-R architecture is the so-called *production module* that contains a representation of the processes in the human brain. Furthermore, ACT-R has modules such as the *declarative module*, which simulates how information can be obtained from memory, or the *goal module*, that keeps track about the progress of the current task. Our work is influenced by ACT-R, because it shows that for a cognitive process, not only the information from the visual and audible channels of cognitive agents are important, they also need to incorporate their plans and goals.

To get a wider overview of the research about cognitive architectures, we refer to two publications that provide excellent surveys of the field. Byrne gives in [22] a description for cognitives architectures in general and reviews the usefulness of cognitive architectures in human-computer interaction (HCI). He argues that cognitive architectures can be used as design and evaluation aid for HCI systems. Additionally, they can be used for training purposes because they copy human behaviour and therefore are well-suited for the replacement of expensive human training partners. Vernon et al. [79] give an overview of cognitive architectures and how they are used to implement mental capabilities in computational agents. They start with a description of the different paradigms that are used for the systems: the *cognitivist* approach and the *emergent* approach. After that, they review cognitive systems that use either one of the two approaches or try to implement a hybrid architecture that involves both of the approaches. The authors also list the key features they believe an autonomous agent needs to exhibit: a reflection of the brain regions and their connections in the architecture, perceptual categorisation, embodiment, a minimal set of innate behaviours, and adaptive behaviour.

Robot architectures are related to cognitive architectures, but instead of modelling human mental processes robot architectures are used to control the actions of a robot. For that, the robot has to reason about the input information it gets from its sensors so that it can infer which action it should execute next, similar to the perception modules that ACT-R uses in its architecture.

An interesting theory was proposed by Wyatt and Hawes [88]. They argue that the recent advances in building cognitive architectures for robots can explain aspects of cognition in nature. Wyatt and Hawes present an architecture for cognition, which is based on multiple shared workspaces that are used to group processing of information in a cognitive system. The hypotheses and representations that are posted by various parts of the system can be constantly updated in parallel in these shared workspaces. Furthermore, they argue that the update of hypotheses can be done by statistical methods, which has already been shown in the past.

Wyatt and Hawes also discuss the properties that a robot architecture for a cognitive system should have: *parallel processing*, which is necessary since the processing times of the components of a cognitive robot cannot be synchronised, *asynchronous updating*, because information from different modalities arrives in the reasoning components at different times, *multiple specialist representations*, since the fields of AI have spread in many directions and each field has its own way to represent information, *understandability*, because robot systems are very complex and thus the single parts must be described in a way that is semantically easy to understand, and *incrementability*, since the robot systems must be extensible so that they can be used to solve tasks they have not been built for in the first place.

This architecture as well as its software implementation BALT & CAST was developed in the CoSy project [24]. CoSy stated three design principles for robot architectures: (*i*) *concurrent modular processing*, the robot architecture has to consist of several modules that run in parallel. This is necessary for example when several subtasks have to be completed at the same time. (*ii*) *Structured management of knowledge*, the information inside the architecture is defined by subarchitecture ontologies and general ontologies. This means that each subcomponent of the system has its own representation for its knowledge. General ontologies are used to structure subcomponents and the information flow between these components. (*iii*) *Dynamic contextual processing*, the robot architecture has to have ways to control which components can influence processing and when they are allowed to do it. This design principal is needed to ensure a goal-oriented behaviour of the system.

To get an overview of the technological side of robot architectures, please refer to Mohamed et al. [59] and Namoshe et al. [63], which give overviews for robot middlewares. Of the many robot middlewares, Player/Stage [38] and YARP [31] are widely used in the robotics community. Recently, Robot Open Source (ROS) [70] by the company Willow Garage got a lot of attention. In the JAST project, Ice [45] was used for the communication between the subcomponents of the system.

2.2.3 Human-Robot Interaction

Multimodal systems and robot architectures are both closely related to human-robot interaction (HRI), since you need processing of multimodal input as well as a technical architecture to enable a robot to interact with a human in a way that is natural for the human. In the following, we will review influential human-robot interaction systems.

The humanoid robot system that is developed at the University of Karlsruhe in the SFB588 (Sonderforschungsbereich 588, Humanoid Robots - Learning and Cooperating Multimodal Robots) is a good example for a standard HRI system [46, 21]. It is able to fuse speech and 3d pointing gestures. For the fusion, it uses a rule-based approach with an independent parser and application-specific rules. The Karlsruhe robot uses attribute value matrices for the representation of speech and gesture input. It is able to navigate in a kitchen environment and to understand multimodal commands by a human user.

Recent projects in HRI more and more focus on socially interactive robots that show the social skills needed for a successful interaction with a human. Well-known examples for social robots are for example Kismet [14, 15], Cogniron [28], or LiReC [23]. Dautenhahn [28] gives an overview of socially interactive robots and also lists a range of domains for which a robot needs more or less social skills. She gives examples such an autonomous robot for space mission, which needs no social skills at all, and robots that serve as companions in the home to assist the elderly, which need a wide range of social skills to get accepted by their owner.

Another recent HRI system was developed by the CoSy project [44]. CoSy used combinatory categorial grammar for speech processing and proposed interesting approaches for incremental speech processing and language grounding, which will be discussed in the next section.

2.2.4 Spoken Language Processing

HRI has the goal to make the interaction for the human as natural as possible. Therefore, any HRI system needs to be able to handle spoken language, which has the problem that humans tend to speak in grammatically incorrect sentences that are also sometimes incomplete. Additionally, automatic speech recognition software often provides erroneous recognition results. Thus, any HRI system needs methods for spoken language processing that deal with these challenges and provide methods for robust processing.

Most approaches for incremental language processing work with the assumption that the input to their system is well-formed. This way, they can use existing grammar formalisms

from computational linguistics. However, we think that grammar plays a less important role in holding the meaning of utterances than it is believed. This is for example backed up by a study presented by Landauer et al. [53], who show that in the latent semantic analysis (LSA), a method for summarising and representing the meaning of large text corpora, word order is not important for the correct summary of written text.

Brick and Scheutz [16] present RISE, the robotic incremental semantic engine. RISE is able to process syntactic and semantic information incrementally and to integrate this information with perceptual and linguistic information. This way, the system is able to generate feedback and actions already during the processing of utterances. The authors show an implementation of RISE, which demonstrates that the system is able to work with erroneous and missing words. However, they still assume that the humans who use their system produce sentences that are grammatically correct.

Kruijff et al. [52] present their view of incremental processing for situated dialogue in human-robot interaction. They show a complete chain of modules they are using for processing of spoken input: parsing, referent resolution, dialogue moves, event structure, and cross-modal binding. For parsing, they use combinatory categorial grammar [77], the same grammar formalism that we are using in our work. They are processing parallel interpretations that are pruned by making use of the context. For example, when the human says "take the ..." then the robot already knows from its context knowledge that probably the human wants it to pick up an object that is in reach of its arms. Interesting is also the idea of the authors to pack the representations of the open hypotheses that the system is currently processing. The approach by Kruijff et al. is still based on the assumption that the utterances by the human are grammatically correct. The authors are also writing in their conclusions that this is a point in which they want to make further investigations.

Schlangen and Skantze [71] describe "a general, abstract model of incremental dialogue processing". Their goal is to provide principles for designing new systems for incremental speech processing. Their approach is based on a token-passing topology in which so-called *incremental units* are passed from module to module inside their system. In each processing module the content of the incremental unit is extended by additional information. For this reason, certain operations can be applied to the incremental units, like the *purge*, *update*, and *commit* operations. The incremental unit also stores meta information, including which module changed the information in the unit so that the processing steps of an information unit can be retraced. The authors also present an example for a system configuration in which they apply

their design principles. However, it cannot be seen from the example how a system based on the described principles is more powerful than system that use other approaches. Also, the authors do not say anything about parallel processing or computational complexity.

2.2.5 Embodiment

Embodiment is a research area of artificial intelligence. The central idea behind embodiment is the notion that any cognitive agent that shows intelligent behaviour needs a physical body that is situated in the real world. The intelligence of embodied agents consists of showing a meaningful behaviour in their environment. The article "Elephants Don't Play Chess" by Brooks [17] was one of the first that mentioned ideas from embodiment. One of these ideas was that through embodiment the symbol grounding problem could be solved because embodied agents can measure their environment with sensors and thus do not need a representation of the world.

In their book "How the body shapes the way we think" [67] Pfeifer and Bongard define a set of design principles for building embodied agents. For our work, we regard two of these principles as important: the *three constituents* principle, which states that when building an embodied agent, the developer needs to consider the agent's environment, the so-called ecological niche, the agent's desired behaviour and tasks, and its form. If the agent shows a meaningful behaviour that follows the defined tasks and for which it makes use of its environment, then Pfeifer and Bongard refer to this as embodied intelligence. The second design principle we want to mention is the principle of *ecological balance*, which states that the complexity of a body of an embodied agent needs to match the complexity of the task it was designed for. For example, a robot that can recognise complex objects but only needs to differentiate between simple colour patterns is over-equipped. On the other hand, a robot that can only recognise colours cannot be used for handling complex objects.

Research in embodiment yields progress in recent years, especially in processing of sensorimotor data or in robot motion control in unknown environments. However, Sloman argues in [76] that the focus on embodiment has thrown back the research in artificial intelligence, because there is no proof yet that embodiment can answer problems like language processing and acquisition. He proposes to build hybrid systems that make use of ideas of classical AI and embodiment. But that raises the question, how the combination of these two fields of AI research can be combined?

One idea that could answer this question comes from researchers that design representations of objects and actions for embodied agents: Gibson [39] introduced the so-called *Affordances*. For Gibson, Affordances are properties of objects, persons, animals, and the environment which are different for each agent that interacts with these entities. For example, an elephant cannot walk on water but a water flea can. Therefore, the surface of water has the Affordance "walk-on-able" only for water fleas but not for elephants. We want to quote two sentences from Gibson [39], which we took as guidelines for our work. The first quote states how complex a representation of the environment of an agent needs to be:

> It is never necessary to distinguish all the features of an object and, in fact, it would be impossible to do so. Perception is economical.

Furthermore, Gibson also thought that the interaction between two agents can be explained by Affordances:

> Behaviour affords behaviour, and the whole subject matter of psychology and of the social sciences can be thought of as an elaboration of this basic fact.

The European project Paco+ extended the basic idea of Affordances. The central idea of this project was that for a cognitive agent objects and actions are intertwined and should not be represented apart from each other. This is due to the notion that objects are only objects because of the actions one can execute with them. For example, a glass can be used as a container for liquids, but the same glass can also be a supporting stand for other objects when it is turned upside down. Vice versa, actions cannot exist without the objects that they are related to. Paco+ called this connection between objects and actions an *object action complex* (OAC). Krüger [51] gives a formal definition of OACs, which states that an OAC consists of a unique identifier, a prediction function that codes the systems belief on how the world will change through the OAC, and a statistical measure that represents the success of the OAC within a window of the past. Wï¿½rgï¿½tter et al. [86] showed how OACs can be used to explain how cognition emerged in primates. They argue that through the ability to predict the outcome of an own action, agents can learn new actions; and they use OACs to model the predictability of the agents world. Furthermore, Geib et al. [37] and Petrick et al. [66] show how high-level planning algorithms can be linked to low-level robot control with OACs.

To conclude the related work chapter, we review two articles that stand for the debate about whether embodiment or so-called *good old-fashioned AI* (GOFAI) will lead to intelligent

autonomous agents. In his article "Today the Earwig, Tomorrow Man?", Kirsh [50] gives an good overview of the theory of action postulated by embodiment researchers. Some of the most important arguments for this theory are ([50] pp. 167–168):

- Behaviour can be partitioned into task-oriented activities or skills which can be ordered in an increasing level of complexity.
- Since the real world provides more information than what is needed for most behavioural skills, e.g. walking or running, most of those skills do not require a comprehensive world model and only a fraction of the world needs to be detected by the agent to successfully execute most of these actions.
- Thus, the most difficult problem in embodiment is to coordinate the various behavioural skills of an autonomous agent to achieve intelligent behaviour.

Kirsh arguments that these points are not necessarily false as long as one only considers simple actions for which an intelligent agent does not need to have representations of concepts. For example, he mentions a walking agent who can move in uneven terrain, for which the agent needs to be able to react fast to unexpected surfaces changes. However, Kirsh also thinks that complex actions that for example require planning cannot be implemented with purely behavioural methods. Kirsh also mentions agent to agent interaction or language understanding and generation as examples for this class of complex actions in which concepts are needed.

Brooks answered to this argumentation with his article "From Earwigs to Humans" [19] a few years later, in which he gives an overview of examples that successfully employ methods of embodiment to control autonomous intelligent robots. He argues that one has to completely understand first how to control the low-level, behavioural skills of autonomous agents before one will be able to build robots that can socially interact with humans and each other.

Our stance in this debate is oriented on Kirsh's argumentation, but we also use approaches from embodiment in our methodology: we argue that many of the low-level actions that a robot has to perform are better solved with methods from embodiment, for example, picking up an object and handing it over to a partner. However, we want to build a robot that is able to plan its own actions and to talk about these plans with a human partner. Thus, we need to use representations and concepts in our approach as well. The two approaches for multimodal fusion, which we will present in the following chapter, mirror this balance between embodiment and GOFAI.

Chapter 3

Multimodal Fusion

Researchers in robot cognition are roughly separated into two competing groups: the one group propagates that intelligent behaviour can only be reached with methods from classical artificial intelligence (AI), the other group believes that embodiment will lead to an emergence of cognitive skills in inanimate agents. The truth probably resides somewhere in the middle between these extreme positions, as it was also proposed by Sloman [76], who postulates that the focus on embodiment in recent years has slowed down AI research and that the combination of ideas from classical AI and embodiment could lead to success.

Similar to robotics, research in multimodal fusion produced solutions that are based on methods from classical AI and work for well-defined problems that follow certain patterns and can be controlled with a set of rules. However, as it turned out, the same approaches for multimodal fusion that work well for human-computer interaction (HCI) are not applicable for Human-Robot Interaction (HRI) since the environment of a robot—i.e. the real world—is much less predictable and cannot be easily described by a set of rules.

In this theoretic chapter, we will compare two approaches for multimodal fusion: one approach that is based on classical methods for multimodal fusion and another approach that incorporates ideas from embodiment. From now on, we will call the two approaches classical multimodal fusion (CMF) and embodied multimodal fusion (EMF), respectively. For comparison, we first discuss the prerequisites that are needed for CMF and EMF in Section 3.1. After that, in Section 3.2 we describe a CMF approach that was implemented for the JAST project, which relied on methods from classical AI. Finally, we propose a new approach for EMF that combines ideas from embodiment and classical AI in Section 3.3. Both of the sections that

describe the CMF and EMF approaches contain discussions about their respective advantages and disadvantages.

3.1 Classical Multimodal Fusion vs. Embodied Multimodal Fusion

In this section, we will explain the main differences between classical multimodal fusion (CMF) and embodied multimodal fusion (EMF) to clarify how we define these terms. Mainly, there is one major difference between the two approaches and all further differences arise from it: CMF systems are focussed on the user input while EMF systems are focussed on the actions of an embodied cognitive agent.

The goal of CMF is to generate representations that combine information from multiple modalities, for example from speech and gestures. These representations are processed by reasoning modules that represent the cognitive part of a system, usually a dialogue manager or some sort of logical reasoning program. Therefore, the integrated representation of the multimodal fusion component has to contain as much information from the single modalities as possible.

Typically, systems that work with CMF, combine input from speech recognition with input from a touch screen (which was often referred to as "gestures"). These systems are able to resolve ambiguous sentences by using gesture input. For example Johnston et al. [48] showed in the QuickSet project an application in which a human gave speech commands that were accompanied by pen-based input on a map that was displayed on a touch screen. In one of the examples that were presented in [48], the user said "FOLLOW THIS ROUTE" and draw a line on the map. QuickSet was able to resolve the deictic expression "this" and to infer which route to follow. Examples like this can be implemented with rule-based methods, because the input from speech and gestures can be represented by grammars and follows certain patterns, since the domain is very restricted.

Systems that use CMF are often centred around speech input that is enhanced with information from other channels. This approach fits dialogue managers that implement the information state-based approach well (for an example see [55]). These dialogue managers work excellently in scenarios in which the dialogue can be defined by a closed set of states, for example in so-called information kiosks, systems that provide information about a certain topic. One could also say that CMF systems are "utterance-oriented".

In contrast to that, the EMF approach that we present later in this chapter is "action-oriented". The main idea behind this is that each cognitive agent is able to execute a defined set of actions. For example, the JAST robot is able to pick up objects and hand them over to a human, it can look and point to objects, and it can speak to the user. If the robot should collaborate with a human in a meaningful way, it needs to produce the correct action at the right time. For this reason, rather then focussing on the input by the human and how to represent it, the robot should have a representation for its own actions and it should evaluate which of these actions it should execute given the current context. For this evaluation, it can use information from multiple modalities, which include of course human utterances, but the robot also has information from other channels as we will see below.

3.2 Classical Multimodal Fusion

After this short comparison of CMF and EMF, in this section we will describe a CMF system we developed for the JAST project. For JAST, this was the right choice because the reasoning component of the JAST robot was an information-state based dialogue manager that needed representations of user utterances as input. The CMF described in this section was used for both of the JAST system evaluations that are published in [34], [35] and [41].

3.2.1 Overview

Figure 3.1 shows the processing steps of the CMF approach. Rectangular boxes represent processing modules, rounded rectangular boxes are standing for context information that is available to the processing modules. The single processing steps are as follows: first, the speech by a human user is recognised with a speech recogniser. Following speech recognition, the output of the speech recogniser is parsed by a grammar to translate it into a logical representation that can be used to resolve the references in the human utterance (i.e. to find out which objects the human is talking about). After reference resolution, the fusion module generates a hypothesis that contains the logical form and information about the resolved entities in the logical form to the output generation part of the robot, which consists of a dialogue manager and specialised output generation modules, including reference generation and robot control.

The context information that is available to the system consists on the one hand of a grammar that is used in the parsing and output generation steps and that can also be translated into another grammar format for the speech recognition component. On the other hand, the

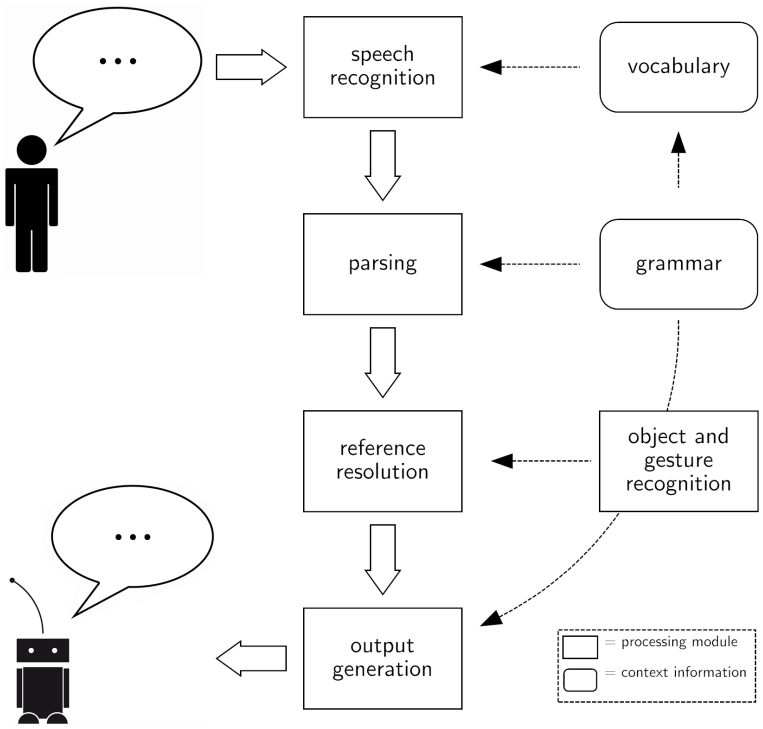


Figure 3.1: Classical multimodal fusion in JAST. Rectangular boxes represent processing modules, rounded boxes stand for context information.

fusion module can access information from visual perception modules that provide information about the objects that are laying in front of the robot and about gestures by the human partner of the robot. In the following sections, we will give a more detailed description of these processing steps.

3.2.2 Speech Recognition

In JAST, we used a commercial speech recognition system, Dragon Naturally Speaking, versions 9 and 10[1]. The speech recognition software itself was not changed, since JAST had no focus on improving speech recognition technology. Dragon Naturally Speaking comes with trained recognition models for German and English, it performs well without any training but can be adapted to certain speakers with a training program. Additionally, the speech recognition output can be pruned by using a grammar that tells the software, which words and sentence structures are allowed as recognition results, a step that is often referred to as post-processing

[1] `http://www.nuance.de/naturallyspeaking/`

in speech recognition literature.

For the JAST user evaluations, we wrote a small grammar that limited speech recognition to only recognise sentences that could be parsed by the combinatory categorial grammar (CCG) we used in the parsing step (see next section for details). This limits the sentences the robot can recognise, but vastly improves speech recognition results. Additionally, we used an external sound card (type Tascam US-122L) and a high-quality head-mounted microphone (type Sennheiser ME-3) to further improve speech recognition results.

3.2.3 Speech Processing

In JAST, we used combinatory categorial grammar (CCG) to parse and represent input sentences from speech recognition. CCG was introduced by Ades [1] and Steedman [77]. It is an extension to the categorial grammar, which is also called lexicalised grammar, of Ajdukiewicz [2] and Bar-Hillel [7]. Traditional context-free grammar formalisms use a top-down approach for parsing sentences, while combinatory grammars utilise a bottom-up approach, which brings advantages in computability and grammar development. Due to the addition of combinatory logic to the grammar formalism, CCGs produce a semantic representation of a sentence during the parsing process. For JAST, we used a CCG that was implemented with OpenCCG [84], which is a Java-based implementation of the CCG formalism. It is capable of both, parsing and realising sentences; that means it can translate utterances into a logical form as well as take a given logical form and convert it back to a sentence. OpenCCG generates hybrid logic expressions for the parsed sentence instead of combinatory logic, as explained in [6]. Figure 3.2 shows such a hybrid logic formula that was parsed with the JAST grammar and represents the sentence "take this yellow cube".

$$
\begin{aligned}
@_{t1:action}(&\text{take-verb} \wedge \\
&\langle \textsc{mood} \rangle \text{ imp} \wedge \\
&\langle \textsc{Actor} \rangle\ x1 : \mathit{animate} - \mathit{being} \wedge \\
&\langle \textsc{Patient} \rangle\ (\ c1 : \mathit{thing} \wedge \text{cube-np} \wedge \\
&\qquad \langle \textsc{det} \rangle \text{ dem-prox} \wedge \\
&\qquad \langle \textsc{num} \rangle \text{ sg} \wedge \\
&\qquad \langle \textsc{HasProp} \rangle\ (\ y1 : \mathit{proposition} \wedge \text{yellow})))
\end{aligned}
$$

Figure 3.2: Hybrid logic formula that was generated with a combinatory categorial grammar for the sentence "take this yellow cube". In hybrid logic, all entities (agents, actions, and objects) in the sentence are marked with so-called *nominals* that uniquely identify each entity.

Figure 3.3: The JAST gesture recognition can recognise three gesture types: pointing, grasping, and holding out.

To understand this logic formula, we have to illustrate the two concepts of *nominals* and *diamond operators* that are part of hybrid logics. *Nominals* can be seen as identifiers that are used to name parts of the logical form. In the present case, nominals are used to name the actions expressed in a sentence and the entities that are involved in the action. In the example, the nominal *t1:action* is used to name the take action expressed in the sentence, while the two nominals *x1:animate-being* and *c1:thing* name the actor that should execute the requested action and the cube that should be taken, respectively. The use of nominals to identify actions and entities is very useful for reference resolution, as we will see in the next section, which was one reason to use the CCG formalism in JAST.

In the logical formula we can also see the *diamond operators* ⟨MOOD⟩, ⟨ACTOR⟩, ⟨PATIENT⟩, ⟨DET⟩, and ⟨NUM⟩. These operators represent syntactic properties of the parsed sentence, including such information as that the sentence was uttered in imperative mood or that a proximal demonstrative was used as determiner to further specify a certain cube.

For the Master's thesis that was written before this work, we developed a comprehensive grammar for JAST and implemented it with OpenCCG. Please refer to [40] for a more detailed overview of this previous work.

3.2.4 Gesture Recognition

The JAST gesture recognition identifies three types of gestures: a pointing gesture, a holding out gesture, and a gripping gesture. Humans typically gesture very fast, which makes the process of automatic gesture recognition a difficult task. Therefore, a human who uses the JAST system has to hold the hand still for a moment so that the gesture recognition can correctly determine which gesture was made. Unfortunately, this prevents a natural interaction with the system, but as we will explain in the next section, the information from the gesture

channel can be used to resolve ambiguous situations in speech processing. Figure 3.3 shows a picture of the three gesture types that can be recognised; please refer to [89] and [90] for a more detailed description of the technical implementation of the JAST gesture recognition.

3.2.5 Entity Resolution

The speech processing step, which was described in Section 3.2.3, yields a logical formula of the analysed sentence that contains its grammatical structure and names all of its entities with nominals. In the next step, these entities need to be resolved (i.e. they need to be grounded in the real world) such that the dialogue manager can process them. For this, ambiguous expressions need to be resolved and each entity that describes an object needs to be mapped to the correct object(s) on the table in front of the robot.

For entity resolution, the classical approach for multimodal fusion uses the parsed spoken utterance and input from gesture and object recognition. The resolution algorithm is composed by the following steps: (1) analysis and enrichment of the logical formula that represents the spoken utterance and of the input from gesture recognition, (2) processing of speech and gesture in a rule engine that generates an integrated representation of both channels and resolves ambiguous expressions, (3) mapping of objects of the world to object entities in the integrated representation with information from object recognition.

(1) In the first step, the input from speech processing and gesture recognition is analysed and enhanced with additional information:

- Deictic expressions in the logical formula are marked, for example pronouns or determinative articles.
- Definitive articles are marked because they can be a cue that the human was talking about a specific object as in the example sentence “give me the cube.” which indicates that the user was talking about the cube before.
- The main verb in the logical expression is extracted. Since in the JAST scenario we are working mostly with imperative sentences, the main verb of most sentences gives a direct clue on the next robot action.
- Finally, the list of nominals and object descriptions and the list of objects the user pointed to are extracted from speech processing input and gesture recognition input, respectively. These lists are used in the next step to determine if the integration of speech and gesture can be resolved.

```
rule
  if
    speech.hasDeicticExpression == true
    &&
    gesture.type == PointingGesture
    &&
    speech.getTalkedAboutObjects == gesture.getPointedAtObjects
    &&
    speech.getStartTime - gesture.getStartTime < 3000 milliseconds
  then
    generate new Hypothesis(
      HypothesisType.Resolved,
      speech.getLogicalForm(),
      unifyObjectDescriptions(
        speech.getTalkedAboutObjects,
        gesture.getPointedAtObjects)
    )
end
```

Figure 3.4: Rule that combines information from speech and gesture recognition, written in a Java-style pseudo code.

(2) In the second entity resolution step, the analysed and enriched input from speech processing and gesture recognition is introduced to the working memory of a rule engine. This engine uses rules that consist of a precondition part and an action part; if the preconditions of a rule are fulfilled it generates integrated representations for speech and gestures. Figure 3.4 shows one of the rules in a Java-style pseudo code, which is used to integrate a speech utterance that contains a deictic expression with a gesture utterance. For that, in the preconditions part the rule looks for a speech utterance with deictic expression and a pointing gesture, it compares if the object descriptions of the objects the human has talked about are similar to the object descriptions of the objects the human has pointed to, and it computes if speech and gesture were uttered less than three seconds apart from each other. The value of three seconds is arbitrarily chosen and can be adjusted to the processing speed of the system. If these preconditions hold, the rule generates a new integrated representation, a so-called *fusion hypothesis*, which is resolved and contains the information from speech and gesture channel. Section 3.2.6 explains the form of fusion hypotheses in more detail. The rule engine we implemented for JAST holds a set of rules that cover all cases of interaction that can occur in the constrained JAST domain.

At this point we will not explain more of the rules, for details on the technical implementation and the scope of all rules please refer to Chapter 4.

(3) In the last step, the object entities of the generated hypotheses are mapped to objects in the real world. For this, the multimodal fusion module has a connection to the JAST world model over which it can query if the object recognition has information about objects that fit the descriptions of the logical formula. If that is the case, the world model sends a unique identifier for each object, which can then be stored in the fusion hypothesis together with nominals from the logical formula of the speech utterance. Again, Section 3.2.6 provides more information about fusion hypotheses.

The task of the multimodal fusion component is done when the three steps of the entity resolution are completed. After this step, the generated fusion hypotheses can be sent to the dialogue manager. To conclude this section about the classical multimodal fusion approach we will first describe the fusion hypotheses in more detail and then discuss the advantages and disadvantages of the approach in Section 3.2.7.

3.2.6 Fusion Hypothesis Representation

After the entity resolution step, multimodal fusion generates fusion hypotheses and sends them to the dialogue manager. This section describes how these hypotheses are represented. A fusion hypothesis consists of

- a *hypothesis type*, which describes if all objects in the logical form have been resolved or not,
- the original *logical form* from the CCG-based speech parser,
- a set of *object links*, which map the nominals of the logical formula to object ids that were obtained by the world model, and
- a *relevance score* that can be used to rank the hypotheses.

The *hypothesis type* can have one of the values *Resolved*, *Ambiguous*, *Conflict*, *Unresolved*, or *Nothing*. The following examples illustrate the usage of the hypothesis class. Each example shows preconditions (objects on table, gesture type, ...), how the hypothesis instance looks like, and how it should be interpreted.

Resolved Hypothesis

There are two cases when a hypothesis can be resolved: either the user says a sentence that has no deictic references in it or the user refers to one or more objects and points at them at the same time. The following three examples display this two cases.

First example for a resolved hypothesis

Preconditions: User says "*take a cube*". The world model knows three cubes with ids *cube14*, *cube17*, and *cube18*. In this case, nominal *c1* of the object link points to all known cubes.

```
hypothesis {
  type = Resolved,
  logical form = "@t1:action(take-verb ^
                   <tense>pres ^
                   <voice>active ^
                   <Actor>x1:animate-being ^
                   <Patient>(c1:thing ^ cube-np ^
                             <det>a ^
                             <num>sg))"
  object links = c1 -> [cube14, cube17, cube18]
  score = 1.0
}
```

Second example for a resolved hypothesis

Preconditions: User says "*take this cube*" and points to a single cube. The world model identifies the cube with id *cube14*. In this case, every object link contains only one id on the left side that points to a world model id on the right side.

```
hypothesis {
  type = Resolved,
  logical form = "@t1:action(take-verb ^
                   <Actor>x1:animate-being ^
                   <Patient>(c1:thing ^ cube-np ^
                             <det>dem-prox ^
                             <num>sg))"
  object links = c1 -> [cube14]
  score = 1.0
}
```

Third example for a resolved hypothesis

The third example shows how two resolved deictic references are represented in a hypothesis.

Preconditions: User says "*take this cube and this bolt*" and points to a single cube and a single bolt. The world model knows the cube with id *cube14* and the bolt with id *bolt05*. Here, the object links contain two items, but both point only to one world model id.

```
hypothesis {
  type = Resolved,
  logical form = "@t1:action(take-verb ^
                    <tense>pres ^
                    <voice>active ^
                    <Actor>x1:animate-being ^
                    <Patient>(a1:sem-obj ^ and ^
                              <Arg1>(c1:thing ^ cube-np ^
                              <det>dem-prox ^
                              <num>sg) ^
                              <Arg2>(b1:thing ^ bolt-np ^
                              <det>dem-prox ^
                              <num>sg)))"
  object links = c1 -> [cube14],
                 b1 -> [bolt05]
  score = 1.0
}
```

Ambiguous Hypothesis

When the user points to two similar objects and it is not clear which one she is talking about, multimodal fusion generates an ambiguous hypothesis.

Example for an ambiguous hypothesis

Preconditions: User says "*take this cube*" and points to two cubes. The world model identifies the cubes with ids *cube23* and *cube25*. Thus, in the object link nominal *c1* links to both cubes but it should only have a link to one of the cubes to become a resolved hypothesis.

```
hypothesis {
  type = Ambiguous,
  logical form = "@t1:action(@t1:action(take-verb ^
                   <Actor>x1:animate-being ^
                   <Patient>(c1:thing ^ cube-np ^
                             <det>dem-prox ^
                             <num>sg))"
  object links = c1 -> [cube23, cube25]
  score = 1.0
}
```

Conflicting Hypothesis

In JAST, there is only one case for a conflicting hypothesis, when the user speaks about a different object then she is pointing to.

Example for a conflicting hypothesis

Preconditions: User says "*take this cube*" and points to a bolt. The world model identifies a cube with id *cube16* and a bolt with id *bolt25*.

```
hypothesis {
  type = Conflict,
  logical form = "@t1:action(@t1:action(take-verb ^
                  <Actor>x1:animate-being ^
                  <Patient>(c1:thing ^ cube-np ^
                            <det>dem-prox ^
                            <num>sg))"

  object links = c1 -> [bolt25]
  score = 1.0
}
```

Unresolved Hypothesis

Unresolved hypotheses occur when the user refers to one (or more) object(s) with a deictic expression(s), but does not point at any object(s). Unresolved hypotheses can also occur when gesture recognition does not work correctly.

Example for an unresolved hypothesis

Preconditions: User says "*take this cube*" and does not point anywhere. The world model identifies a cube with id *cube16*. However, in the object link, nominal *c1* does not link to the cube, because there was no pointing gesture by the human.

```
hypothesis {
  type = Unresolved,
  logical form = "@t1:action(@t1:action(take-verb ^
                  <Actor>x1:animate-being ^
                  <Patient>(c1:thing ^ cube-np ^
                            <det>dem-prox ^
                            <num>sg))"
  object links = c1 -> []
  score = 1.0
}
```

When multimodal fusion sends its hypotheses to the dialogue manager, its work is done at this point. The dialogue manager has to decide how the robot should react on the different types of hypotheses. Thus, in the next section we will give a short summary of the CMF approach and discuss its advantages and disadvantages.

3.2.7 Discussion

The CMF approach that we have presented above has a very clear structure, it processes the input it gets from several modalities in a stepwise fashion. This is the main reason for the advantages and disadvantages that we discuss in this section.

The advantages of this approach are:

- Ambiguous situations in the interaction can be resolved by combining the information from speech and gesture recognition.
- Timing of speech and accompanying gestures can be represented in the rules that are used in the rule engine we used to implement the CMF approach. This ensures that each gesture is mapped to the right speech utterance.
- Fusion hypotheses are well defined and well-suited for processing with classical dialogue managers that implement the information state-based approach for dialogue handling. Also they contain the logical expression from the CCG parser which can be used for output generation.
- The modular setup of the approach allows to port the system to other domains. Especially, the utilisation of a rule engine can be used to change the behaviour of the robot, for example by reordering rules, and also to use the same system for other applications by loading a new set of rules.

On paper, this approach for multimodal fusion looks very convincing and it has also been shown to work very well for projects that use speech and pen input in constraint domains, see for example [26]. However, the method is not able to deal with the typical problems that occur in HRI, for example that spoken language often does not follow grammatical rules and that input recognition programs are not robust. Thus, the list of disadvantages of the CMF approach is mainly related to its rigid structure:

- There is no procedure to react on uncertain or not correctly recognised input. For example, when speech recognition does not recognise the input sentence completely correct, this error transfers through all processing steps. Of course, this means also that the robot can never execute wrong actions, which makes CMF better suited for applications in which wrong robot actions could potentially do damage to humans or objects.
- In general, the approach is too focussed on speech. One could say that this CMF method is just an extended parser that parses speech and adds information from other modalities. However, as the work by Clark [25] shows, in joint action, people often do not use as much language as one would think.
- The approach is not only speech-focused, it is also "human utterance"-focused, which means that it does not use any context information; it does not know in which state of interaction human and robot are.
- The system is extensible, but adding a new modality to the domain would mean that not only a completely new set of rules needs to be added to the rule engine, many of the old rules would also need to be rewritten.
- The CMF approach, at least in its current form, does not include anticipative behaviour of the robot. It only reacts when the user makes an utterance. However, without own initiative the robot will never be able to successfully collaborate with a human.
- Finally, this approach has the typical symbol grounding problem: every time the human mentions an object, it explicitly needs to be grounded in the robot's knowledge of the environment which consumes computing power and is an additional source for uncertainty and errors.

Therefore, in the next section we present a new approach for multimodal fusion that uses the innate abilities of the robot to overcome the disadvantages of the classical approach for modality fusion.

3.3 Embodied Multimodal Fusion

In the last section, we have seen that classical approaches for multimodal fusion focus on the integration of several channels of human utterances to yield an integrated representation that can be used for further processing. This approach was successfully applied in information systems or in multimodal devices, for example for computers that use speech recognition and touch screen input. However, if you implement this approach on an interactive robot, the robot can only react to user input, it is not able to proactively decide which action to execute next without the user doing an action first. But obviously, for a real interactive behaviour it is indispensable for a collaborative system to be able to decide on its own actions and when to execute them.

For this reason, in this section we present a new modality fusion approach that is called embodied multimodal fusion (EMF). The basic idea behind EMF is that the robot should primary focus on its own actions and abilities and that it should use the information from the human utterances (and from other input channels as we will see later in this section) to evaluate which actions it can execute given a certain context. To illustrate this, think of the following example:

Imagine a robot that can only move forward and move backward. The robot has two sensors, one in the front and one in the back, that tell the robot how much space it has in front of and behind it. This robot has the goal to be in motion all the time and that it does not touch any obstacles in the environment. The robot can use context information from its sensors to evaluate if it is better to move forward or backward, for example with a cost function that says that its always better to go to the direction in which more empty space is available. This would probably yield a movement pattern in which the robot moves back and forth in small steps. That means, the robot focusses on the two actions it is able to do and it uses the context information from its sensors to decide which of the two actions it should execute. At this point, we want to remind the reader that we want to develop a method for multimodal fusion for a goal-oriented robot. This means that we will have a robot that executes actions as long as it has reached a chosen goal. If we need a robot that also can wait between two actions, then *waiting* also needs to be one of the robot actions.

Since we are working with the JAST robot in this thesis, the actions the robot can execute are more advanced: the robot is able to manipulate objects, it can recognise objects and gestures, it can understand building plans, and it is able to talk about building steps and about

the objects in its environment. Furthermore, the robot has the goal to build the target objects that we showed in Figure 2.3, such as the windmill or the railway signal. For that, it needs to follow plans together with a human partner. Thus, the robot needs a representation formalism with which it can describe its own actions and their connection to objects, and it needs to be able to interpret the information from modalities that display human utterances, but also from modalities that present data about the state of the task and of the robot's environment.

The next sections will show that in EMF we have to solve other problems than in CMF: Section 3.3.1 shows how instead of an integrated representation for the input channels, EMF uses a representation for the robot's actions in combination with the objects in its environment. After that, Section 3.3.2 describes how the information from input modalities is used to generate actions and to evaluate their relevance given a context. Finally, Section 3.3.5 describes how the robot can decide which action to execute next, before Section 3.3.6 discusses the advantages and disadvantages of EMF.

3.3.1 Objects and Actions

The basic idea behind EMF is that the robot should evaluate at any given time, which actions it is able to execute and how likely these actions are. Therefore, it is important to make some thoughts about objects and actions at this point: how are objects and actions related to each other? Which kinds of actions are there? How can combinations of objects and actions be separated?

In the EMF approach, representations of relations between objects and actions are mainly influenced by the European project Paco+ that we reviewed already in Section 2.2.5, but we want to clarify the connection between objects and actions once more since it is one of the cornerstones of this thesis. The central idea of Paco+ was that for any cognitive agent objects and actions are inseparably intertwined. On the one hand, objects are only objects because of the actions one can execute with them. For example, a cup can be used as a container that holds liquids, but if the cup is turned upside down and some other object is placed on it, the cup becomes a supporting stand. On the other hand, actions cannot exist by their own: only when applied to the appropriate object, actions are "born" and "exist" in the world. Therefore, objects and actions should not be examined apart from each other and Paco+ used the term *object action complex* (OAC) to name this relation between objects and actions.

The Paco+ project saw OACs as a representation format that can be used in several levels of processing in a cognitive agent. Therefore, the original definition of OACs also contains

information about two states of the agent's environment that represent the precondition and the outcome of a transfer function, which is also part of the OAC [51]. In this thesis, we do not need the full capabilities of OACs, but only a way to represent objects and actions in the EMF approach; thus, from now on we will talk about *OAClets* by which we mean a complex of an object and an action that has no further innate functions.

Definition 1. *An OAClet is defined as a triple*

$$(\mathcal{O}, \mathcal{A}, \mathcal{R}) \tag{3.1}$$

containing

- *an object* $\mathcal{O}$ *which is associated to*
- *an action* $\mathcal{A}$ *and vice versa, and*
- *a relevance score* $\mathcal{R}$*.*

Objects and actions will be further defined in the following sections. The score $\mathcal{R}$ shows the relevance of an OAClet given a context, it can be used to decide whether to execute an OAClet or to sort a set of OAClets.

3.3.1.1 Objects

Each OAClet contains an object $\mathcal{O}$ that has a set of properties. We call these properties an *object description* and we express the query to show the description of an object by

$$\mathcal{O}.description() \tag{3.2}$$

The examples we present to illustrate the EMF approach are taken from the JAST domain. The environment of the JAST robot is quite restricted. Hence, the object representations are not very complex. However, EMF could also be applied to richer domains that apply ontology-based representations which are quite powerful in their expressiveness.

Objects are clearly defined in the JAST scenario: the robot knows Baufix objects, which are used to build complex assemblies. Baufix objects are defined by their *type*, their *colour*, and/or their *size*. A type can be one of the following: cube, bolt, slat, nut, tower, l-shape, windmill, or railway signal. Since the robot is only able to recognise the more complex assemblies and their substeps (tower, l-shape, windmill, railway signal) but cannot disassemble them or execute any other actions with them than with other objects, we do not make a difference between simple and complex objects. Object colours can have one of the following values: blue, green, orange,

red, or yellow. The size of an object is one of the following: small, medium, or large. Type, colour, and size of an object can not be combined randomly, which lies in the nature of the Baufix pieces, for example nuts are always orange. We use the following expressions when we want to formally describe that the type, size, or colour of an object $\mathcal{O}$ are queried[1]:

$$\mathcal{O}.type() \tag{3.3}$$

$$\mathcal{O}.size() \tag{3.4}$$

$$\mathcal{O}.colour() \tag{3.5}$$

Besides its object description, every object also has external properties that are defined through their use in combination with an action. Therefore, we define a set of predicates that define these action-related object properties:

$$instantiatedObject(\mathcal{O}) \tag{3.6}$$

$$abstractObject(\mathcal{O}) \tag{3.7}$$

$$planned(\mathcal{O}) \tag{3.8}$$

$$graspable(\mathcal{O}) \tag{3.9}$$

$$grasped(\mathcal{O}) \tag{3.10}$$

The first two of these predicates generally describe objects: An object $\mathcal{O}$ is called *instantiated object*, if it physically exists in the environment of the robot and if it can be recognised by one of the robot's sensors. Instantiated objects are expressed by the predicate $instantiatedObject(\mathcal{O})$. In JAST, instantiated objects have an additional property, which is the object's position on the table in front of the robot and we write

$$\mathcal{O}.position() \tag{3.11}$$

to describe a query for the position of an object $\mathcal{O}$. An object $\mathcal{O}$ is called *abstract object*, if it has not been recognised by one of the robot's sensors, but the robot needs to be able to talk about it. Abstract objects are expressed by the predicate $abstractObject(\mathcal{O})$. In JAST, abstract objects are introduced when the robot loads a building plan and cannot see all objects that are needed to complete the plan. For more details please refer to Section 3.3.3.2

[1]Please note: the query functions for an object's colour and size are only applicable for simple objects.

The rest of the action-related object property predicates describe objects in the context of the JAST task: If an object $\mathcal{O}$ is needed to complete the next step of a plan it is called a *planned object* and we express this by the predicate $planned(\mathcal{O})$. Planned objects can be either instantiated or abstract and the actions for planned objects change, depending on if they are instantiated or abstract. If an object $\mathcal{O}$ is in reach of the robot's arms, we express this by the predicate $graspable(\mathcal{O})$. Finally, if the position of an object $\mathcal{O}$ is in one of the robot grippers, we express this by the predicate $grasped(\mathcal{O})$.

3.3.1.2 Actions

To formally work with the action of an OAClet, we need to define some predicates and functions. These are listed in Equations 3.12 to 3.14. Formally, we mark an action $\mathcal{A}$ by the predicate $action(\mathcal{A})$; furthermore, each action has a name and a type which can be queried by the functions $\mathcal{A}.name()$ and $\mathcal{A}.type()$

$$action(\mathcal{A}) \tag{3.12}$$

$$\mathcal{A}.name() \tag{3.13}$$

$$\mathcal{A}.type() \tag{3.14}$$

The type of an action is defined by using the action-related object predicates that were defined in the last section. These predicates can be used to classify actions. An action classification cannot be defined generally, because it depends on the abilities of the robot (or cognitive agent) for which the classification is done. Therefore, we will show an example action classification for the JAST robot. This robot can execute actions with its arms and its head:

- *take*, take an object
- *give*, take an object and give it to the human
- *pointTo*, point to an object on the table
- *show*, show an object that is already in the robot's hand
- *open [left/right] hand*, open the left or right gripper
- *close [left/right] hand*, close the left or right gripper
- *lookAt*, use the head to look at an object or to the human

Furthermore, the robot can generate speech output. Speech can be used in various ways and for innumerable purposes. In this work, we will focus on only a few ways to use speech, which are related to the task of the robot building an assembly object with the human:

- *askFor*, the robot can ask the human to put a certain object on the table
- *tellAbout*, the robot can tell the human to pick up a certain object

The classification that divides these actions by object predicates is displayed in Table 3.1. The table has two columns that are built by the domain-independent object predicates *instantiatedObject*() and *abstractObject*(). The actions of the JAST robot can then be filled in according to the domain-dependent predicates. Since there exist actions that can only be applied to objects that are graspable, but there are also actions that can be applied to all instantiated objects, the predicate *graspable*() also needs to be listed in its negated form. The table does not show the two actions *close hand* and *open hand*, since those actions are not related to objects, but to parts of the robot.

	instantiatedObject()	*abstractObject*()
graspable()	give, lookAt, pointTo, take	—
¬*graspable*()	lookAt	—
grasped()	show	—
planned()	tellAbout	askFor

Table 3.1: Action classification for the actions of the JAST robot.

Until now, we only described action types and how they can be classified. However, actions also have temporal properties, i.e. each action has a start time and an end time, which can be used to measure the duration of an action and also to express temporal relations between two actions. In equations 3.15 to 3.18, we define a set of predicates that formally describe the temporal relations between actions, where $\mathbb{A}$ stands for the set of all actions. These predicates are similar to those defined in Allen's temporal logic [3].

$$precedes() : (\mathbb{A} \times \mathbb{A}) \rightarrow bool \tag{3.15}$$

precedes() returns true if the end time of an action $\mathcal{A}$ is earlier than the start time of a second action $\mathcal{A}'$.

$$succeeds() : (\mathbb{A} \times \mathbb{A}) \rightarrow bool \tag{3.16}$$

succeeds() returns true if the start time of an action $\mathcal{A}$ is later than the end time of an action $\mathcal{A}'$.

$$includes() : (\mathbb{A} \times \mathbb{A}) \rightarrow bool \tag{3.17}$$

includes() returns true if the start time of an action $\mathcal{A}$ is earlier than the start time of an action $\mathcal{A}'$ and the end time of $\mathcal{A}$ is later than the end time of $\mathcal{A}'$.

$$includedBy() : (\mathbb{A} \times \mathbb{A}) \rightarrow bool \tag{3.18}$$

includedBy() returns true if the start time of an action $\mathcal{A}$ is later than the start time of an action $\mathcal{A}'$ and the end time of $\mathcal{A}$ is earlier than the end time of $\mathcal{A}'$.

To shortly summarise this section: actions and objects are closely coupled together and can only exist in parallel. We use OACIets to represent combinations of objects and actions. An OACIet consists of an object $\mathcal{O}$, an action $\mathcal{A}$, and a relevance score $\mathcal{R}$. Object descriptions consist of an object's *type*, *size*, *colour*, and a *position* in case of instantiated objects; the combination of type and other properties is not random but follows the objects innate features. Action $\mathcal{A}$ can only be applied to object $\mathcal{O}$ when certain preconditions are met. These preconditions can be expressed by object predicates which are either domain-independent or domain-dependent. Furthermore, actions also have a start time and an end time, which can be used to express the temporal relations between two actions.

3.3.2 Input Channels

In the last section, we have shown how objects and actions are represented in EMF. Now, we will explain how the information from various input channels can be used to generate and evaluate OACIets; or to phrase it differently, in this section we will show how context information influences the robot in its decision which action to execute next.

In the EMF approach we are following one of the design principles that were established by the CoSy project [44]: *concurrent modular processing.* In Cosy, this principle was applied on robot architectures, which have to consist of single modules that run in parallel. With this design, complicated tasks can be split into subtasks that are executed in parallel by specialised parts of the system, for example the vision system may track a human while the actuator control makes sure that the robot does not collide with the human. This design principle also implies that the system consists of several specialised subarchitectures, which can be renewed

or extended easily due to the modular architecture design. In the EMF approach, we apply this design principle on how we process information from input channels. In contrast to the CMF approach, in which the data was processed in a stepwise fashion, EMF uses the input data to constantly update the relevance of the robot's OAClets in parallel.

EMF can handle input from diverse modalities such as object recognition, task planning, robot body input, speech recognition, and gesture recognition. For this, it separates these channels into two categories: *action-generating channels* and *action-evaluating channels*. The first category, action-generating channels, are modalities that provide information that is valid for a longer time, for example from object recognition, task planning, and robot body input. OAClets are generated by the information from these channels. The latter category of action-evaluating input channels stands for modalities that are used to compute the relevance scores from a list of OAClets. Speech recognition and gesture recognition are the two channels that belong to this category in the case of JAST. In the following sections we will give formal definitions for action-generating and action-evaluating channels and demonstrate how these apply to the modalities of the JAST domain.

3.3.3 Action-Generating Channels

Action-generating channels (AGC) are input modalities that provide information through which OAClets can be generated. This means that these channels provide information about objects and about object predicates, as they were defined in Section 3.3.1. Informally speaking, AGCs have a function that generates OAClets. This is formally expressed in Definition 2.

Definition 2. *We define an action-generating channel as an input modality that has a function*

$$\mathcal{G} : \mathcal{S} \rightarrow \mathcal{S}' \tag{3.19}$$

where $\mathcal{S}$ and $\mathcal{S}'$ are states of the world and $\mathcal{S}$ holds all preconditions that allow the generation of an OAClet that is part of $\mathcal{S}'$.

Furthermore, we need definitions for functions to generate and update OAClets, and a function with which we can query, if a certain action or object is part of an OAClet. While the function to generate a new OAClet is external, the update function and the query function are class functions of all OAClets.

Definition 3. *We define the function generate() as a function that takes an object $\mathcal{O}$, an action $\mathcal{A}$, and a relevance score $\mathcal{R}$, and generates a new OAClet with the given values.*

$$generate() : (\mathcal{O} \times \mathcal{A} \times \mathcal{S}) \rightarrow OAClet(\mathcal{O}, \mathcal{A}, \mathcal{R}) \tag{3.20}$$

Definition 4. *We define the function update() as a function that is part of an OAClet, which takes new values for the OAClet's object, action, or relevance score and exchanges it inside the OAClet.*

$$OAClet.update() : OAClet \rightarrow OAClet \tag{3.21}$$

Definition 5. *We define the function contains() as a function that is part of an OAClet. The function takes an object $\mathcal{O}$ or an action $\mathcal{A}$ and returns a boolean value, depending on whether $\mathcal{O}$ or $\mathcal{A}$ are part of the OAClet.*

$$OAClet.contains() : \mathcal{X} \rightarrow bool \qquad \text{where} \qquad \mathcal{X} \in \{\mathcal{O}, \mathcal{A}\} \tag{3.22}$$

In the JAST scenario, there are three modalities that are AGCs: object recognition, task planning, and robot body input. In the following sections, we will show how these input channels generate OAClets so that the robot can assess at each time point during an interaction, which actions it can execute and how likely they are.

3.3.3.1 Object Recognition

The JAST object recognition provides information about objects that are laying on the table in front of the JAST robot. This information consists of the objects' type, its properties, its position, and a unique ID that identifies the object (which is basically the objects "name"). The object information is updated as fast as possible, which effectively means several times per second. However, the objects on the table are usually not moving very often, thus the information from object recognition can be seen as semi-permanent.

Object recognition generates three types of events: either a new object is recognised, an already known object changes its location on the table in front of the robot, or an object disappears from the table. Therefore, we need to define how the EMF approach handles these three events and how the action-generating function $\mathcal{G}$ is defined for object recognition in these three cases. First, we look at the event when object recognition sends information about a new object. In that case, EMF executes a series of steps to handle this event:

(1) *Match with OAClets that contain abstract objects.* In the first step, EMF looks into the set of currently available OAClets, to see if there are any OAClets that contain abstract objects that match the object description of the new instantiated object. Remember, abstract objects are objects that are not visible to the robot, but it is able to talk about them, for example because they are part of a building plan. If the object description of the new object matches the object description of the abstract object, the action of the OAClet is changed. This is

expressed by function $\mathcal{G}_{updateAbstObj}$ in Equation 3.23.

$$\begin{aligned}&\mathcal{G}_{updateAbstObj} : if\\&\quad \exists oaclet(abstractObject(\mathcal{O}), \mathcal{A}, \mathcal{R}) \wedge\\&\quad \exists instantiatedObject(\mathcal{O}') \wedge\\&\quad \mathcal{O}.description() = \mathcal{O}'.description()\\&\quad \rightarrow oaclet.update(\mathcal{O}', \mathcal{A}', \mathcal{R})\end{aligned} \tag{3.23}$$

were the new action $\mathcal{A}'$ needs to be chosen from Table 3.1 to match the instantiated object $\mathcal{O}'$. Actions for abstract objects are for example the action *askFor* that can be used by the robot to ask for an object that is needed for a plan. If the fusion module finds such actions with matching object descriptions, the action can be deleted and replaced by new actions in the next step.

(2) *Determine suitable action type for object and generate OAClets.* Depending on the location of the newly recognised object, a set of new OAClets has to be generated. The actions of these OAClets can be determined using Table 3.1. This is expressed by function $\mathcal{G}_{generateOAClet}$ in Equation 3.24.

$$\begin{aligned}&\mathcal{G}_{generateOAClet} : if\\&\quad \exists instanitatedObject(\mathcal{O})\\&\quad \rightarrow generate(\mathcal{O}, \mathcal{A}, \mathcal{R})\end{aligned} \tag{3.24}$$

Consider the following example that clarifies this step: object recognition recognises a yellow cube that is in reach of the robot and has ID *o001*. In this case EMF generates five new OAClets

$$\begin{aligned}&0.3, give(cube(o001, yellow))\\&0.3, lookAt(cube(o001, yellow))\\&0.3, pointTo(cube(o001, yellow))\\&0.3, show(cube(o001, yellow))\\&0.3, take(cube(o001, yellow))\end{aligned}$$

The example shows the OAClets in the order $relevance, action(object(properties))$. The number of the relevance score is not important right now, it will be explained in more detail in Chapter 4. If the cube is not reachable by the robot, EMF only generates the OAClet

$$0.3, lookAt(cube(o001, yellow))$$

The second type of object recognition event—change of an object's location—is easier to handle than the introduction of new objects. In this case, EMF first needs to lookup if the object predicates of the object that changed location have changed. If that is the case, the actions of all OAClets that contain the object that changed location need to be updated or

deleted according to Table 3.1. This is expressed by function $\mathcal{G}_{changedObjLoc}$ in Equation 3.25.

$$\begin{aligned}&\mathcal{G}_{changedObjLoc} : if\\&\quad \exists instantiatedObject(\mathcal{O}) \wedge\\&\quad \mathcal{O}.location.hasChanged = true \wedge\\&\quad \exists oaclet(\mathcal{O}, \mathcal{A}, \mathcal{R})\\&\quad \rightarrow oaclet.update(\mathcal{O}, \mathcal{A}', \mathcal{R})\end{aligned} \tag{3.25}$$

OAClets have to be deleted in two cases: when an object changes location and thus certain actions cannot be applied to the object any more, and when an object disappears from the table in front of the robot. This is expressed by function $\mathcal{G}_{deleteOAClet}$ in Equation 3.26.

$$\begin{aligned}&\mathcal{G}_{deleteOAClet} : if\\&\quad \neg\exists instantiatedObject(\mathcal{O})\\&\quad \rightarrow delete(\forall oaclet.contains(\mathcal{O}))\end{aligned} \tag{3.26}$$

3.3.3.2 Task Planning

Usually, multimodal fusion components make no use of input from a planning component, for example in the CMF approach that was presented in Section 3.2. However, every cognitive agent that should execute sensible actions needs to be goal-directed, a property that was also noted by Hawes et al. in [44]. The goal of the robot in the JAST scenario is to build Baufix assemblies together with the human. Thus, the goal of the robot is closely related to an assembly plan and an integration of the information from the JAST task planner in the multimodal fusion component is crucial for successful action selection and execution.

The JAST task planner follows a standard approach: plans are saved in a tree structure, where each leaf represents a Baufix piece that is needed for the plan. Nodes represent substeps of the plan and the root node represents the completely assembled object. Figure 3.5 shows an example plan tree for a Baufix windmill. When the planner gets the information about an executed plan step (i.e., an object was taken from the table) it is able to determine to which part of the plan the plan step belongs and if a subplan was completed. It can also give information about the objects that are needed to complete the currently active plan step.

The main problem while using the task planner is that the JAST robot has a very limited view on its environment. It cannot tell if the human completed a plan step correctly or not, it has to guess the current state of the plan by other clues. For this reason, in one of the JAST system evaluations we asked the subjects to explicitly tell the robot when they finish a plan substep or a complete plan [34, 35]. However, this often led to confusion among the experiment participants. Thus, for the EMF approach we decided to take only the information about disappearing objects from object recognition as a clue about finished plan steps.

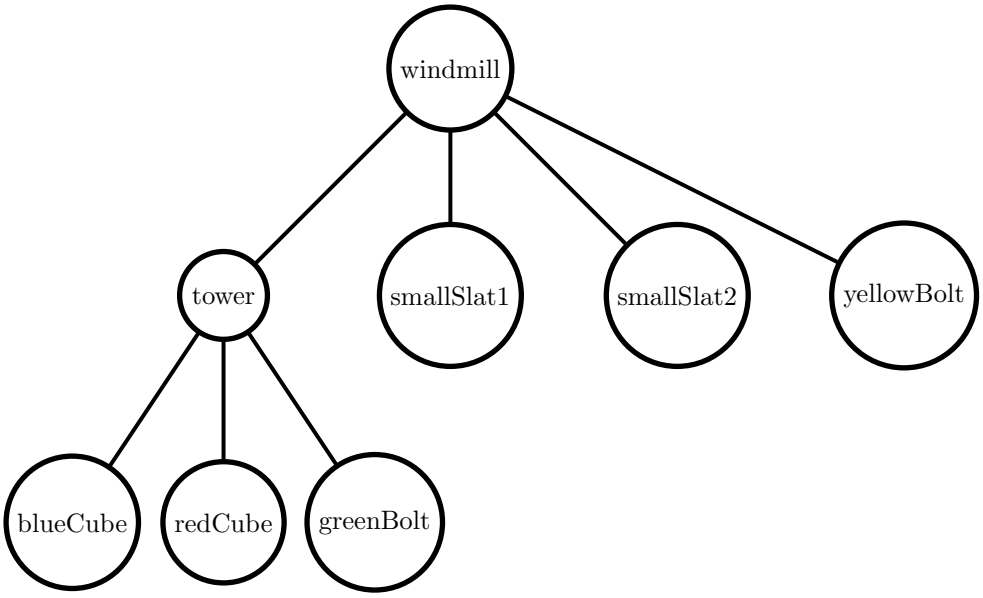


Figure 3.5: Assembly plan for a Baufix windmill, represented in a tree structure.

The task planner generates two types of events that are relevant for EMF: on the one hand, when loading a new plan, new OACLets need to be generated and old OACLets need to be reevaluated. On the other hand, if a substep of a plan is completed, the existing OACLets need to be reevaluated as well. Thus, the task planner is an AGC as well as an AEC.

First, we look at the case when the task planner loads a new plan. For this, EMF gets information by the task planner about which plan was loaded, what the building steps of that plan are, and which instantiated objects are needed to build these steps. EMF needs to check for each of the objects of the building plan, if it already knows an OACLet, which has an instantiated object and from which the object description is similar to the object description of the planned object. If EMF does not know such an object, it has to generate a new OACLet with an abstract object. This is expressed by function $\mathcal{G}_{generateAbstrOAClet}$ in Equation 3.27.

$$\begin{aligned}&\mathcal{G}_{generateAbstrOAClet} : if \\ &\quad \exists planned(\mathcal{O}) \wedge \\ &\quad \neg\exists oaclet(instantiatedObject(\mathcal{O}'), \mathcal{A}', \mathcal{R}') \wedge \\ &\quad \mathcal{O}.description() = \mathcal{O}'.description() \\ &\quad \rightarrow generate(\mathcal{O}, \mathcal{A}, \mathcal{R})\end{aligned} \tag{3.27}$$

If EMF already knows an OACLet with an instantiated object that has the same object description as the object from the plan, EMF updates the relevance score of this OACLet. This is

expressed by function $\mathcal{RE}_{PlanInstOAClet}$ in Equation 3.28.

$$\begin{aligned}&\mathcal{RE}_{PlanInstOAClet} : if\\&\quad \exists planned(\mathcal{O}) \wedge\\&\quad \exists oaclet(instantiatedObject(\mathcal{O}'), \mathcal{A}', \mathcal{R}') \wedge\\&\quad \mathcal{O}.description() = \mathcal{O}'.description()\\&\quad \rightarrow oaclet.update(\mathcal{O}', \mathcal{A}', \mathcal{R})\end{aligned} \tag{3.28}$$

The completion of a substep of a plan triggers the second task planner event. In this case, no new OAClets need to be generated, but the already existing OAClets need to be reevaluated. This reevaluation is similar to the update that was shown in equation 3.28, but in this case EMF makes no difference between OAClets with instantiated or abstract objects, which is expressed in equation 3.29.

$$\begin{aligned}&\mathcal{G}_{updatePlanOAClet} : if\\&\quad \exists planned(\mathcal{O}) \wedge\\&\quad \exists oaclet(\mathcal{O}', \mathcal{A}', \mathcal{R}') \wedge\\&\quad \mathcal{O}.description() = \mathcal{O}'.description()\\&\quad \rightarrow oaclet.update(\mathcal{O}', \mathcal{A}', \mathcal{R})\end{aligned} \tag{3.29}$$

To illustrate the formal description of processing of task planner input, consider the following example: human and robot should build a windmill together, following the plan that was presented in Figure 3.5. Object recognition identifies six objects on the table in front of the robot, a blue cube, a yellow cube, a green bolt, a yellow bolt, and two small slats. For these objects, EMF has already generated a set of OAClets (the set is shortened here, only one action per object is displayed):

$$\begin{aligned}&0.3, give(cube(o001, blue))\\&0.3, give(cube(o002, yellow))\\&0.3, give(bolt(o003, green))\\&0.3, give(bolt(o004, yellow))\\&0.3, give(slat(o005, small))\\&0.3, give(slat(o006, small))\end{aligned}$$

When the user loads the plan for the windmill, EMF adds a new OAClet for an abstract red cube, since this Baufix piece is also needed to build the target object. The action for this action is selected from Table 3.1. Furthermore, EMF raises the relevance scores of all OAClets that contain an object with an object description that fits to one of the pieces that is needed to build a windmill. In the example, all pieces that are needed to build the tower, which is the first substep of the windmill, are raised to a value of 0.9 all relevance scores of OAClets with objects that are needed later in the plan are raised to a value of 0.6. As we will show in Chapter 4, a rating function does the reevaluation in the actual implementation of the EMF approach on the JAST robot.

$$0.9, give(cube(o001, blue))$$
$$0.9, give(cube(o002, yellow))$$
$$0.9, give(bolt(o003, green))$$
$$0.6, give(bolt(o004, yellow))$$
$$0.6, give(slat(o005, small))$$
$$0.6, give(slat(o006, small))$$
$$0.6, askFor(abstract(cube(red))$$

To summarise this section: we believe that task planning is essential for any goal-directed behaviour of a cognitive robot and thus must be included in multimodal fusion. The information from the task planner can on the one hand be used to generate OACIets with abstract objects; on the other hand, the knowledge about pieces that are needed to complete a building plan can be used to reevaluate the relevance score of already existing OACIets.

3.3.3.3 Robot Body Input

Since the new approach for multimodal fusion is centred around the actions of the robot, there is an additional input channel that is usually not considered in multimodal fusion: the information that is generated by the robot itself. Since the robot is able to measure information about its own current status, for example it can evaluate the position of its joints, this information can also be used to generate OACIets. This idea goes back to Pfeifer [67] who states in one of his design principles for embodied agents that they are also generating sensor input which can be fed back into their reasoning system.

In case of the JAST robot, the available input information is rather limited. At the time of writing this thesis, there is one interface that can be used to get information about how much the robot grippers are opened and another interface works as a trigger to tell the rest of the system when the robot is moving. However, even this little information can be used to generate OACIets that relate the actions *close* and *open* to the left or right robot gripper respectively. This cannot be expressed formally with our current set of definitions. However, we want to give an example: if the left gripper is open then only the OACIet

$$1.0, close(gripper(left))$$

is active. The action of the OACIet has to be changed every time when the grippers is opened or closed.

The robot body input channel has much potential. For JAST, we did not exploit this modality too much, but in future versions of the system it would be thinkable to for example

publish the pose information of the arms, which could be used to decide if the robot needs to move to defined positions given a certain context.

3.3.4 Action-evaluating Channels

In contrast to action-generating input channels, there are those modalities that are solely used to reevaluate already generated OAClets. Thus, one could say that the information from these *action-evaluating channels* (AEC) is used by one partner of the interaction to direct the attention of the other partner to certain objects or actions. This follows the work by Clark [25], who pointed out that in joint action most activities can be divided into the two classes *directing-to* and *placing-for*. Both of these activity classes are used to draw the partner's attention to actions or objects either by concretely directing the attention to them or by placing the objects (or sometimes agents also place themselves) in the visual field of the partner. Formally, we express an AEC according to Definition 6.

Definition 6. *We define an action-evaluating channel as an input modality that has a function*

$$\mathcal{RE} : OAClet \rightarrow OAClet \tag{3.30}$$

which takes an OAClet and reevaluates the relevance score of that OAClet.

In the JAST project, two modalities, which have been relevant for the CMF approach already, can be counted as AECs: speech recognition and gesture recognition. These modalities have in common that they represent information from human utterances and that the information from these channels is only valid for a short period of time.

3.3.4.1 Speech Processing

In the EMF approach, speech is similarly processed as in the CMF approach, which was also described in Section 3.2.3: the input of a speech recognition software is parsed with CCG to yield a logical formula. After the parsing step, unlike in the CMF approach, EMF directly uses the logical expressions to reevaluate relevance scores of existing OAClets. In this section, we will show how this reevaluation is formally defined. Furthermore, we show how the use of OAClets can increase the robustness of the system in case of grammatically not correct sentences.

The approach described here assumes that the human utterances in the interaction with the robot mainly consist of imperative sentences, in which the humans directly express what they want the robot to do. These sentences are parsed with a grammar to yield logical expressions. Figure 3.6 shows the hybrid logic formula for the input sentence "take a yellow cube".

$$
\begin{aligned}
&@_{t1:action}(\text{take-verb} \wedge \\
&\qquad \langle \textsc{mood} \rangle \text{ imp} \wedge \\
&\qquad \langle \textsc{Actor} \rangle\ x1 : animate - being \wedge \\
&\qquad \langle \textsc{Patient} \rangle\ (\ c1 : thing \wedge \text{cube-np} \wedge \\
&\qquad\qquad \langle \textsc{det} \rangle \text{ indef} \wedge \\
&\qquad\qquad \langle \textsc{num} \rangle \text{ sg} \wedge \\
&\qquad\qquad \langle \textsc{HasProp} \rangle\ (\ y1 : proposition \wedge \text{yellow})))
\end{aligned}
$$

Figure 3.6: Hybrid logic formula for the sentence "take a yellow cube".

In this example, you can see that the action as well as the object mentioned in the sentence are marked by the nominals *t1:action* and *c1:thing*, respectively. These nominals can be automatically extracted and can be used to reevaluate OAClets that contain similar actions and objects as the extracted ones. Please note, that the object that was extracted from the hybrid logic formula is abstract, while the objects from the OAClets can either be instantiated or abstract. This is expressed by function $\mathcal{RE}_{reevaluateActionAndObject}$ in Equation 3.31

$$
\begin{aligned}
&\mathcal{RE}_{reevaluateActionAndObject} : if \\
&\quad \exists action(\mathcal{A}) \wedge \\
&\quad \exists abstractObject(\mathcal{O}) \wedge \\
&\quad \exists oaclet(\mathcal{A}', \mathcal{O}', \mathcal{R}') \wedge \\
&\quad \mathcal{A}.name() = \mathcal{A}'.name() \wedge \\
&\quad \mathcal{O}.description() = \mathcal{O}'.description() \\
&\quad \rightarrow oaclet.update(\mathcal{A}, \mathcal{O}, \mathcal{R})
\end{aligned}
\tag{3.31}
$$

Until now, we were only looking at the case in which speech processing parses a whole sentence, similar to what we described in Section 3.2.3. With EMF, you can also reevaluate OAClets with only partially or wrongly recognised sentences. For this, in Equations 3.32 and 3.33 we define the two functions $\mathcal{RE}_{reevaluateAction}$ and $\mathcal{RE}_{reevaluateObject}$ with which OAClets can be reevaluated with the information of a single action, object, or even object property that was recognised by speech recognition. In Chapter 4 we will demonstrate some examples that use these functions.

$$
\begin{aligned}
&\mathcal{RE}_{reevaluateAction} : if \\
&\quad \exists action(\mathcal{A}) \wedge \\
&\quad \exists oaclet(\mathcal{A}', \mathcal{O}', \mathcal{R}') \wedge \\
&\quad \mathcal{A}.name() = \mathcal{A}'.name() \\
&\quad \rightarrow oaclet.update(\mathcal{A}, \mathcal{O}, \mathcal{R})
\end{aligned}
\tag{3.32}
$$

$$\begin{aligned}&\mathcal{RE}_{reevaluateObject} : if\\&\quad \exists abstractObject(\mathcal{O}) \wedge\\&\quad \exists oaclet(\mathcal{A}', \mathcal{O}', \mathcal{R}') \wedge\\&\quad \mathcal{O}.description() = \mathcal{O}'.description()\\&\quad \rightarrow oaclet.update(\mathcal{A}, \mathcal{O}, \mathcal{R})\end{aligned} \tag{3.33}$$

In summary, one can say that speech processing in EMF is still based on traditional techniques from computational linguistics, but it differs from CMF because it uses the information from parsed sentences to directly evaluate the robot's actions. Furthermore, EMF is equipped with the basic tools to implement a strategy to reevaluate OAClets based on only partially recognised sentences, which increases the robustness of the approach.

3.3.4.2 Gesture Recognition

In CMF approaches, gesture recognition is usually one of the key modalities, or to put it differently: gestures are most of the time the only other input modality that are integrated with speech. In these approaches, gesture recognition is used to resolve ambiguous situations in which information from speech is not enough to fully resolve an utterance. However, when working with a robot in the real world, gesture recognition often is not working robustly and recognition results are poor.

Furthermore, two publications from the JAST project show that in joint action gestures might be of less importance than it is usually reported in literature. Foster et al. showed in [33] that the analysis of a data corpus revealed that in situations in which two persons work together and in which both see objects they are working with, they preferably use haptic-ostensive references for objects rather then pointing at them, which means, that they pick up the object to steer the partner's attention to it. Additionally, de Ruiter et al. showed in [29] that people use iconic gestures to describe objects that are redundant with the information in the speech and does not add any new information to what they have said.

In EMF, gesture recognition is used to reevaluate OAClets. At this point, we should define three update functions for reevaluating OAClets, which cover the three types of gestures that the JAST gesture recognition can process: pointing, holding out, and grasping. However, in practice only pointing gestures are used by the human to direct the robot's attention to Baufix objects. Therefore, in Equation 3.34 we only define update function $\mathcal{RE}_{reevaluatePointedToObject}$, which is similar to function $\mathcal{RE}_{reevaluateObject}$ that was defined in Equation 3.33, but handles

instantiated instead of abstract objects.

$$\begin{aligned}&\mathcal{RE}_{reevaluatePointedToObject} : if\\&\quad \exists instantiatedObject(\mathcal{O}) \wedge\\&\quad \exists oaclet(\mathcal{A}', \mathcal{O}', \mathcal{R}') \wedge\\&\quad \mathcal{O}.description() = \mathcal{O}'.description()\\&\quad \rightarrow oaclet.update(\mathcal{A}, \mathcal{O}, \mathcal{R})\end{aligned} \tag{3.34}$$

In Chapter 4, we will provide more information and examples in gesture recognition. For now, we want to draw your attention to the fact that the use of OAClets in the EMF approach improves the robustness of the system, because even if gesture recognition fails to deliver any results, the robot is still able to complete its tasks, which stands in contrast to CMF.

3.3.5 Action Selection

The EMF approach enables the robot to represent its own actions and to evaluate how relevant they are given a context. However, this yields a new problem, which does not arise in the CMF approach: the robot needs to decide when it wants to execute an OAClet and it needs to define which OAClet it should choose for execution. In CMF, this decision is easy to make. Every time when the human says an utterance to the robot, it reacts on the utterance with an action. This behaviour is not sufficient for a true interaction, the robot needs to proactively take part in the interaction and to anticipate its next own actions. Therefore, we need to develop strategies that help the robot to decide, which OAClet it should execute at a given time. In this thesis, we will not be able to completely solve this issue, but we present our view on action selection and show an implementation for an action selection mechanism in Chapter 4.

We see three events that could trigger that the robot chooses an OAClet for execution:

- *Triggering by a dedicated modality that is only responsible for choosing the right moment to execute an OAClet.* This could be for example a human that helps the robot, how it is done in Wizard of Oz experiments[1], or a component that recognises human social signals that for example measures how long the human looks at the robot and draws conclusions from this measurement.

- *Triggering by one of the action-generating or action-evaluating channels.* All modalities, which are used to generate or evaluate OAClets, generate events that could also be cues for the robot to select one of the OAClets for execution. For example, the human could give the robot a direct command, which certainly is an obvious cue for action selection.

[1] In Wizard of Oz experiments, subjects interact with a computer or robot system that is controlled by a human being, but the subjects believe that the system is autonomous.

- *Triggering by a mathematical model.* The robot could use the relevance scores of the OAClets to decide which and when to select an OAClet. A simple model would be to execute OAClets when their score increases over a certain threshold or a more advanced statistical model could be trained by using the input data by the modalities.

For JAST, we developed an action selection strategy that uses cues from several modalities to trigger the selection of an OAClet for execution. Furthermore, we use an effective strategy to select the right OAClet in the case when the relevance scores of several OAClets are similar. The triggers for the OAClet selection mechanism are:

- *Direct command of the human.* When the human gives a direct command , for example "give me a blue bolt", to the robot, the action selection strategy tests if an OAClet that fits the command exists and executes it right away. When the robot cannot find an OAClet that directly fits to the human order, EMF uses the information of the analysed utterance to reevaluate the remaining OAClets. This might lead to a situation in which EMF triggers one of the other action selection mechanisms.

- *Pieces disappear from table.* Since Baufix pieces only disappear from the table when the robot hands them over to the human or when the human picks up one of the pieces, this is a trigger for the OAClet selection mechanism. In this case, EMF first waits for the update information from the task planner and then calls the mechanism.

- *Time-controlled external trigger.* We did some experiments with an external trigger that calls the OAClet selection mechanism every time when the robot does not get input from any of the modalities for a predefined time. However, we discovered that this trigger is not trivial to set up. Therefore, we decided not to use an external trigger for OAClet selection in this thesis. In the outlook in Chapter 6 we discuss some of the possible directions in which we want to go with the external trigger in the future.

Finally, we shortly describe the rather simple but effective OAClet selection mechanism, we are using: when EMF selects an OAClet for execution directly because of an command by the human, the mechanism simply executes this OAClet. In all other cases, the mechanism filters all OAClets with the highest relevance scores. If the resulting list contains more than one OAClet, the mechanism sorts the remaining entries according to an *action priority list.* For example the action *give* could have the highest priority, so that the robot always hands over pieces to the human before executing other OAClets. This is inspired by the task hierarchies

that were introduced by Sentis and Kathib [73]. Section 4.3.4 discusses the OAClet selection mechanism in more detail.

3.3.6 Discussion

Similar to the discussion in Section 3.2.7, in this section we will discuss the advantages and disadvantages of the approach for embodied multimodal fusion we presented above.

The main difference between the CMF and EMF approaches is that the central idea of EMF is that the robot should represent its own actions in combination with objects. This leads to the situation, in which the multimodal fusion module has to integrate more modalities than before, which also means that the module needs to handle more tasks than in CMF, for example by using the task planner and by incorporating action selection mechanisms. We argue that this is an advantage, because, despite losing some of the modularity of the CMF approach, at the same time the robustness of the system increases and the information of the input channels is optimally used. Furthermore, we see the following advantages:

- In EMF, the symbol grounding problem does not exist, since as soon as an object is recognised this information is represented in OAClets, which deletes an explicit symbol grounding step. Of course, one could also argue that in EMF we ground objects as soon as the multimodal fusion module gets information about objects from its input modules—in case of JAST those modules are object recognition and task planning. Hence, EMF has a symbol grounding step, but executes it at an earlier processing stage as in the CMF approach.

- Although the new approach is not as modular as the old approach, it is even more extensible. New context-providing modules can always be added to the system, because the representation of OAClets is not coupled to a certain input modality, which is unlike in the CMF approach, where utterance representation is closely linked to speech recognition. Furthermore, in EMF, OAClet evaluation is separated from OAClet generation, which makes it possible for developers to add new action-evaluating or action-generating channels to the system.

- If one of the input modules fails, in EMF the robot is still able to generate and evaluate OAClets; also the focus on speech does not exist any more in the new approach. This increases the robustness of the system.

- EMF is well-suited for statistical methods. These methods could be used to train the parameters that define the robot's behaviour or they could be used as an action selection mechanism.
- Finally, the new approach combines Information from modules that are based on logic, for example the task planner, and from modules that use statistical methods, for example object recognition. This property is again made available through the use of OAClets, which was also noted in [37] and [66].

Of course there are also disadvantages in EMF:

- The robot can potentially execute the wrong action at the wrong time. This means that EMF cannot be used for applications in which it is crucial that the robot never makes any errors. However, we see that EMF can be extended, for example with a list of actions the robot is not allowed to execute, which has to be altered given the current context. Also, if there are certain actions in an application, which the robot should *never* execute, then those actions or action-object combinations can be restricted in EMF.
- In the current form of EMF, there is no way to influence, when the robot executes an action. For example in the case of the JAST construction task, there are situations in which the human has to assemble the target objects, which takes some time. In these cases the robot should wait for the human to finish, before continuing with the interaction. However, we think that this also means that the robot needs to be equipped with a module that monitors the human's actions, which is not the case in the current version of the JAST robot.
- EMF has no memory, for example it does not record the things that have already been said. This knowledge would be needed for example to resolve anaphora or to generate referring expressions that make reference to the past. However, we think that this problem could be solved by using external modules that provide memory, for example a dialogue history, which could be another source of input to multimodal fusion.
- Finally, EMF is harder to port to new domains than CMF. With that, we do not mean the theory of action-generating and action-evaluating channels but the concrete implementation of the approach for the JAST robot.

Chapter 4

Implementation

In this chapter, we give an overview for the implementation of the two approaches for multimodal fusion that we developed in this thesis. First, in Section 4.1 we show the interfaces of the JAST robot architecture that we use for communication between input modules, multimodal fusion, and output modules. After that, we introduce the implementations for the CMF approach in Section 4.2 and the for the EMF approach in Section 4.3. Both of the latter sections also contain processing examples to demonstrate the differences between the two methods.

4.1 JAST Robot Architecture

The JAST robot architecture, which we showed in Figure 2.2, consists of a set of modules that implement the robot's abilities. The communication between these modules is implemented with the middleware Internet Communication Engine (Ice)[1] [45], which is designed as a communication layer for components of a distributed system. For that, Ice supports various operating systems and programming languages. In JAST, we are using components that are running on Linux and Windows and are programmed in C++, Java, Prolog, and Python.

Ice has an own language to define interfaces for the communication between components of a distributed system. In the following, we will describe some of the interfaces that were defined for the JAST robot. We will only describe the interfaces that are relevant for our work.

4.1.1 Commonly Used Definitions

The environment of the JAST robot is encoded in its architecture by a set of constructs that can by applied in all interface definitions. In JAST, *locations* can be described precisely by

[1] Ice is availble for download at `http://www.zeroc.com`

table coordinates or imprecisely by a semantic description. Figure 4.1 depicts the constructs for these descriptions.

Baufix objects are described by type, colour, and size. However, in JAST, we have two views on objects, one view from object recognition and another view from the world model. Figure 4.2 shows the Ice definitions for these two views: objects from object recognition are described with an *OrecObject*, which contain an ID that was generated during the recognition process and the exact coordinates of the object on the table in front of the robot. The robot uses the coordinates for example when it picks up an object. The object view by the world model is described in the structure *InstantiatedObject*. An instantiated object contains an ID by the world model, which is for example used by multimodal fusion and dialogue manager, and a semantical representation of the objects location, for example "TableArea→User". OrecObject and InstantiatedObject both have an object description that contains their type and other properties, which are stored in the structure *ObjectDesc*.

```
enum LocationType {
  User, RobotHand, Table, Assembled, UnknownLocation, AnyLocation
};
enum TableArea {
  UserArea, RobotArea, CommonArea, OutsideArea, AnyArea
};
```

Figure 4.1: Ice interfaces for locations on the table in front of the robot.

4.1.2 Interfaces for Input Modalities

Figure 4.3 shows the interface for *speech recognition*, which sends an n-best list of recognised sentences. It stores the best hypothesis in the string *top* and the rest of the alternative recognition results in a separate list of strings. Furthermore, speech recognition sends two timestamps that mark when the human started and ended to speak.

Figure 4.4 shows the interfaces for *gesture recognition*, which has two several separate methods over which it sends information. In multimodal fusion we use only two of these interfaces. Gesture recognition uses the interface *handMoved* when it recognises a hand in the viewing sight of the robot, and it uses the interface *handPositionStable* to publish a recognised gesture when the human holds the hand still. The latter interface sends a list of so-called *GestureHypotheses*, which contains the three gesture types that gesture recognition can classify along with

```
struct ObjectDesc {
  ObjectType type;
  ObjectPropMap props;
};
struct OrecObject {
  ObjectDesc description;
  TableCoordinates coords;
  string orecId;
};
struct InstantiatedObject {
  string worldId;
  ObjectDesc desc;
  Location loc;
};
```

Figure 4.2: Object descriptions in Ice.

a confidence score of each gesture.

For multimodal fusion, the *world model* is the interface to object recognition. The world model can either directly publish object recognition events or multimodal fusion can query information that is stored in the world model. Figure 4.5 shows the world model publishing interfaces. It uses the interface *objectIntroduced*, to send information when a new object appears on the table, and it uses the interfaces *objectChangedLocation* and *objectChangedCoordinates* when human or robot move an object around on the table. World model publishes the semantic location and the coordinates of objects separately to be consistent with the two views on objects.

Figure 4.6 shows the world model query interfaces. Here we only show the two interfaces we are using, which is on the one hand the interface *getObjectIds*, which sends a list of object ids that fit to a given object description, and on the other hand the interface *getInstantiatedObject*, which returns an instantiated object for a given world model id.

```
void recognizedTurn (string top,
    ::Ice::StringSeq alternatives,
    ::jast::common::Timestamp startTime,
    ::jast::common::Timestamp endTime);
```

Figure 4.3: Ice interface for speech recognition.

```
void handMoved(::jast::common::TableCoordinates coords,
    ::jast::common::Timestamp time);
void handPositionStable(GestureHypothesisList hypotheses,
    ::jast::common::Timestamp time);
```

Figure 4.4: Ice interfaces for gesture recognition.

```
void objectIntroduced (string objectId,
    ::jast::common::Location loc,
    ::jast::common::Timestamp time);
void objectChangedLocation (string objectId,
    ::jast::common::Location oldLoc,
    ::jast::common::Location newLoc,
    ::jast::common::Timestamp time);
void objectChangedCoordinates (string objectId,
    ::jast::common::TableCoordinates oldCoords,
    ::jast::common::TableCoordinates newCoords,
    ::jast::common::Timestamp time);
```

Figure 4.5: Publishing Ice interfaces for world model.

4.1.3 Interfaces to Reasoning Components

In this section, we review the interfaces from multimodal fusion to the reasoning layer of the JAST robot architecture. Figure 4.7 shows the interfaces to the *dialogue manager*. CMF uses the interface *setInput* to send the fusion hypotheses which were presented in Section 3.2.6 and are also shown in their Ice definition in Figure 4.7.

EMF uses information from the *task planner* and Figure 4.8 shows the interfaces for that. Task planner has many interfaces, but we use only the interfaces *getRequiredPieces*, to query the Baufix pieces that are needed for the currently loaded building plan, interface *getBasicSteps*, to query the building steps of the plan, and interface *userAssembledPieces* to tell task planner when the user finished a substep of the plan.

4.1.4 Interfaces to Output Components

Finally, we show the interfaces to the output components of the JAST robot. Figure 4.9 shows the interfaces to the robot arms. With these interfaces we can control all the actions the robot can execute, for example *open* or *close* one of the robot's hand, *pickUp* an object, or *moveAway* the robot arms to their default position. The figure also shows the interfaces we use to query

```
::Ice::StringSeq getObjectIds(::jast::common::ObjectDesc desc,
    ::jast::common::Location loc);
::jast::common::InstantiatedObject getInstantiatedObject(string id);
```

Figure 4.6: Query Ice interfaces for world model.

```
struct Hypothesis {
  HypothesisType type;
  string lf;
  ObjectLinkSet objectlinks;
  ::jast::common::InstantiatedObject instobj;
  ::jast::listener::GoalInferenceOutput giOutput;
};
void setInput(Hypothesis hyp);
```

Figure 4.7: Ice hypothesis definition and interface for dialogue manager.

the robot status, for example *getGripperWidth* sends the information how much the robot's grippers are currently opened, and we use the interface *isReachable* to check if a certain object is reachable by a given robot hand.

4.2 Classical Multimodal Fusion

In this section, we describe the processing flow in the CMF approach.

4.2.1 Overview

Figure 4.10 shows an overview for the processing in the CMF approach. The central reasoning component of CMF is the so-called *working memory* that uses a *rule engine* to generate fusion hypotheses. The *multimodal fusion* module translates the input from *speech recognition* and *gesture recognition* into *speech* and *gesture* elements which the working memory can process. For the translation, multimodal fusion uses a CCG to parse the spoken utterances. The recognised gestures can be translated directly. Furthermore, the working memory uses the interfaces to the world model to get information about objects on the table. When the working memory has completed a fusion hypothesis, it sends the hypothesis to the dialogue manager.

```
::jast::common::ObjectDescList getRequiredPieces (
    ::jast::common::ObjectType target);
BasicStepList getBasicSteps (::jast::common::ObjectType obj,
    string stepId);
void userAssembledPieces (::Ice::StringSeq pieceIds,
    ::jast::common::ObjectType result);
```

Figure 4.8: Ice interfaces for task planner.

```
enum Hand {
    HandLeft, HandRight, HandAny
};
void close(Hand h);
double getGripperWidth(Hand h);
void give(Hand h);
bool isReachable(Hand h,
    ::jast::common::TableCoordinates coords);
void moveaway();
void open(Hand h);
void pickUp(Hand h,
    ::jast::common::TableCoordinates coords);
void point(Hand h,
    ::jast::common::TableCoordinates coords);
void putDown(Hand h,
    ::jast::common::TableCoordinates coords);
void show(Hand h);
void take(Hand h);
```

Figure 4.9: Ice interfaces for robot body.

4.2.2 Speech Processing with OpenCCG

In this thesis, we will only shortly explain the single steps of the parsing process. Please refer to [40] for a more detailed description of CCG and OpenCCG. Shortly, the steps in the parsing process are: loading of a grammar, parsing of all hypotheses from speech recognition, filtering of complete sentences, and information extraction of the generated logical form of the complete sentence.

When no complete sentence was parsed, we are using a strategy to extract as much information from the speech recognition result as possible. For that we split the recognition result into bigrams and parse these sentence parts. We extract information from these sentence parts

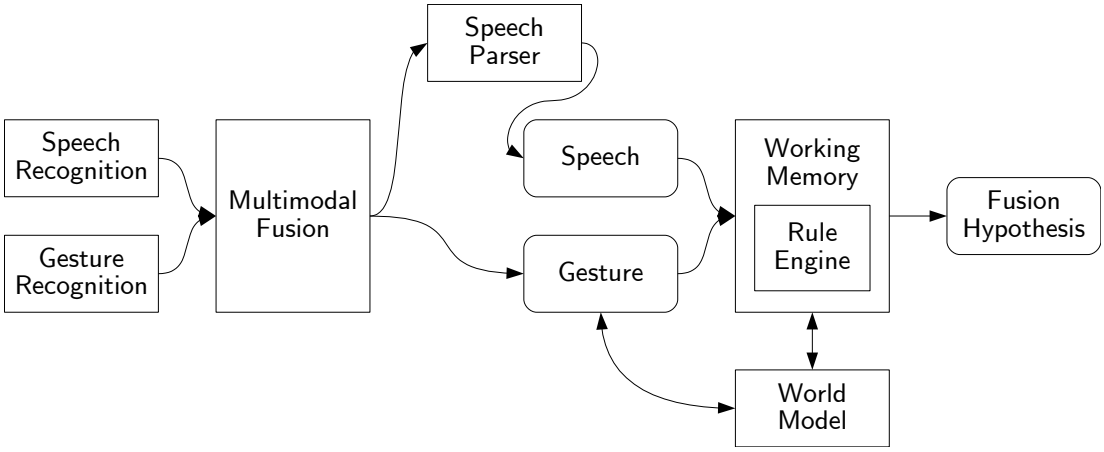


Figure 4.10: Overview for processing in the classical multimodal fusion approach.

if possible or repeat the parsing process with single words if no information extraction was possible with bigrams. The extracted information is then represented in speech elements, as we will show in the following section.

4.2.3 Speech and Gesture Elements

The information of the parsed sentences and the recognised gestures has to be packed into speech and gesture elements before the working memory can process them. The extracted information from the parsed sentences is

- *deictic expression*, which are a cue for the working memory that it needs a pointing gesture to resolve the sentence that contains the expression.
- *definite article*, which is a cue that either a gesture that refers to a certain object or only one object that fits to the object description in the spoken sentence needs to be present to resolve the utterance.
- *logical formula* that was generated during the parsing process, which is needed by dialogue manager and output generation.
- *nominals in the logical formula*, which stand for objects. These are stored with the object descriptions that fit to the nominal and to the object description in the spoken utterance.

Similar to verbal utterances, gestures are translated to a gesture element. This element contains

- *a gesture type*, which can be pointing, holding out, or grasping.
- *coordinates* of the place on the table on which the gesture was executed. This can be a cue on which object the user really pointed at in case he/she pointed at several objects at once.
- *a list of objects the human pointed to.* This list contains the world model id of the objects as well as the corresponding instantiated objects which contains the object description and the object position.

Besides this modality-specific information, speech and gesture elements both additionally hold a start and end time that are needed by the working memory to decide if a speech and a gesture element should be joined together.

4.2.4 Working Memory

The *working memory* is the heart of the CMF approach: it fuses the data from speech, gesture, and object recognition and generates fusion hypotheses. The working memory is implemented with Drools [20], a Java-based rule engine. The engine can work with facts, which is why we translate speech and gestures into basic elements that consist of statements. Additionally to this information, we introduce timestamps that contain the current time into the working memory on a regular basis. This way, the working memory can handle time points as facts, which is used in some of the rule definitions. For example the timestamp can be used to delete speech and gesture elements that are too old.

For JAST, the rule engine contains 13 rules that cover the fusion of spoken utterances and gestures for the user evaluations in which CMF was used. Appendix A.1 shows the complete listing of rules, in the following list we document what the rules do and which kind of fusion hypothesis they generate.

- Rule “pointing gesture” handles a single pointing gesture that is older than a defined timeout. Sets the gesture element inactive and generates no hypothesis.
- Rule “deictic speech and pointing gesture, resolved” fuses a sentence with a deictic expression and a pointing gesture that have timestamps that are close to each other, generates a resolved hypothesis.

- Rule "deictic speech and pointing gesture, unresolved" fuses a sentence with a deictic expression and a pointing gesture that have timestamps that are close to each other, but generates an unresolved hypothesis.
- Rule "deictic speech and pointing gesture, conflict" fuses a sentence with a deictic expression and a pointing gesture that have timestamps that are close to each other, but have object descriptions that do not fit to each other, which thus generates a conflicting hypothesis.
- Rule "deictic speech and pointing gesture, ambiguous" fuses a sentence with a deictic expression and a pointing gesture that have timestamps that are close to each other, but generates an ambiguous hypothesis, for example because the user points at two similar objects at the same time.
- Rule "speech, deictic expression, no gesture" generates an unresolved hypothesis, because it has a speech element with a deictic expression in it, but cannot find a gesture element.
- Rule "speech, no deictic expression, definite determiner, ambiguous" generates an ambiguous hypothesis, because it has a speech element with a definite determiner but no gesture element to resolve the situation.
- Rule "speech, no deictic expression, no definite determiner, resolved" generates a resolved hypothesis, because it found a speech element with no deictic expression or definite determiner.
- Rule "speech, no deictic expression, definite determiner, resolved" generates a resolved hypothesis, for example when the user says "give me the cube" and there is only one cube.
- Rule "speech, no deictic expression, no definite determiner, unresolved" generates an unresolved hypothesis, because the user talked about objects the robot cannot see.
- Rule "speech, no deictic expression, definite determiner, unresolved" generates an unresolved hypothesis, because the user talked about a specific object the robot cannot see.
- Rule "set old utterances inactive", this rule sets speech and gesture elements inactive that are older than a defined timestamp, so that the other rules do not use these utterances.
- Rule "nothing happened for TIMEOUT msecs", this rule simply prints a message when no new gesture and speech items were added to the working memory for a time duration that is defined in variable TIMEOUT.

4.2.5 Processing Example

To conclude the section about the implementation of CMF, we present an example in which the human speaks a sentence that contains a deictic expression and makes a pointing gesture, which are then unified by CMF.

In the example, a red cube, a yellow cube, and a yellow bolt are laying in front of the robot. The yellow cube and the yellow bolt are laying close to each other. Object recognition has correctly recognised the three objects and published the results to the world model. World model assigned the IDs *o001* to the red cube, *o002* to the yellow cube, and *o003* to the yellow bolt. The human says to the robot "give me this cube" and points on the table. Gesture recognition cannot classify the pointing gesture precisely and publishes that it found a pointing gesture with which the human pointed to the yellow cube and the yellow bolt.

Speech recognition has recognised the spoken utterance correctly and sends it to CMF. CMF parses the sentence with CCG which yields the logical expression that we present in Figure 4.11. From this expression, CMF extracts information and builds a speech element, which is shown in Figure 4.12.

$$
\begin{aligned}
@_{g1:action}(&\text{give-verb} \wedge \\
&\langle \textsc{mood} \rangle \text{ imp} \wedge \\
&\langle \textsc{Actor} \rangle\ x1 : animate - being \wedge \\
&\langle \textsc{Patient} \rangle\ (\quad c1 : thing \wedge \text{cube-np} \wedge \\
&\qquad\qquad \langle \textsc{det} \rangle \text{ dem-prox} \wedge \\
&\qquad\qquad \langle \textsc{num} \rangle \text{ sg }) \wedge \\
&\langle \textsc{Recipient} \rangle\ (\ p1 : animate - being \wedge \text{pron} \wedge \\
&\qquad\qquad \langle \textsc{num} \rangle \text{ sg} \wedge \\
&\qquad\qquad \langle \textsc{pers} \rangle \text{ 1st }))
\end{aligned}
$$

Figure 4.11: Hybrid logic formula that was generated with a combinatory categorial grammar for the sentence "give me this cube".

CMF also generates a gesture element for which it combines information from gesture and object recognition. We show the gesture element in Figure 4.13.

Speech and gesture element also contain timestamps, when the utterances started and ended. For the example, we assume that both elements were not more than three seconds apart. CMF introduces the speech and gesture element into the working memory which uses rule "deictic speech and pointing gesture, resolved" to generate a resolved hypothesis. We show the rule in Figure 4.14.

```
String top == "give me this cube";
String[] alternatives == empty;
boolean hasDeictic == true;
private boolean hasDefDet == false;
private String mainVerb == "give";
private String logicalForm; // see Figure of logical expression
private Document document;  // see Figure of logical expression
Hashtable<String, ObjectDesc> idsAndObjects == c1:thing -> cube;
```

Figure 4.12: Example for a speech element in the working memory.

```
String type == Pointing;
private TableCoordinates coords == 1, 1;
private Hashtable<String, ObjectDesc> pointedAtIds ==
    o002 -> cube, yellow
    o003 -> bolt, yellow;
private Vector<InstantiatedObject> pieces;
```

Figure 4.13: Example for a gesture element in the working memory.

The rule generates a resolved hypothesis, because all preconditions of the rule are fulfilled: the spoken sentence has a deictic expression, the gesture is a pointing gesture, and the object descriptions from the spoken utterance and the pointed at objects can be unified. Figure 4.15 shows the generated resolved hypothesis that CMF sends to dialogue manager. Please note, that the nominal in the object link points to the correct world model ID after the unification process.

4.3 Embodied Multimodal Fusion

In this section, we present our implementation for the EMF approach. We give a short overview and highlight some of the details. Finally, we give a processing example for an interaction between human and robot.

4.3.1 Overview

Figure 4.16 shows an overview of the processing flow in EMF. In EMF, the *multimodal fusion* component gets information from more input modules than in the CMF approach: the figure shows the AGCs *object recognition*, *task planner*, and *robot body* on the top, and the AECs

```
rule "deictic speech and pointing gesture, resolved"
  when
    speech : Speech( hasDeictic == true, active == true )
    gesture : Gesture( type == "Pointing", active == true )
    timer : Timestamp()

    // test if objects talked about and pointed at match
    eval( getHypothesisType(speech.getIdsAndObjects(),
        gesture.getPointedAtIds()) == HypothesisType.Resolved )
  then
    insert (
        new FusionHypothesis(
            HypothesisType.Resolved,
            speech.getLogicalForm(),
            speech.getDocument(),
            unifyHashtables(speech.getIdsAndObjects(),
                gesture.getPointedAtIds()),
            getEmptyInstantiatedObject(),
            getEmptyGoalInferenceOutput()
        )
    );

    speech.setActive(false);
    update( speech );
    gesture.setActive(false);
    update( gesture );
end
```

Figure 4.14: Example for a rule in the working memory.

speech recognition and *gesture recognition* on the bottom of the input modules. Multimodal fusion uses the information from these channels to generate and evaluate OAClets. For this, multimodal fusion has an *OAClet container*. From the OAClet container, an *action selection and execution* mechanism chooses OAClets for execution that fit to the current situation.

4.3.2 OAClets

An efficient implementation of the representation of OAClets is crucial for EMF, because the more actions and objects the robot can handle, the more OAClets it has to generate and evaluate at any given point in time. Therefore, we need to find strategies so that the robot can add new OAClets or evaluate them without having to look at all OAClets every time it gets new

```
hypothesis {
  type = Resolved,
  logical form = @g1:action(give-verb ^
                            <mood>imp ^
                            <Actor>x1:animate-being ^
                            <Patient>(c1:thing ^ cube-np ^
                                      <det>dem-prox ^
                                      <num>sg) ^
                                      <Recipient>(p1:animate-being ^ pron ^
                                                  <num>sg ^
                                                  <pers>1st))
  object links = c1:thing -> [o002]
  score = 1.0
}
```

Figure 4.15: Example for a resolved hypothesis.

information from the input modalities.

In our implementation of the EMF approach, we chose to store actions and objects as separate entities that have links to each other. That means, the robot has a list of actions it can execute, which EMF loads at the beginning of an interaction. Only when object recognition sends information about objects on the table or when the task planner wants to generate an OAClet with an abstract object, EMF generates links from the new object to the appropriate actions. For this, every action has a list of objects it is related to and vice versa. This way, when EMF gets information about a particular action it can reevaluate all objects related to that action in one step without needing to reevaluate other actions. In the same way, when EMF gets updates for an object and needs to reevaluate all OAClets related to this object, it simply calculates the new relevance score of that object and all actions related to that objects are also automatically updated.

To clarify this, consider the following example: the JAST robot knows the actions take, give, lookAt, PointAt, tellAbout, and askFor. When we place an object in the robot's workspace, object recognition sends information about this instantiated object and EMF generates links between the actions take, give, lookAt, and pointAt and the instantiated object. Figure 4.17(a) shows these links. If we move the instantiated object from the robot's workspace to the human's workspace, EMF has to disconnect the object from the actions that the robot can only execute with graspable objects. In the example this are the actions take, give, and pointAt. Figure 4.17(b) shows this new link configuration. Finally, when the task planner sends information

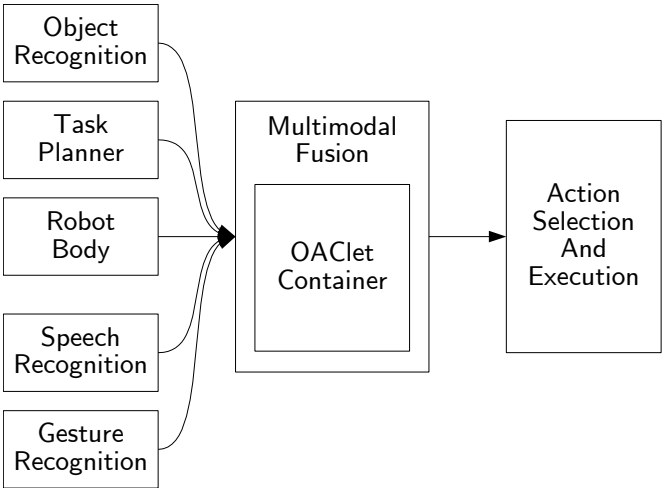


Figure 4.16: Overview for processing in the embodied multimodal fusion approach.

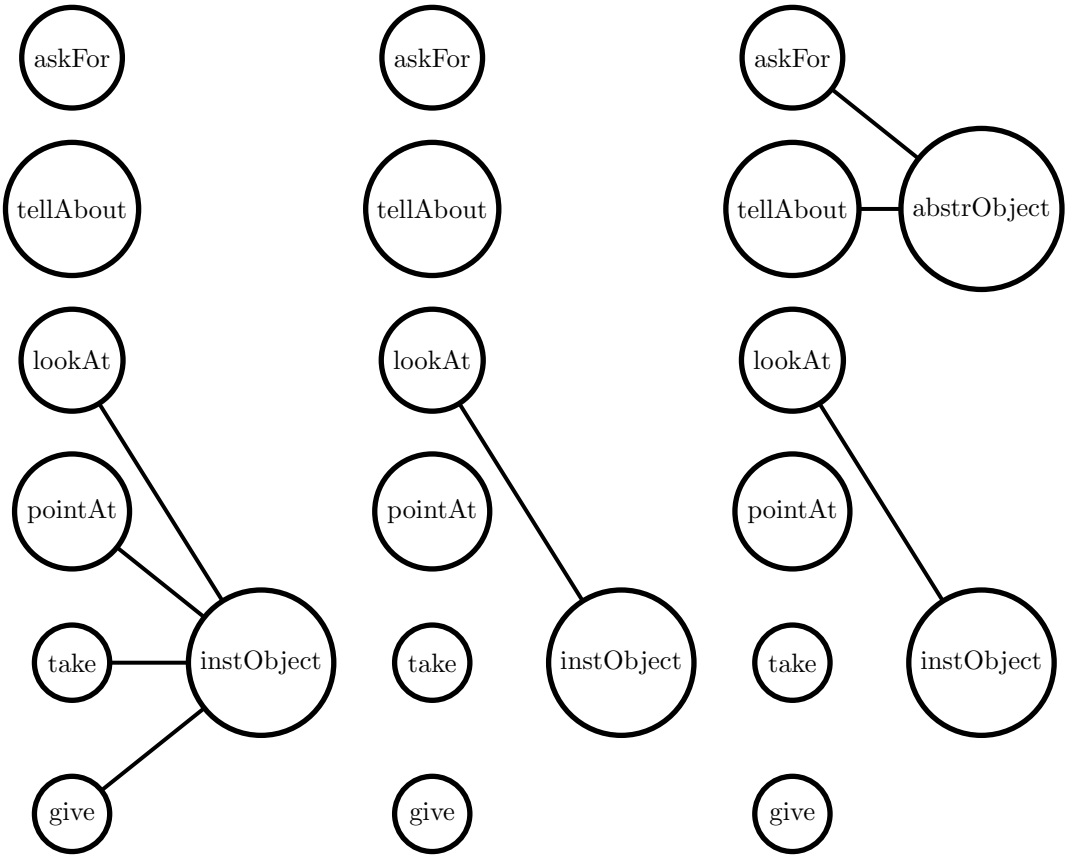
about an abstract object that is needed for the current plan, but EMF cannot find an object on the table that fits to the object description of the abstract object, EMF generates links from actions askFor and tellAbout to the abstract object. Figure 4.17(c) shows these new links.

4.3.3 Relevance Calculation

EMF stores actions and objects separately to represent OAClets, thus it also needs to split the relevance score of an OAClet and store it separately in the action and object entities. In our implementation, every action and object holds their own relevance score for the current context. This means, that EMF needs to calculate the relevance scores of OAClets on the fly and cannot simply readout the score from a list. This has the advantage that the relevance score of OAClets can be influenced by application-specific guidelines. For example, the importance of input channels can be set differently, depending on whether the channel is more relevant for a domain or how robust the input processing from the channel is. In our EMF implementation, we calculate the relevance score of an OAClet with the formula in Equation 4.1.

$$r = a + o \tag{4.1}$$

Where r stands for the relevance score, and a and o are the current scores for the action and the object of the OAClet. When an AEC sends information to multimodal fusion, EMF reevaluates the relevance scores of an action or object according to the formula in Equation 4.2.



(a) Graspable instantiated object (b) Instantiated object out of reach (c) Abstract object

Figure 4.17: Examples for links between actions and objects.

$$v = v + c_i \tag{4.2}$$

Where v stands for the old value of the action or object and c_i is the confidence value for input channel i from which the information for the reevaluation came. In our current implementation we have set the values for the input channels by hand to the following values:

- 0.4 speech recognition, we still regard speech as the most important input channel
- 0.25 object recognition, has the same value as task planner
- 0.25 task planner, has the same value as object recognition, since we want to give all AGCs the same importances

- 0.1 gesture recognition, has only a low value because the results from gesture recognition are not very reliable.

The sum of the channel confidence values sums up to 1 so that we can normalise the OAClet relevance scores. EMF treats input by the robot body separately because it is not linked to objects on the table.

4.3.4 Action Selection

We already explained our action selection mechanism in Section 3.3.5. In the implementation the algorithm for action selection follows these steps:

1. Get set of OAClets with highest values.
2. If number of selected OAClets is above pre-defined threshold k, display message that no OAClet can be selected.
3. If number of selected OAClets is below pre-defined threshold k, chose OAClet that has action with highest priority.

For the last step in this algorithm the robot uses a priority list in which the actions the robot can execute are sorted by their priority. The list can be used to control the robot's behaviour. For example, if the list has the order $give > tellAbout > askFor$ the robot would show a rather active behaviour, because it would try to give the user an object, and only if that is not possible (or not likely enough) it would tell the user which Baufix piece to take next or it would ask for a piece that is needed for the current plan but is not on the table. If the priority list is ordered differently, for example $tellAbout > give > askFor$ the robot would show a more passive behaviour, or it could also be configured to make sure that all pieces for the current building plan are on the table before the interaction starts, by applying the priority sorting $askFor > give > tellAbout$. This method is influenced by hierarchical robot control that was introduced by [73] and for example implemented in [56] and [42].

4.3.5 Processing Example

To conclude the section about the EMF implementation, we present an example for an interaction between human and robot and show how EMF handles the information from the input channels. For this, we use the JAST construction task that was introduced in Section 2.1.3, specifically we look at the case that the robot builds a windmill together with the human.

At the start of the interaction, the table in front of the robot is set up according to the windmill initial table layout which is presented in Figure A.1 of the appendix. To make the example more appealing, we leave out a red cube which should be placed in the human's workspace and is needed for the windmill's tower. This means, we have a green cube and a blue cube, two small slats and a medium slat in the robot's workspace. On the human's side there is a yellow cube, a yellow bolt, a green bolt, a red bolt, and an orange nut.

For the example, the robot has the following task hierarchy: $give > askFor > tellAbout > pointAt > take > lookAt$. That means, the robot should preferably handover pieces to the human, when this cannot be done than it should ask for pieces that are needed for the current building plan, or tell the users that pieces for the loaded target object are on their side, and so on. Furthermore, we use the confidence values for the input channels that were already listed above: 0.4 speech recognition, 0.25 object recognition, 0.25 task planner, 0.1 gesture recognition. Object recognition has classified all objects on the table correctly, but the robot has not loaded a plan yet. Therefore, EMF generates the list of OAClets presented in Figure 4.18.

$0.25, give(cube(blue))$	$0.25, take(cube(blue))$
$0.25, give(cube(green))$	$0.25, take(cube(green))$
$0.25, give(slat(middle))$	$0.25, take(slat(middle))$
$0.25, give(slat(small))$	$0.25, take(slat(small))$
$0.25, give(slat(small))$	$0.25, take(slat(small))$
$0.25, pointAt(cube(blue))$	$0.25, lookAt(cube(blue))$
$0.25, pointAt(cube(green))$	$0.25, lookAt(cube(green))$
$0.25, pointAt(slat(middle))$	$0.25, lookAt(slat(middle))$
$0.25, pointAt(slat(small))$	$0.25, lookAt(slat(small))$
$0.25, pointAt(slat(small))$	$0.25, lookAt(slat(small))$

$0.25, lookAt(bolt(green))$
$0.25, lookAt(bolt(red))$
$0.25, lookAt(bolt(yellow))$
$0.25, lookAt(cube(red))$
$0.25, lookAt(cube(yellow))$
$0.25, lookAt(nut(orange))$

Figure 4.18: EMF example, step 1.

The relevance scores of all OAClets are equally set to 0.25 since the information about the

objects came from object recognition and no plan was loaded yet. To inrease the readability of the example we are not normalising the relevance scores of the OAClets. Furthermore, we ordered the list of OAClets: in the first double-columned paragraph you can see the OAClets that are related to objects on the robot's workspace, in the second single-columned paragraph you can see the OAClets that are related to objects on the human's workspace. In the next step, the robot loads the plan for the windmill. Because of this, EMF reevaluates all OAClets that contain objects that are part of the windmill and generates new OAClets for abstract objects and planned objects.

$0.75, give(cube(blue))$	$0.50, give(slat(small))$
$0.75, take(cube(blue))$	$0.50, take(slat(small))$
$0.75, pointAt(cube(blue))$	$0.50, pointAt(slat(small))$
$0.75, lookAt(cube(blue))$	$0.50, lookAt(slat(small))$
$0.75, askFor(cube(red))$	$0.50, give(slat(small))$
$0.75, tellAbout(bolt(green))$	$0.50, take(slat(small))$
$0.75, lookAt(bolt(green))$	$0.50, pointAt(slat(small))$
	$0.50, lookAt(slat(small))$
	$0.50, lookAt(bolt(red))$
	$0.50, tellAbout(bolt(red))$
...	

Figure 4.19: EMF example, step 2.

In Figure 4.19, we only show new OAClets or reevaluated OAClets to save space. You can see here that EMF increases all OAClets that are related to Baufix pieces for the windmill by 0.25, the confidence value of task planner. Additionally EMF increases OAClets that are related to pieces of the tower by another 0.25 points, which is part of the windmill and has to be build first. Furthermore, it generates the OAClets $askFor(cube(red))$, $tellAbout(bolt(green))$, and $tellAbout(bolt(red))$ so that the robot can ask the human to put the missing pieces on the table and tell the human that the green bolt and the red bolt are needed for the windmill, which are out of reach of the robot arms.

Since *give* is the action with the highest priority, the robot choses to execute the action $give(cube(blue))$. Thus, object recognition sends the information that the blue cube disappeared. EMF sends this information to the task planner, which checks the current plan, finds the information that the blue cube is part of the tower, and sends information to EMF that the red cube and the green bolt are the pieces which are needed in the next building step. EMF

uses this information to reevaluate the relevance scores of all OAClets that are related to the green bolt or to the red cube and it deletes the OAClets that are related to blue cubes because of the input from object recognition. We show the reevaluated OAClets after these updates in Figure 4.20.

$1.0, askFor(cube(red))$	$0.50, give(slat(small))$
$1.0, tellAbout(bolt(green))$	$0.50, take(slat(small))$
$1.0, lookAt(bolt(green))$	$0.50, pointAt(slat(small))$
	$0.50, lookAt(slat(small))$
	$0.50, give(slat(small))$
	$0.50, take(slat(small))$
	$0.50, pointAt(slat(small))$
	$0.50, lookAt(slat(small))$
	$0.50, lookAt(bolt(red))$
	$0.50, tellAbout(bolt(red))$

...

Figure 4.20: EMF example, step 3.

If the human does not pick up the green bolt, EMF chooses OAClet $askFor(cube(red))$ for execution, because it has the highest relevance score and task priority. This means, the robot asks the human to put the missing red cube on the table. We assume that the human is collaborative, follows the robot's suggestion, and adds a red cube to the human's workspace. Object recognition sends information about the new instantiated object and EMF uses the data to update and generate OAClets. Figure 4.21 shows the reevaluated OAClets after these steps.

EMF generates the new OAClets $tellAbout(cube(red))$ and $lookAt(cube(red))$, which are also immediately reevaluated and increased by 0.25 points. Thus, the robot chooses OAClet $tellAbout(cube(red))$ for execution since it has the highest relevance score and task priority. This means the robot tells the human that the red cube is needed for the windmill in the next step. The human picks up the red cube, object recognition sends information about the disappeared object, EMF sends this information to the task planner, which then sends information that the green bolt is needed to complete the tower. This leads to the list of reevaluated OAClets in Figure 4.22.

To shorten this example, we assume that the human picks up the green bolt at this point. EMF sends the information about the disappearing object to the task planner again, which

$1.25, tellAbout(cube(red))$	$0.50, give(slat(small))$
$1.25, lookAt(cube(red))$	$0.50, take(slat(small))$
$1.0, tellAbout(bolt(green))$	$0.50, pointAt(slat(small))$
$1.0, lookAt(bolt(green))$	$0.50, lookAt(slat(small))$
	$0.50, give(slat(small))$
	$0.50, take(slat(small))$
	$0.50, pointAt(slat(small))$
	$0.50, lookAt(slat(small))$
	$0.50, lookAt(bolt(red))$
	$0.50, tellAbout(bolt(red))$

...

Figure 4.21: EMF example, step 4.

$1.25, tellAbout(bolt(green))$	$0.50, give(slat(small))$
$1.25, lookAt(bolt(green))$	$0.50, take(slat(small))$
	$0.50, pointAt(slat(small))$
	$0.50, lookAt(slat(small))$
	$0.50, give(slat(small))$
	$0.50, take(slat(small))$
	$0.50, pointAt(slat(small))$
	$0.50, lookAt(slat(small))$
	$0.50, lookAt(bolt(red))$
	$0.50, tellAbout(bolt(red))$

...

Figure 4.22: EMF example, step 5.

then sends the information that the two small slats and the red bolt are needed to complete the windmill. Figure 4.23 shows the list of OAClets after these updates.

Since EMF has many OAClets with the same relevance score now, it does not directly choose one of the OAClets for execution in the next step. Thus, the human helps the robot, points to one of the small slats and says "give me this slat". EMF gets the information about the utterances from speech and gesture recognition and uses it to reevaluate the appropriate OAClets, which we show in Figure 4.24.

Please note that through the information from speech recognition, EMF increases the values of all OAClets that are related to small slats and adds 0.4 points to all OAClets with small slats and 0.4 points to all OAClets with action *give*. However, the user also pointed to one of

$0.75, give(slat(small))$	$0.75, give(slat(small))$
$0.75, take(slat(small))$	$0.75, take(slat(small))$
$0.75, pointAt(slat(small))$	$0.75, pointAt(slat(small))$
$0.75, lookAt(slat(small))$	$0.75, lookAt(slat(small))$
$0.75, lookAt(bolt(red))$	$0.75, tellAbout(bolt(red))$
...	

Figure 4.23: EMF example, step 6.

$1.65, give(slat(small))$	$1.15, take(slat(small))$
$1.55, give(slat(small))$	$1.15, pointAt(slat(small))$
$1.25, take(slat(small))$	$1.15, lookAt(slat(small))$
$1.25, pointAt(slat(small))$	$0.75, lookAt(bolt(red))$
$1.25, lookAt(slat(small))$	$0.75, tellAbout(bolt(red))$
...	

Figure 4.24: EMF example, step 7.

the slats, thus EMF also raises the score of the OACIets that related to this specific slat by 0.1 points. Therefore, in the next step EMF chooses the OACIet $give(slat(small))$ with the highest priority for execution. The robot gives the small slat to the human, object recognition sends information about the disappeared object, EMF sends this information to task planner, and so on. The following steps until the windmill is completed are similar to the steps we have already seen. Thus, we conclude the processing example at this point.

Chapter 5

Evaluation

In this chapter, we present the results of three studies, in which we used the two approaches for classical and embodied multimodal fusion that we developed in this thesis. The scenario of all three studies was the JAST construction task that we explained in Chapter 2. This allows us to study different aspects of joint action between human and robot, and to compare the results. The studies were all designed as in-between subject experiments and followed a similar experiment procedure: first, we asked the experiment participants to build the target objects windmill and railway signal with the robot. After that, the subjects had to fill out a user questionnaire that asked them about their subjective opinion of the interaction. In the questionnaire, we used similar questions for all three studies and only adjusted the questions that were not applicable in the context of the particular studies.

Besides the subjective measurements from the user questionnaire, we also collected objective measurements in all three studies, for example the duration it took the participants to build windmill and railway signal or the number of times the users asked the robot to repeat its last utterance. In all three evaluations, we used these objective measurements to make predictions about the subjective ratings of the experiment participants. This way, we for example found that subjects who had to ask the robot for repetition because they did not understand the robot's utterances, significantly more often rate the quality of their interaction with the robot worse than users who did not have to ask for repetition. The idea of aligning objective to subjective measurements goes back to techniques for evaluating spoken language dialogue systems that generally require a large-scale user study, which can be a time-consuming process both for the experimenters and for the experimental subjects. In recent years, techniques have

been introduced that are designed to predict user satisfaction based on more easily measured properties of an interaction such as dialogue length and speech recognition error rate.

In our studies we used a PARADISE-style method to predict user satisfaction from objective measurements. The PARADISE framework (PARAdigm for DIalogue System Evaluation [83]) describes a method for using data to derive a performance function that predicts user satisfaction scores from the results on other, more easily computed measures. PARADISE uses stepwise multiple linear regression to model user satisfaction based on measures representing the performance dimensions of task success, dialogue quality, and dialogue efficiency, and has been applied to a wide range of systems e.g. [81, 57, 60]. The idea behind this process is that the resulting performance function that uses easy to collect objective measurements makes it possible to evaluate HRI systems automatically and to improve the quality of the system with keeping user satisfaction in mind, but without having to conduct extensive user studies in every development step.

The first two studies in this chapter were conducted as part of the JAST project. In these studies, we used the CMF approach. The third study was executed additionally as part of this thesis to show the applicability of the EMF approach. The various aspects of joint action that were researched in the studies were:

- In the first study (Section 5.1), the robot took the role of an instructor. For this, the human had no building plan for windmill and railway signal, such that the robot had to give instructions to the human how to execute the single building steps. In this study, we altered the strategy for describing the plan and also the strategy for generating referring expressions.

- In the second study (Section 5.2), human and robot were equal partners and both had a building plan for the target objects. However, the building plan of the human contained an error, so that the robot had to be able to detect when the human executed that error and to give an explanation to the human, what the error was and how it could be solved. In this study, we again altered the strategy for generating referring expressions.

- In the third study (Section 5.3), human and robot were again equal partners and both had a building plan for windmill and railway signal. Here, we used the EMF approach which enables the robot to show anticipatory behaviour. In this study, we altered the robot behaviour to see if the subjects prefer a robot that directly starts to hand over pieces to the human when building the target objects and only gives instructions when necessary,

or if the users prefer a robot that first gives instructions on which pieces to pick up and then hands over pieces itself.

5.1 Evaluation 1

The first evaluation we are presenting here was published in [34, 35]. In this study human and robot built the two target objects windmill and railway signal together, following the building plan that can be seen in Appendix A.3.1. These building plans consist of named subplans, for example the railway signal is built of a tower (which was called snowman in this study) and an L shape. However, in this study only the robot knew the building plan, so it had to give instructions to the human on how to assemble the Baufix pieces to build the target objects, while also handing over the right pieces that fit to the instructions.

In this study, we varied two aspects in the robot behaviour. Firstly, the robot used different task description strategies when it gave instructions to the human. We implemented a *pre-order task description strategy*, in which the robot first announced the name of the subplan to build and then gave the instructions how to build that subplan; and we implemented a *post-order task description strategy*, in which the robot first gave the instructions and then named the finished subplans and target objects when they were completed. Figure 5.1 shows examples for dialogues in which the robot uses the two task description strategies.

Secondly, the robot used different strategies for generation of *referring expressions*. When humans work together and have to manipulate objects in a shared work space, they use spoken utterances when they refer to these objects. These utterances are called referring expressions. Generation of referring expressions is one of the core tasks in the research area of natural language generation (NLG). The goal here is to generate expressions with which an entity can be uniquely identified from a set of entities. The first referring expression generation strategy that we implemented for the first evaluation was based on the *incremental algorithm* by Dale and Reiter [27], which selects a set of attributes of a target object to single it out from a set of distractor objects. Attributes are selected repeatedly until only the target object remains in the distractor set. However, this algorithm makes no use of context information, but Foster et al. [33] noted a type of multimodal reference which is particularly useful in embodied, task-based contexts: *haptic-ostensive* reference, in which an object is referred to as it is being manipulated by the speaker. Therefore, we implemented a new referring expression generation strategy that makes use of context information, for example the robot said “I give you *this* cube” instead of

Pre-order description strategy, basic reference strategy

SYSTEM First we will build a windmill. Okay?

USER Okay.

SYSTEM To make a windmill, we must make a snowman.

SYSTEM *[picking up and holding out red cube]* To make a snowman, insert the green bolt through the end of the red cube and screw it into the blue cube.

USER *[takes cube, performs action]* Okay.

Post-order description strategy, full reference strategy

SYSTEM First we will build a windmill. Okay?

USER Okay.

SYSTEM *[picking up and holding out red cube]* Insert the green bolt through the end of this red cube and screw it into the blue cube.

USER *[takes cube, performs action]* Okay.

SYSTEM Well done. You have made a snowman.

Figure 5.1: Sample dialogue excerpts showing the various description and reference generation strategies.

"I give you *a* cube" when handing over a Baufix cube from the table to the human. Figure 5.1 shows more examples for referring expressions that were generated with the two strategies, which are underlined in the text.

5.1.1 System Set-up

In this section, we explain how the JAST robot was configured for the first study. Since we already introduced the system in Chapter 2, here we highlight the differences of the system for this study to the final version of the system.

In this study, we used the CMF approach that was configured with the rules that we listed in Section 4.2.4. We had problems with speech recognition in this study, thus the spoken utterances by the experiment participants were typed in by the experimenter. However, the participants were not aware of this fact. Multimodal fusion generated the fusion hypotheses

that we presented in Section 3.2.6 with input from speech, gesture, and object recognition.

Once the dialogue manager had selected a response to the input by multimodal fusion, it sent a high-level specification of the desired response to the output planner, which in turn sent commands to produce appropriate output on each of the individual channels to meet the specification: linguistic content (including appropriate multimodal referring expressions), facial expressions and gaze behaviours of the talking head, and actions of the robot manipulators. Here, output generation used the two strategies for generating referring expressions.

Once the system had described a plan step, the user responded, using a combination of the available input modalities. The user's contribution was processed by the input modules and the CMF approach, a new hypothesis was sent to the dialogue manager, and the interaction continued until the target object had been fully assembled.

5.1.2 Experiment design

Using a between-subjects design, this study compared all of the combinations of the two description strategies with the two reference strategies, measuring the fluency of the resulting dialogue, the users' success at building the required objects and at learning the names of new objects, and the users' subjective reactions to the system.

In this experiment, each subject built the same three objects in collaboration with the JAST system, always in the same order. The first target was the *windmill* (Figure 2.3(a)), which had a sub-component called a *snowman* (Figure 2.3(c)) (which was referred to as *tower* in later iterations of the system). Once the windmill was completed, the system then described how to build an *L shape* (Figure 2.3(d)). Finally, the robot instructed the user on building a *railway signal* (Figure 2.3(b)), which combines an L shape with a snowman.

Before the system explained each target object, the experimenter first configured the workspace with exactly the pieces required to build it. The pieces were always distributed across the two work areas in the same way to ensure that the robot would always hand over the same pieces to each subject. For the windmill, the robot handed over one of the cubes and one of the slats; for the L shape, it handed over both of the required slats; while for the railway signal, it handed over both cubes and both slats. For objects requiring more than one assembly operation (i.e., all but the L shape), the system gave names to all of the intermediate components as they were built. For example, the windmill was always built by first making a snowman and then attaching the slats to the front. When the railway signal was being built, the system always asked the user if they remembered how to build a snowman and an L shape. If they did not

remember, the robot explained again; if they did remember, the robot simply asked them to build another one using the pieces on the table.

For the purpose of this experiment, we constrained the possible user inputs to a limited range of (German) speech commands. Users were informed as part of their instructions that these were the only commands that the system would understand. The allowed speech commands were as follows:

- *hallo* and *guten Tag* (*good day*) in response to greetings from the system;
- *ja* (*yes*) or *nein* (*no*) to answer questions;
- *okay* or *fertig* (*done*) to indicate that a requested assembly operation has been completed; and
- *wie bitte?* (*pardon me?*) to request that the last system utterance be repeated.

5.1.3 Subjects

43 subjects (27 male) took part in this experiment; the results of one additional subject were discarded because the system froze halfway through the interaction. The mean age of the subjects was 24.5, with a minimum of 14 and a maximum of 55. Of the subjects who indicated an area of study, the two most common areas were Informatics (12 subjects) and Mathematics (10). On a scale of 1–5, subjects gave a mean assessment of their knowledge of computers at 3.4, of speech recognition systems at 2.3, and of human-robot systems at 2.0. Subjects were compensated for their participation in the experiment.

5.1.4 Data Acquisition

Independent variables In this study, we manipulated two independent variables, description strategy and reference strategy, each with two different levels. The two possible description strategies were *pre-order* and *post-order*, while the two possible reference strategies were *basic* and *full*. Users were assigned to conditions using a between-subjects design, so that each subject interacted with the system using a single combination of description strategy and reference strategy throughout. Subjects were assigned to each combination of factors in turn, following a Latin-square design. As shown in Table 5.1, 10 subjects interacted with the system that combined the post-order description strategy with the full reference strategy, while each of the other combinations was used by 11 subjects.

Table 5.1: Distribution of subjects.

	Pre-order	**Post-order**
Basic	11	11
Full	11	10

Dependent variables We measured a wide range of dependent values in this study: objective measures based on the logs and recordings of the interactions, as well as subjective measures based on the users' ratings of their experience. The objective metrics fall into the following three classes, based on those used by [83] and [57]: *dialogue efficiency* (the length and timing of the interaction), *dialogue quality* (indications of problems), and *task success*.

The *dialogue efficiency* measures concentrated on the timing of the interaction: the time taken to complete the three construction tasks, the number of system turns required for the complete interaction, and the mean time taken by the system to respond to the user's requests.

We considered four measures of *dialogue quality*. The first two measures looked specifically for signs of problems in the interaction, using data automatically extracted from the logs: the number of times that the user asked the system to repeat its instructions, and the number of times that the user failed to take an object that the robot attempted to hand over. The other two dialogue quality measures were computed based on the video recordings: the number of times that the user looked at the robot, and the percentage of the total interaction that they spent looking at the robot. We considered these gaze-based measures to be measures of dialogue quality since it has previously been shown that, in this sort of task-based interaction where there is a visually salient object, participants tend to look at their partner more often when there is a problem in the interaction [5].

The *task success* measures addressed user success in the two main tasks undertaken in these interactions: assembling the target objects following the robot's instructions, and learning and remembering to make a snowman and an L shape. We measured task success in two ways, corresponding to these two main tasks. The user's success in the overall assembly task was assessed by counting the proportion of target objects that were assembled as intended (i.e., as in Figure 2.3), which was judged based on the video recordings. To test whether the subjects had learned how to build the sub-components that were required more than once (the snowman and the L shape), we recorded whether they said *yes* or *no* when they were asked if they remembered each of these components during the construction of the railway signal.

In addition to the above objective measures, we also gathered a range of subjective measures. After the interaction, the subjects also filled out a user satisfaction questionnaire, which was based on that used in the user evaluation of the COMIC dialogue system [85], with modifications to address specific aspects of the current dialogue system and the experimental manipulations in this study. There were 47 items in total, each of which requested that the user choose their level of agreement with a given statement on a five-point Likert scale. The items were divided into the following categories: *opinion of the robot as a partner* (21 items addressing the ease with which subjects were able to interact with the robot), *instruction quality* (6 items specifically addressing the quality of the assembly instructions given by the robot), *task success* (11 items asking the user to rate how well they felt they performed on the various assembly tasks), and *feelings of the user* (9 items asking users to rate their feelings while using the system). At the end of the questionnaire, subjects were also invited to give free-form comments. Appendix A.4.1 shows the full questionnaire in German and English.

5.1.5 Hypotheses

For each of the experimental manipulations, we had a hypothesis as to its effect on users' interactions:

H1 Subjects will find assembly plans described using the pre-order strategy easier to follow than those described by the post-order strategy.

H2 Subjects will find instructions generated using the full reference strategy easier to follow than instructions generated using the basic reference strategy.

Since we gathered a wide range of subjective and objective measures in this study, we did not make specific predictions as to which specific measure the experimental manipulations will have an effect.

5.1.6 Results

None of the demographic factors (age, gender, area of study, experience with computers) affected any of the results presented here. To determine the impact of the two independent measures on each of the dependent measures, we performed an ANOVA analysis on each class of dependent measures, using both of the independent measures as categorical predictors—in no case was there a significant interaction between the two factors. We list the primary significant factor for each independent measure below, giving the significance values from the ANOVA analysis.

5.1.6.1 Description strategy

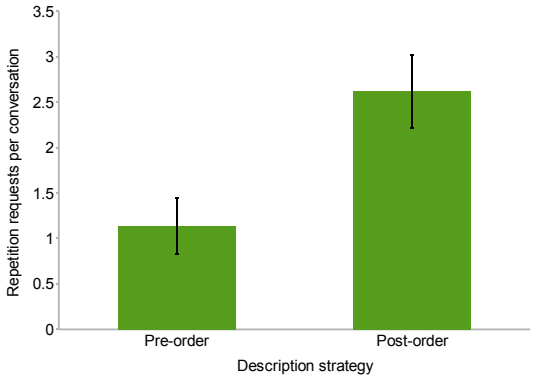


Figure 5.2: Number of repetition requests, divided by description strategy.

The primary difference between the two description strategies (pre-order vs. post-order) was found on one of the dialogue quality measures: the rate at which subjects asked the system to repeat itself during an interaction. As shown in Figure 5.2, subjects in the pre-order condition asked for instructions to be repeated an average of 1.14 times over the course of an interaction, while subjects who used the post-order version of the system asked for repetition 2.62 times on average—that is, more than twice as frequently. The ANOVA analysis indicated that the difference between the two means is significant: $F_{1,39} = 8.28, p = 0.0065$.

5.1.6.2 Reference strategy

- Es war einfach den Anweisungen des Roboters zu folgen.
  It was easy to follow the robot's instructions.
- Der Roboter gab zu viele Anweisungen auf einmal.
  The robot gave too many instructions at once.
- Die Anweisungen des Roboters waren zu ausführlich.
  The robot's instructions were too detailed.

Figure 5.3: Questionnaire items addressing the understandability of the robot's instructions.

The choice of referring expression strategy had no significant effect on any of the objective measures. However, this factor did have an impact on users' responses to a set of items from

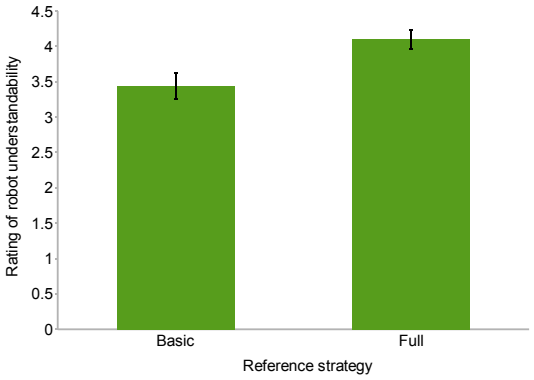


Figure 5.4: Robot understandability rating, divided by reference strategy.

Table 5.2: Dialogue efficiency results.

	Mean (Stdev)	**Min**	**Max**
Length (sec)	305.1 (54.0)	195.2	488.4
System turns	13.4 (1.73)	11	18
Response time (sec)	2.79 (1.13)	1.27	7.21

the questionnaire which specifically addressed the understandability of the robot's instructions. The relevant items are shown in Figure 5.3. The responses of subjects to these three items was different across the two groups. Subjects using the system which employed full referring expressions tended to give higher scores on the first question and lower scores on the second and third, while subjects using the system with basic referring expressions showed the opposite pattern. The mean perceived understandability—i.e., the mean of the responses on these three items, using an inverse scale for the latter two—was 3.44 for the system with basic references and 4.10 for the system with full references; these results are shown in Figure 5.4. The ANOVA analysis indicated that the difference between the two means is significant: $F_{1,39} = 8.32, p = 0.0064$.

5.1.6.3 Dialogue efficiency

The results on the dialogue efficiency measures are shown in Table 5.2. The average subject took 305.1 seconds—that is, just over five minutes—to build all three of the objects, and an average dialogue took 13 system turns to complete. When a user made a request, the mean delay before the beginning of the system response was about three seconds, although for one user this time was more than twice as long. This response delay resulted from two factors. First, preparing long system utterances with several referring expressions (such as the third

Table 5.3: Dialogue quality results.

	Mean (Stdev)	Min	Max
Repetition requests	1.86 (1.79)	0	6
Failed hand-overs	1.07 (1.35)	0	6
Looks at the robot	23.55 (8.21)	14	50
Time looking at robot (%)	27 (8.6)	12	51

and fourth system turns in Figure 5.1) takes some time; second, if a user made a request during a system turn (i.e., a *barge-in* attempt), the system was not able to respond until the current turn was completed. These three measures of efficiency were correlated with each other: the correlation between length and turns was 0.38; between length and response time 0.47; and between turns and response time 0.19 (all $p < 0.0001$).

5.1.6.4 Dialogue quality

Table 5.3 shows the results for the dialogue quality measures: the two indications of problems, and the two measures of the frequency with which the subjects looked at the robot's head. On average, a subject asked for an instruction to be repeated nearly two times per interaction, while failed hand-overs occurred just over once per interaction; however, as can be seen from the standard-deviation values, these measures varied widely across the data. In fact, 18 subjects never failed to take an object from the robot when it was offered, while one subject did so five times and one six times. Similarly, 11 subjects never asked for any repetitions, while five subjects asked for repetitions five or more times. On average, the subjects in this study spent about a quarter of the interaction looking at the robot head, and changed their gaze to the robot 23.5 times over the course of the interaction. Again, there was a wide range of results for both of these measures: 15 subjects looked at the robot fewer than 20 times during the interaction, 20 subjects looked at the robot between 20 to 30 times, while 5 subjects looked at the robot more than 30 times.

The two measures that count problems were mildly correlated with each other ($R^2 = 0.26, p < 0.001$), as were the two measures of looking at the robot ($R^2 = 0.13, p < 0.05$); there was no correlation between the two classes of measures.

5.1.6.5 Task success

Table 5.4 shows the success rate for assembling each object in the sequence. Objects in italics represent sub-components, as follows: the first snowman was constructed as part of the wind-

Table 5.4: Task success results.

Object	Rate	Memory
Snowman	*0.76*	
Windmill	0.55	
L shape	0.90	
L shape	*0.90*	*0.88*
Snowman	*0.86*	*0.70*
Railway signal	0.71	
Overall	**0.72**	**0.79**

mill, while the second formed part of the railway signal; the first L shape was a goal in itself, while the second was also part of the process of building the railway signal. The *Rate* column indicates subjects' overall success at building the relevant component—for example, 55% of the subjects built the windmill correctly, while both of the L shapes were built with 90% accuracy. For the second occurrence of the snowman and the L shape, the *Memory* column indicates the percentage of subjects who claimed to remember how to build it when asked. The *Overall* row at the bottom indicates subjects' overall success rate at building the three main target objects (windmill, L shape, railway signal): on average, a subject built about two of the three objects correctly.

The overall correct-assembly rate was correlated with the overall rate of remembering objects: $R^2 = 0.20, p < 0.005$. However, subjects who said that they did remember how to build a snowman or an L shape the second time around were no more likely to do it correctly than those who said that they did not remember.

5.1.6.6 Paradise Study

We applied a PARADISE-style process to our data. For that, we built models to predict the results of the subjective user satisfaction measures, based on the objective measures. The results indicate that the most significant contributors to user satisfaction were the number of system turns in the dialogues, the users' ability to recall the instructions given by the robot, and the number of times that the user had to ask for instructions to be repeated. The former two measures were positively correlated with user satisfaction, while the latter had a negative impact on user satisfaction; however the correlation in all cases was relatively low.

To determine the relationship among the factors, we employed the procedure used in the PARADISE evaluation framework. The PARADISE model uses stepwise multiple linear regression to predict subjective user satisfaction based on measures representing the performance

Table 5.5: Predictor functions for PARADISE study of first evaluation.

Measure	Function	R^2	Significance
Robot as partner	$3.60 + 0.53 * \mathcal{N}(\text{Turns}) - 0.39 * \mathcal{N}(\text{Rep})$ $-0.18 * \mathcal{N}(\text{Len})$	0.12	Turns: $p < 0.01$, Rep: $p < 0.05$, Length: $p \approx 0.17$
Instruction quality	$3.66 - 0.22 * \mathcal{N}(\text{Rep})$	0.081	Rep: $p < 0.05$
Task success	$4.07 + 0.20 * \mathcal{N}(\text{Mem})$	0.058	Mem: $p \approx 0.07$
Feelings	$3.63 + 0.34 * \mathcal{N}(\text{Turns}) - 0.32 * \mathcal{N}(\text{Rep})$	0.044	Turns: $p \approx 0.06$, Rep: $p \approx 0.08$
Overall	$3.73 - 0.36 * \mathcal{N}(\text{Rep}) + 0.31 * \mathcal{N}(\text{Turns})$	0.062	Rep: $p < 0.05$, Turns: $p \approx 0.06$
Emotion change	$0.07 + 0.14 * \mathcal{N}(\text{Turns}) + 0.11 * \mathcal{N}(\text{Mem})$ $-0.090 * \mathcal{N}(\text{Rep})$	0.20	Turns: $p < 0.05$, Mem: $p < 0.01$, Rep: $p \approx 0.17$

dimensions of task success, dialogue quality, and dialogue efficiency, resulting in a predictor function of the following form:

$$\textit{Satisfaction} \quad = \quad \sum_{i=1}^{n} w_i * \mathcal{N}(\text{measure}_i)$$

The measure_i terms represent the value of each measure, while the $\mathcal{N}$ function transforms each measure into a normal distribution using z-score normalization. Stepwise linear regression produces coefficients (w_i) describing the relative contribution of each predictor to the user satisfaction. If a predictor does not contribute significantly, its w_i value is zero after the stepwise process.

Using stepwise linear regression, we computed a predictor function for each of the subjective measures that we gathered during our study: the mean score for each of the individual user satisfaction categories (Table 5.6), the mean score across the whole questionnaire (the last line of Table 5.6), as well as the difference between the users' emotional states before and after the study (the last line of Table 5.6). We included all of the objective measures as initial predictors.

The resulting predictor functions are shown in Table 5.5. The following abbreviations are used for the factors that occur in the table: *Rep* for the number of repetition requests, *Turns* for the number of system turns, *Len* for the length of the dialogue, and *Mem* for the subjects' memory for the components that were built twice. The R^2 column indicates the percentage of the variance that is explained by the performance function, while the *Significance* column gives significance values for each term in the function.

Although the R^2 values for the predictor functions in Table 5.5 are generally quite low, indicating that the functions do not explain most of the variance in the data, the factors that remain after stepwise regression still provide an indication as to which of the objective measures had an effect on users' opinions of the system. In general, users who had longer interactions with the system (in terms of system turns) and who said that they remembered the robot's instructions tended to give the system higher scores, while users who asked for more instructions to be repeated tended to give it lower scores; for the robot-as-partner questions, the length of the dialogue in seconds also made a slight negative contribution. None of the other factors contributed significantly to any of the predictor functions.

5.1.7 Discussion

That the factors included in Table 5.5 were the most significant contributors to user satisfaction is not surprising. If a user asks for instructions to be repeated, this is a clear indication of a problem in the dialogue; similarly, users who remembered the system's instructions were equally clearly having a relatively successful interaction.

In the current study, increased dialogue length had a positive contribution to user satisfaction; this contrasts with results such as those of [57], who found that increased dialogue length was associated with *decreased* user satisfaction. We propose two possible explanations for this difference. First, the system analysed by [57] was an information-seeking dialogue system, in which efficient access to the information is an important criterion. The current system, on the other hand, has the goal of joint task execution, and pure efficiency is a less compelling measure of dialogue quality in this setting. Second, it is possible that the sheer novelty factor of interacting with a fully-embodied humanoid robot affected people's subjective responses to the system, so that subjects who had longer interactions also enjoyed the experience more. Support for this explanation is provided by the fact that dialogue length was only a significant factor in the more *subjective* parts of the questionnaire, but did not have a significant impact on the users' judgements about instruction quality or task success. Other studies of human-robot dialogue systems have also had similar results: for example, the subjects in the study described by [75] who used a robot that moved while talking reported higher levels of engagement in the interaction, and also tended to have longer conversations with the robot.

While the predictor functions give useful insights into the relative contribution of the objective measures to the subjective user satisfaction, the R^2 values are generally lower than those found in other PARADISE-style evaluations. For example, [82] reported an R^2 value of 0.38,

the values reported by [81] on the training sets ranged from 0.39 to 0.56, [57] reported an R^2 value of 0.71, while the R^2 values reported by [60] for linear regression models similar to those presented here were between 0.22 and 0.57. The low R^2 values from this analysis clearly suggest that, while the factors included in Table 5.5 did affect users' opinions—particularly their opinion of the robot as a partner and the change in their reported emotional state—the users' subjective judgements were also affected by factors other than those captured by the objective measures considered here.

In most of the previous PARADISE-style studies, measures addressing the performance of the automated speech recognition system and other input processing components were included in the models. For example, the factors listed by [60] include several measures of word error rate and of parsing accuracy. However, the scenario that was used in the current study required minimal speech input from the user (see Figure 5.1), so we did not include any such input-processing factors in our models.

5.2 Evaluation 2

In the second study, which was published in [41], we implemented an algorithm for generating referring expressions in the context of errors. Unlike the first evaluation, human and robot were equal partners in this study, which means they both had a building plan for the target objects. However, the building plan of the human had a deliberate error in it. The robot had to detect when the human did an error, and it had to explain to the human, what the error was and how to solve it.

In this study, we once more tested two different strategies for generation of referring expressions: a constant reference strategy that was based on [27], and an adaptive reference strategy that made use of the context of the interaction. For more details on the reference generation algorithms please refer to [41]. In the context of the HRI system, a constant reference strategy is sufficient in that it makes it possible for the robot's partner to know which item is needed. On the other hand, while the varied forms produced by the more complex mechanism can increase the naturalness of the system output, they may actually be insufficient if they are not used in appropriate current circumstances—for example, "this cube" is not a particularly helpful reference if a user has no way to tell which "this" is. As a consequence, the system for generating such references must be sensitive to the current state of joint actions and—in effect—of joint attention. The difference between the two systems is a test of the adaptive version's ability to

adjust expressions to pertinent circumstances. It is known that people respond well to reduced expressions like "this cube" or "it" when another person uses them appropriately [8]. With this study, we wanted to find out if the robot system can also achieve the benefits that situated reference could provide.

5.2.1 System Set-up

In this study, we used the full JAST robot system as it was described in Section 2.1. Especially, here we integrated a module called goal inference, a component that is based on dynamic neural fields [30, 11, 10], which selects the robot's next actions based on the human user's actions and utterances. Given a particular assembly plan and the knowledge of which objects the user has picked up, this module can determine when the user has made an error.

For this study, we used the CMF approach to process user input. However, the multimodal fusion component also played a role as communicator between goal inference and dialogue manager. For that, the fusion module processed the human utterances with the CMF approach to generate a fusion hypothesis, but also translated and sent the input to goal inference. When goal inference finished processing the input, it sent back robot control instructions on which action the robot should execute next. Multimodal fusion then combined the instructions from goal inference with the own fusion hypothesis and sent this data to the dialogue manager.

5.2.2 Experiment Design

This study used a between-subjects design with one independent variable: each subject interacted either with the system that used the constant strategy to generate referring expressions (19 subjects), or else with the system that used the adaptive strategy (22 subjects).[1]

Each subject built the two target objects in collaboration with the system—windmill and railway signal. For both target objects, the user was given a building plan on paper. To induce an error, both of the plans given to the subjects instructed them to use an incorrect piece: a yellow cube instead of a red cube for the windmill, and a long (seven-hole) slat instead of a medium (five-hole) slat for the railway signal. The subjects were told that the plan contained an error and that the robot would correct them when necessary, but did not know the nature of the error.

[1]The results of an additional three subjects in the constant-reference condition could not be analysed due to technical difficulties.

When the human picked up or requested an incorrect piece during the interaction, the system detected the error and explained to the human what to do in order to assemble the target object correctly. When the robot explained the error and when it handed over the pieces, it used referring expressions that were generated using the constant strategy for half of the subjects, and the adaptive strategy for the other half of the subjects.

The participants stood in front of the table facing the robot, equipped with a headset microphone for speech recognition. The pieces required for the target object—plus a set of additional pieces in order to make the reference task more complex—were placed on the table, using the same layout for every participant. The layout was chosen to ensure that there would be points in the interaction where the subjects had to ask the robot for building pieces from the robot's workspace, as well as situations in which the robot automatically handed over the pieces. Appendix A.2 shows the table layouts for windmill and railway signal. Along with the building plan mentioned above, the subjects were given a table with the names of the pieces they could build the objects with.

5.2.3 Subjects

41 subjects (33 male) took part in this experiment. The mean age of the subjects was 24.5, with a minimum of 19 and a maximum of 42. Of the subjects who indicated an area of study, the two most common areas were Mathematics (14 subjects) and Informatics (also 14 subjects). On a scale of 1 to 5, subjects gave a mean assessment of their knowledge of computers at 4.1, of speech recognition systems at 2.0, and of human-robot systems at 1.7. Subjects were compensated for their participation in the experiment.

5.2.4 Data Acquisition

At the end of a trial, the subjects responded to a usability questionnaire consisting of 39 items, which fell into four main categories: *intelligence of the robot* (13 items), *task ease and task success* (12 items), *feelings of the user* (8 items), and *conversation quality* (6 items). The items on the questionnaire were based on those used in the first user evaluation, but were adapted for the scenario and research questions of the current study. The questionnaire was presented using software that let the subjects choose values between 1 and 100 with a slider. We show the full questionnaire in Appendix A.4.2. In addition to the questionnaire, the trials were also video-taped, and the system log files from all trials were kept for further analysis.

5.2.5 Hypotheses

We made no specific prediction regarding the effect of reference strategy on any of the objective measures: based on the results of the first evaluation, there is no reason to expect an effect either way. Note that—as mentioned above—if the adaptive version makes incorrect choices, that may have a negative impact on users' ability to understand the system's generated references. For this reason, even a finding of no objective difference would demonstrate that the adaptive references did not harm the users' ability to interact with the system, as long as it was accompanied by the predicted improvement in subjective judgements.

5.2.6 Results

We analysed the data resulting from this study in three different ways. First, the subjects' responses to the questionnaire items were compared to determine if there was a difference between the responses given by the two groups. A range of summary objective measures were also gathered from the log files and videos—these included the duration of the interaction measured both in seconds and in system turns, the subjects' success at building each of the target objects, the number of times that the robot had to explain the construction plan to the user, and the number of times that the users asked the system to repeat its instructions. Finally, we compared the results on the subjective and objective measures to determine which of the objective factors had the largest influence on subjective user satisfaction.

5.2.6.1 Subjective Measures

The subjects in this study gave a generally positive assessment of their interactions with the system on the questionnaire—with a mean overall satisfaction score of 72.0 out of 100—and rated the perceived intelligence of the robot particularly highly (overall mean of 76.8). Table 5.6 shows the mean results from the two groups of subjects for each category on the user satisfaction questionnaire, in all cases on a scale from 0–100 (with the scores for negatively-posed questions inverted).

To test the effect of reference strategy on the usability questionnaire responses, we performed a Mann-Whitney test comparing the distribution of responses from the two groups of subjects on the overall results, as well as on each sub-category of questions. For most categories, there was no significant difference between the responses of the two groups, with p values ranging from 0.19 to 0.69 (as shown in Table 5.6). The only category where a significant difference was found was on the questionnaire items that asked the subjects to assess the robot's quality as a

Table 5.6: Overall usability results of second evaluation.

	Constant	**Adaptive**	**M-W**
Intell.	79.0 (15.6)	74.9 (12.7)	$p = 0.19$, n.s.
Task	72.7 (10.4)	71.1 (8.3)	$p = 0.69$, n.s.
Feeling	66.9 (15.9)	66.8 (14.2)	$p = 0.51$, n.s.
Conv.	66.1 (13.6)	75.2 (10.7)	$p = 0.036$, sig.
Overall	72.1 (11.2)	71.8 (9.1)	$p = 0.68$, n.s.

Table 5.7: User responses to questionnaire items addressing the robot's quality as a conversational partner. The questions were posed in German; the table also shows the English translation.

Statement	**Constant**	**Adaptive**	**M-W**
Ich fand den Roboter schwierig zu verstehen. *I found the robot difficult to understand.*	21.5 (26.3)	14.5 (15.9)	$p = 0.58$
Der Roboter hat nicht verstanden was ich gesagt habe. *The robot didn't understand what I said.*	18.9 (29.6)	21.1 (31.0)	$p = 0.60$
Manchmal wenn der Roboter mit mir gesprochen hat, konnte ich nicht verstehen, was er meinte. *Sometimes, when the robot talked to me,* *I didn't understand what it meant.*	24.8 (28.7)	16.6 (20.1)	$p = 0.47$
Wenn der Roboter mich nicht verstand, dann war mir klar, wie ich reagieren musste. *When the robot did not understand me,* *it was clear what I had to do.*	34.7 (23.7)	50.4 (28.8)	$p = 0.091$
Ich habe nicht verstanden was der Roboter gesagt hat. *I didn't understand what the robot said.*	9.9 (16.1)	8.9 (11.0)	$p = 0.81$
Ich wusste zu jedem Zeitpunkt der Unterhaltung, was ich machen oder sagen konnte. *At each point in the conversation,* *I knew what I could do or say.*	37.6 (25.1)	61.6 (29.6)	$p = 0.012$

conversational partner; for those items, the mean score from subjects who heard the adaptive references was significantly higher ($p < 0.05$) than the mean score from the subjects who heard references generated by the constant reference module. Of the six questions that were related to the conversation quality, the most significant impact was on the two questions which assessed the subjects' understanding of what they were able to do at various points during the interaction.

Table 5.7 shows the six conversation quality questions, along with the mean responses from subjects from the two groups and the standard deviation of each. The final column of the table shows the p value from a Mann-Whitney test (two-tailed) comparing the distribution of

responses from the two groups of subjects. On almost all of these questions, the responses of the subjects who experienced the adaptive references were more positive than those who experienced constant references, although not in general significantly so: the adaptive reference subjects gave higher responses to questions 4 and 6, and lower responses to all of the other (negatively-posed) questions except for question 2. The most pronounced difference was on the two questions which assessed the subjects' understanding of what they were able to do at various points during the interaction (questions 4 and 6).

5.2.6.2 Objective Measures

Based on the log files and video recordings, we computed a range of objective measures. These measures were divided into three classes, again based on those used in the PARADISE dialogue system evaluation framework [81]:

- Two **dialogue efficiency** measures: the mean duration of the interaction as measured both in seconds and in system turns,
- Two **dialogue quality** measures: the number of times that the robot gave explanations, and the number of times that the user asked for instructions to be repeated, and
- One **task success** measure: how many of the (two) target objects were constructed as intended (i.e., as shown in Figure 2.3).

For each of these measures, we tested whether the difference in reference strategy had a significant effect, again via a Mann-Whitney test. Table 5.8 illustrates the results on these objective measures, divided by the reference strategy.

The results from the two groups of subjects were very similar on all of these measures: on average, the experiment took 404 seconds (nearly seven minutes) to complete with the constant strategy and 410 seconds with the adaptive, the mean number of system turns was close to 30 in both cases, just over one-quarter of all subjects asked for instructions to be repeated, the robot gave just over two explanations per trial, and about three-quarters of all target objects (i.e. 1.5 out of 2) were correctly built. The Mann-Whitney test confirms that none of the differences between the two groups even came close to significance on any of the objective measures.

Table 5.8: Objective results (all differences n.s.).

Measure	Constant	Adaptive	M-W
Duration (s.)	404.3 (62.8)	410.5 (94.6)	$p = 0.90$
Duration (turns)	29.8 (5.02)	31.2 (5.57)	$p = 0.44$
Rep requests	0.26 (0.45)	0.32 (0.78)	$p = 0.68$
Explanations	2.21 (0.63)	2.41 (0.80)	$p = 0.44$
Successful trials	1.58 (0.61)	1.55 (0.74)	$p = 0.93$

Table 5.9: Predictor functions for PARADISE study of second evaluation.

Measure	Function	R^2	Significance
Robot as partner	$3.60 + 0.53 * \mathcal{N}(\text{Turns}) - 0.39 * \mathcal{N}(\text{Rep})$ $-0.18 * \mathcal{N}(\text{Len})$	0.12	Turns: $p < 0.01$, Rep: $p < 0.05$, Length: $p \approx 0.17$
Instruction quality	$3.66 - 0.22 * \mathcal{N}(\text{Rep})$	0.081	Rep: $p < 0.05$
Task success	$4.07 + 0.20 * \mathcal{N}(\text{Mem})$	0.058	Mem: $p \approx 0.07$
Feelings	$3.63 + 0.34 * \mathcal{N}(\text{Turns}) - 0.32 * \mathcal{N}(\text{Rep})$	0.044	Turns: $p \approx 0.06$, Rep: $p \approx 0.08$
Overall	$3.73 - 0.36 * \mathcal{N}(\text{Rep}) + 0.31 * \mathcal{N}(\text{Turns})$	0.062	Rep: $p < 0.05$, Turns: $p \approx 0.06$
Emotion change	$0.07 + 0.14 * \mathcal{N}(\text{Turns}) + 0.11 * \mathcal{N}(\text{Mem})$ $-0.090 * \mathcal{N}(\text{Rep})$	0.20	Turns: $p < 0.05$, Mem: $p < 0.01$, Rep: $p \approx 0.17$

5.2.6.3 Paradise Study

In the preceding sections, we presented results on a number of objective and subjective measures. While the subjects generally rated their experience of using the system positively, there was some degree of variation, most of which could not be attributed to the difference in reference strategy. Also, the results on the objective measures varied widely across the subjects, but again were not generally affected by the reference strategy. In this section, we examine the relationship between these two classes of measures in order to determine which of the objective measures had the largest effect on users' subjective reactions to the HRI system.

As in the first evaluation, we used multiple linear regression to compute predictor functions. Table 5.9 shows these functions that were derived for each of the classes of subjective measures in this study, using all of the objective measures from Table 5.8 as initial factors. The R^2 column indicates the percentage of the variance in the target measure that is explained by the predictor function, while the *Significance* column gives significance values for each term in the function.

In general, the two factors with the biggest influence on user satisfaction were the number of repetition requests (which had a uniformly negative effect on user satisfaction), and the number of target objects correctly built by the user (which generally had a positive effect). Aside from the questions on user feelings, the R^2 values are generally in line with those found in previous PARADISE evaluations of other dialogue systems [81, 57], and in fact are much higher than those found in the first evaluation.

5.2.7 Discussion

The subjective responses on the relevant items from the usability questionnaire suggest that the subjects perceived the robot to be a better conversational partner when it used contextually varied, situationally-appropriate referring expressions than when it always used a baseline, constant strategy; this supports the main prediction for this study. The result also agrees with the findings of the first evaluation. These studies together support the current effort in the natural-language generation community to devise more sophisticated reference generation algorithms.

On the other hand, there was no significant difference between the two groups on any of the objective measures: the dialogue efficiency, dialogue quality, and task success were nearly identical across the two groups of subjects. A detailed analysis of the subjects' gaze and object manipulation behaviour immediately after various forms of generated references from the robot also failed to find any significant differences between the various reference types. These overall results are not particularly surprising: studies of human-human dialogue in a similar joint construction task [9] have demonstrated that the collaborators preserve quality of construction in all cases, though circumstances may dictate what strategies they use to do this. Combined with the subjective findings, this lack of an objective effect suggests that the references generated by the adaptive strategy were both sufficient and more natural than those generated by the constant strategy.

The analysis of the relationship between the subjective and objective measures analysis has also confirmed and extended the findings from the first evaluation. In that study, the main contributors to user satisfaction were user repetition requests (negative), task success, and dialogue length (both positive). In the current study, the primary factors were similar, although dialogue length was less prominent as a factor and task success was more prominent. These findings are generally intuitive: subjects who are able to complete the joint construction task are clearly having more successful interactions than those who are not able to complete

the task, while subjects who need to ask for instructions to be repeated are equally clearly not having successful interactions. The findings add evidence that, in this sort of task-based, embodied dialogue system, users enjoy the experience more when they are able to complete the task successfully and are able to understand the spoken contributions of their partner, and also suggest that designers should concentrate on these aspects of the interaction when designing the system.

5.3 Evaluation 3

In the third study, we tested the EMF approach for the first time on the JAST robot to see if the method not only works in theory but also on a real system with naïve subjects that have not seen the robot before. Since we used EMF here, we were able to easily implement two different robot behaviours to compare the user ratings of these behaviours to each other. We call these behaviours *instructive behaviour*, because in this setting the robot first instructs the user which pieces to pick up and after that hands over pieces itself, and *proactive behaviour*, because in this setting the robot first hands over pieces itself and then gives instructions to the user.

In a way, this study can be seen as a comparison of the first two evaluations, since the robot takes over different roles: on the one hand, the robot plays the role of an instructor as in the first evaluation and on the other hand, the robot plays the role of an equal partner as in the second evaluation. Therefore, in this study we were mainly interested to find out if the participants generally preferred one of the two robot behaviours and also if the users adapt their role to that of the robot.

5.3.1 System Set-up

In this study, we used the EMF approach as we described it in Sections 3.3 and 4.3. However, we did not use the full spectrum of the capabilities of the JAST robot since they where not needed for this experiment. The robot had to generate and evaluate OAClets with the following actions:

- *give*, the robot hands over an Baufix piece from its own workspace to the human.
- *tellAbout*, the robot tells the human to pick up a piece from the human's workspace, because it fits a building step of the currently loaded plan.

- *askFor*, the robot asks the human to put a certain Baufix piece on the table, because it is needed for the current plan and the robot cannot detect it with its object recognition.
- *tellBuild*, the robot asks the human to build one of the substeps of the currently loaded plan. For example the windmill has one substep, the tower. When the robot registers that the human picked up all pieces needed to build the tower, it asks the human to build it and put it on the table.
- *thankFor*, the robot thanks the human for the piece the human puts on the table. This action is added in an OAClet to the OAClet container when the robot executes one of the actions *askFor* or *tellBuild* so that the robots thanks the human, when the needed piece appears on the table.

As we already started to explain in the introduction, in this experiment, the robot showed two different behaviours: in the *instructive behaviour* setting, the robot preferably executed action *tellAbout* and thus gave instructions to the user first and then handed over pieces to the experiment participants. In the *proactive behaviour*, the robot was configured so that it preferably executed the action *give* and handed over Baufix pieces to the human.

The implementation of the EMF approach made it easy to realise these two different robot behaviours through reconfiguration of the action priority list that we introduced in Section 4.3.4. For the instructive behaviour we configured the task hierarchy with

$$tellAbout > give > askFor > tellBuild > thankFor$$

In comparison to that we configured the task hierarchy for the proactive behaviour with

$$give > tellAbout > askFor > tellBuild > thankFor$$

As you can see, the only difference in the two hierarchies are the first two elements. However, this small change already resulted in a notably different behaviour of the robot, which we also show in the dialogue example in the next section.

5.3.2 Experiment Design

This study used a between-subjects design with one independent variable: each subject interacted either with a system that used the proactive robot behaviour setting, or else with a system that used the instructive robot behaviour. Each subject built the two target objects

in collaboration with the system, always in the same order, first the windmill, after that the railway signal. For both target objects, the user was given a building plan on paper.

The participants stood in front of the table facing the robot, equipped with a headset microphone for speech recognition. The pieces required for the target object were placed on the table, using the same layout for every participant. The layout was chosen to ensure that there would be enough similar Baufix pieces on both sides of the table for every subplan of the target objects so that the robot could either perform the action *give* and handover an object from its side of the table or the action *tellAbout* and instruct the users to pick up an object from their side of the table. For example, for the tower there was a red cube in both table areas, so that the robot could either hand over the cube from its side or instruct the subjects to pick up the cube from their side. Appendix A.2 shows the table layouts for windmill and railway signal. Along with the building plan mentioned above, the subjects were given a table with the names of the pieces they could build the objects with.

Additionally, subjects got instructions that they could speak with the robot. In this study, the experiment participants could either ask the robot for one of the pieces in the robot's workspace by giving a direct order, for example by saying "gib mir einen blauen Würfel" ("give me a blue cube"), or they could ask the robot to repeat its last utterance by saying "wie bitte?", ("pardon me?"). Figure 5.5 shows two example dialogues between human and robot in the proactive and instructive behaviour settings, which we extracted from two of the experiment videos. From the dialogues you can clearly see that the robot gave more instructions in the instructive behaviour setting.

5.3.3 Subjects

40 subjects (27 male) took part in this experiment. The mean age of the subjects was 27.2, with a minimum of 17 and a maximum of 59. Of the subjects who indicated an area of study, the two most common areas were Mathematics (11 subjects) and Informatics (8 subjects). On a scale of 1 to 5, subjects gave a mean assessment of their knowledge of computers at 3.7, of speech recognition systems at 1.9, and of human-robot systems at 1.6. For their participation in the experiment, the subjects got the chance to win a voucher for an online shop.

5.3.4 Data Acquisition

At the end of a trial, the subjects responded to a usability questionnaire consisting of 29 items, which fell into four main categories: *feelings of the user* (10 items), *intelligence of the robot* (7

Instructive behaviour setting

SYSTEM Nehmen sie bitte die mittlere Leiste. *(Please pick up a medium slat.)*

USER *[picks up medium slat]*

SYSTEM Nehmen sie bitte die Mutter. *(Please pick up a nut.)*

USER *[picks up nut]*

SYSTEM Ich gebe ihnen eine gelbe Schraube. *(I will give you a yellow bolt.)*

SYSTEM *[picks up yellow bolt and hands it over to human]*

USER Gib mir die kleine Leiste. *(Give me a small slat.)*

SYSTEM Ich gebe ihnen eine kleine Leiste. *(I will give you a small slat.)*

SYSTEM *[picks up small slat and hands it over to human]*

SYSTEM Sie haben alle Teile für einen Buchstaben genommen. Können sie ihn bitte zusammenbauen und auf die Arbeitsfläche legen. *(You picked up all pieces for an l shape. Could assemble it and place it on the workspace, please.)*

Proactive behaviour setting

SYSTEM Ich gebe ihnen eine kleine Leiste. *(I will give you a small slat.)*

SYSTEM *[picks up small slat and hands it over to human]*

USER *[picks up medium slat]*

USER *[picks up nut]*

SYSTEM Ich gebe ihnen eine gelbe Schraube. *(I will give you a yellow bolt.)*

SYSTEM *[picks up yellow bolt and hands it over to human]*

SYSTEM Sie haben alle Teile für einen Buchstaben genommen. Können sie ihn bitte zusammenbauen und auf die Arbeitsfläche legen. *(You picked up all pieces for an l shape. Could assemble it and place it on the workspace, please.)*

Figure 5.5: Sample dialogue excerpts showing the proactive and instructive robot behaviour.

items), *robot behaviour* (6 items), and *task success* (6 items). The items on the questionnaire were based on those used in the previous user evaluations, but were adapted for the scenario and

research questions of the current study. The questionnaire was presented using software that let the subjects choose values between 1 and 100 with a slider. We show the full questionnaire in Appendix A.4.3. In addition to the questionnaire, the trials were also video-taped, and the system log files from all trials were kept for further analysis.

5.3.5 Hypotheses

Since we are comparing different robot behaviours in this study, we had two hypotheses how these behaviours affect user ratings:

H1 Subjects who work with the robot in the proactive behaviour setting, generally asses their interaction with the robot more positive.

H2 Subjects who work with the robot in the proactive behaviour setting also display a more proactive behaviour, while subjects using the instructive robot will take a more passive role in the interaction.

Since we gathered a wide range of subjective and objective measures in this study, we did not make specific predictions as to which specific measure the experimental manipulations will have an effect.

5.3.6 Results

In this study, we analysed the collected data in several ways. First, we compared the subjective answers of the experiment participants to the user questionnaire to find out if there are any significant differences between the answers of the group that worked with the proactive robot and the group that worked with the instructive robot. Second, we compared the objective measurements that we took from the system logs and the videos to find differences between the two groups. Third, we made a PARADISE-style calculation to find which of the objective measurements could potentially predict the subjective answers by the experiment participants.

5.3.6.1 Subjective Measurements

We applied a Mann-Whitney test on the answers to the user questionnaire to find if the different robot behaviours had a significant effect on the ratings by the two participant groups. Generally, subjects gave a positive feedback of an average 83 of 100 points on the questions that asked them if they liked working with the robot. However, the participants rated the robot's intelligence

with only 56.35 points, but the standard deviation was quite high for this question with 26.16 points. There was no significant difference in these questions between the two groups.

We found significant differences (p-value < 0.05) in the ratings for 4 of the 29 statements of the user questionnaire, which are displayed in Table 5.10. Here, one of these statements belongs to the category *feelings of the user* ("I found the robot easy to use."), two statements of the statements belongs to category *intelligence of the robot* ("I knew what I could say or do at each point in the conversation.", "It was clear what to do when the robot did not understand me."), and one statement belongs to category *robot behaviour* ("The robot gave too many instructions.").

Table 5.10: Statements with significant differences between user groups of user questionnaire for third evaluation.

Statement	**Proactive**	**Instructive**	**M-W**
Ich fand, der Roboter war einfach zu benutzen. *I found the robot easy to use.*	83.80 (12.81)	90.80 (13.03)	$p \approx 0.043$
Ich habe zu jedem Zeitpunkt in der Konversation gewusst was ich tun oder sagen kann. *I knew what I could say or do at each point in the conversation.*	71.05 (30.32)	90,10 (12.04)	$p \approx 0.038$
Wenn der Roboter mich nicht verstand, dann war mir klar was ich tun musste. *It was clear what to do when the robot did not understand me.*	70.65 (21.46)	57.33 (15.26)	$p \approx 0.034$
Der Roboter gab zu viele Anweisungen. *The robot gave too many instructions.*	33.95 (28.21)	16.71 (21.95)	$p \approx 0.026$

5.3.6.2 Objective Measures

We collected a set of objective measurements from the automatically generated system log files and from annotations of the videos we took during the experiments. All in all, we had four different objective measurements:

- the number of verbal utterances by the subjects, which is the number of times the users asked the robot for a certain Baufix piece or to repeat its last utterance,
- the number of instructions the robot gave to the subjects, i.e. only the instructions in which the robot told the human which piece to pick up next from the workspace,

- the overall duration the subjects needed to build windmill and railway signal, and
- the number of times the subjects picked up an Baufix piece from their side of the table, where the robot did not instruct them to pick up the object.

We were able to gather the first two measurements from the system log files; we annotated the videos of the experiment participants with Anvil [49] to collect the the remaining two measurements. Not all subjects agreed that we videotaped them, thus we only have video data for 32 of the 40 subjects, 15 videos of participants who used the proactive robot and 17 videos of participants who used the instructive robot.

We show the results of the objective measurements in Table 5.11. We computed if there is a significant difference between the two user groups, again via a Mann-Whitney test. We found a significant difference for the number of robot instructions, which is not surprising, but shows that the robot gave significantly more instructions to the user in the instructive behaviour setting. Furthermore, users who worked with the robot in the proactive behaviour setting significantly picked up more Baufix pieces without getting instructions from the robot to do so.

Table 5.11: Objective results for third evaluation.

Measure	**Instructive**	**Proactive**	**M-W**
No. user utterances	1.65 (1.69)	1.25 (1.94)	$p \approx 0.33$
No. robot instructions	10.3 (1.49)	4.60 (2.28)	$p < 0.01$
Duration (s.)	265.86 (46.22)	258.80 (51.32)	$p \approx 0.82$
No. anticipative user actions	0.76 (0.90)	4.80 (1.97)	$p < 0.01$

5.3.6.3 Paradise Study

To complete the result analysis of this study, we made a PARADISE study to compute if the objective measurements we collected in the third evaluation could predict the subjective statements of the user questionnaire. Table 5.12 shows the predictor functions that we calculated using stepwise multiple linear regression. For the calculation we used all four objective measurements, which are abbreviated in the table with *Dur* (duration to build both target objects), *Pickup* (number of anticipatory pick up actions by experiment participant), *Utt* (number of utterances by experiment participant), and *Inst* (number of robot instructions).

The calculated predictor functions show that all of the objective measurements influence user satisfaction in one way or the other:

Table 5.12: Predictor functions for PARADISE study of third evaluation.

Measure	Function	R^2	Significance
Feelings	$324.68 + 0.77 * \mathcal{N}(\text{Dur}) + 27.35 * \mathcal{N}(\text{Pickup})$ $-40.26 * \mathcal{N}(\text{Utt}) + 20.96 * \mathcal{N}(\text{Inst})$	0.27	Dur: $p \approx 0.16$ Pickup: $p \approx 0.16$ Utt: $p < 0.01$ Inst: $p \approx 0.13$
Intelligence	$405.02 + 0.58 * \mathcal{N}(\text{Dur}) - 18.70 * \mathcal{N}(\text{Utt})$	0.15	Dur: $p \approx 0.10$ Utt: $p < 0.05$
Behaviour	$487.33 - 10.96 * \mathcal{N}(\text{Pickup})$	0.12	Pickup: $p \approx 0.05$
Task success	$447.74 + 0.40 * \mathcal{N}(\text{Dur}) - 17.54 * \mathcal{N}(\text{Utt})$	0.23	Dur: $p \approx 0.10$ Utt: $p < 0.01$

- The number of user utterances has a strongly negative influence on the three categories *feelings of the user* (abbreviated with *Feelings* in table), *intelligence of the robot* (abbr. *Intelligence*), and *task success*. However, the duration to build both target objects had a slight positive effect in the same three categories.
- The number of anticipatory pick up actions by the user had an positive influence on category *feelings of the user* and a negative influence on category *robot behaviour*.
- The number of robot instructions had a strong positive influence on the category *feelings of the user*, but not on the other categories.

The R^2 values of this study are in the same range as the values of the previous studies and thus confirm the findings of the first two evaluations. However, the values are not as high as those reported in [81, 57].

5.3.7 Discussion

The results of this study show an interesting correlation: we expected that the experiment participants will prefer the proactive robot over the instructive robot. However, the data suggests that the users accept both robot roles and simply take the counterpart in the interaction with the robot. This can be seen from the significant answers to the statements of the user questionnaire, where the users that worked with the proactive robot answered more positive to the statement "I knew what I could say or do at each point in the conversation". This indicates that the subjects showed a more proactive behaviour themselves and followed the building plan more when the robot gave less instructions. In contrast to that, the users who worked with the instructive robot rated the statement "The robot gave too many instructions" lower

than the users from the other group, which means that they wanted to hear more instructions by the robot, even though the robot already gave them significantly more instructions. This supports our claim that the users took the counterpart role to the robot. One of the objective measurements also supports our opinion: users who worked with the proactive robot also showed a proactive behaviour and executed anticipatory pick up actions significantly more often than users of the other group. These results are in line with research from cognitive psychology and cognitive neuroscience. Sebanz et al. review in [72] a set of studies from these fields, which also prove that humans attune their actions when working together.

The results of the calculated PARADISE predictor functions are not very surprising. However, it is interesting to note that the number of anticipatory pick up actions had a positive influence on the statements in the category *feelings of the user* and a negative effect on the category *robot behaviour*. In our opinion, this shows that the users prefer to show their own initiative. That means that they would rather work with the proactive robot, which supports hypothesis H1 that we stated in Section 5.3.5. The negative effect of this measurements on the assessment of the robot's behaviour can in our opinion be explained with robot errors during the interaction: when the robot made an error and for example gave the wrong instructions to the user or stopped working (which could happen sometimes during the experiments because of wrongly recognised Baufix pieces), the users had to pick up the pieces to finish building the target objects without getting instructions by the robot.

The number of user utterances also had a negative influence on the user satisfaction. This can be easily explained: in this experiment the EMF approach was configured so that the users did not have to speak with the robot, as long as the system performed well. The users only had to talk to the robot when they either did not understand the robot's utterances and had to ask for repetition or they needed to give a direct command to the robot to ask for a piece of the robot's workspace, which almost only happened when the robot made an error. Thus, the number of user utterances is a clear indicator for problems during the experiment. This result confirms the findings of the first two evaluations, where the number of repetition requests also had a negative influence on user satisfaction.

Chapter 6

Conclusion

In the last chapter of this thesis, we take a step back to get an overview of the presented work and to discuss its contribution to the field of human-robot interaction. In Section 6.1, we summarise this thesis and list the main contributions of our work. After that, in Section 6.2, we compare the properties of the two proposed approaches for multimodal fusion and discuss their applicability for human-robot interaction. Finally, in Section 6.3 we provide an outlook on future development of multimodal fusion for human-robot interaction.

6.1 Summary

In this thesis, we compared two approaches for multimodal fusion for human-robot interaction (HRI). A robot that should interact with a human, needs multimodal fusion to interpret and merge information from different modalities that recognise human verbal and non-verbal utterances, recognise objects in the robot's environment, or store information about the robot's task.

In Section 2, we presented related work, which originates from diverse research fields, including multimodal dialogue systems, cognitive and robot architectures, human-robot interaction, spoken language processing, and embodiment. The literature review raised two questions: (1) multimodal dialogue systems use methods from classical artificial intelligence (AI), for example rule-based processing, to integrate events from different modalities to a unified representation. This approach works well in structured environments, as for example in screen-based information systems that combine information from speech recognition and touch-based gestures. Can multimodal dialogue systems also be used for human-robot interaction? (2) The research area embodiment takes a contrary position to the notion of AI that intelligence can be implemented

detached from a body. The basic idea of embodiment is that intelligence cannot exist without a body and that an embodied agent can exploit its environment to synthesise intelligent behaviour. In the last decade, robots that implement their sensorimotor coordination based on this paradigm have shown substantial progress in moving in unstructured environments and handling objects. However, the question remains if high-level AI problems, such as multimodal fusion and natural language processing, can be implemented with methods from embodiment?

Based on these two observations, we developed two approaches for multimodal fusion, one approach based on methods from classical AI and another approach based on embodiment, which we introduced theoretically in Chapter 3:

- The *classical multimodal fusion* (CMF) implements a human-centred view on a robot's input data and is based on methods from artificial intelligence. In CMF, the fusion process is triggered by a verbal utterance of the human. When the human talks to the robot, CMF processes the spoken utterance in several steps: first, CMF analyses the recognised string that it gets from the robot's speech recognition module in a parsing step. For that, it uses a combinatory categorial grammar [77], which analyses the syntactic structure of the human utterance and generates a logical formula that represents its semantic content. CMF then uses this logical representation in a reference resolution step to ground the objects that the human has talked about to objects that the human has pointed to or to objects that the robot can see in its environment. For this, CMF uses a rule engine, which uses information from gesture and object recognition to generate a unified representation, called fusion hypothesis, which represents the information from the speech, gesture, and object recognition channels. CMF finally sends this information to a dialogue manager, which calculates the robot's next actions and multimodal output to the human.

- The *embodied multimodal fusion* (EMF) implements a robot-centred view on the robot's input channels. The data representation in EMF is based on the notion that for an embodied agent actions and objects are inseparably intertwined. Thus, the objects in the robot's environment define the actions that it can execute with these objects and the actions define the purpose of the objects. This connection between objects and actions is called object-action complex (OAC), which was described in more detail in [51]. Since we did not use the full definition of OACs, in our work we talked about OAClets. In EMF, we defined two types of input channels for the robot: EMF uses *action-generating channels* to generate a list of OAClets, which represent the possible actions that the robot

can execute in a given situation. When the robot has filled its list of OACIets, EMF uses *action-evaluating channels* to calculate the relevance of these OACIets in a given context. We described the theoretical background for both channel types and introduced object recognition and task planning as examples for action-generating channels and speech and gesture recognition as example for action-evaluating channels. Finally, we showed an action selection algorithm, which uses a combination of thresholds and an action hierarchy to determine which of the robot's OACIets should be selected for execution in a given situation.

Our goal was to compare CMF and EMF and to research their advantages and disadvantages. In order to make the two approaches comparable, we implemented both of them on the human-robot interaction system of the project Joint Action Science and Technology (JAST). The robot, which we introduced in Section 2.1.1 and can be seen in Figure 2.1, has a humanoid form. It consists of two industrial robot arms that are setup to resemble human arms and an animatronic head which is capable of presenting lip-synchronised speech and basic emotions to a human user. This robot has different input channels, namely speech, gesture, and object recognition. The task of the robot is to work together with a human on a common construction task in which they both assemble target objects together. For that, the robot has an additional input channel, the task planner, which stores information about building plans for target objects.

After introducing the theoretical background of CMF and EMF, in Chapter 4 we presented a description of the implementation of both approaches on the JAST human-robot interaction system. CMF's implementation is based on a rule engine that combines speech and gesture events with information from object recognition, as described above. We presented the rules that we are using in the implementation of CMF on the JAST system. In the EMF approach, the implementation has to be efficient, since the robot-centred data view leads to more computation. Here, we presented how OACIets need to be stored efficiently to enable action-generating and action-evaluating channels to efficiently add new OACIets and compute their relevance, respectively. Furthermore, we presented the implementation of the EMF action selection algorithm in more detail.

In Chapter 5, we presented several experiments, in which CMF and EMF were used. All of these experiments were based on the JAST common construction task and we used both multimodal fusion approaches in different settings to show their functionality and applicability in various scenarios:

- In the first study, we used the CMF approach. Here, the robot took the role of an instructor that explained the human how to build certain target objects. We varied the strategy to explain the plans as well as the strategy to generate referring expression to talk about objects in the environment of the robot. The users in this experiment clearly preferred the robot that used a plan explaining strategy in which it first named the target object and then explained single building steps. Additionally, we found that users who heard the robot that used a referring expression generation strategy, which made use of the context of the current situation, rated the robot as a better dialogue partner.
- In the second study, we used a modified version of CMF. Here, human and robot were equal partners and both had a building plan for the target objects. However, we included a deliberate error in the human's building plan so that the experiment participants did an error during the construction. The robot was able to recognise when the participants made an error and it explained to the users how to solve the error. We varied the robot's strategy to generate referring expressions in its utterances and were able to confirm the findings from the first evaluation: users who heard the robot using a referring expression generation method that made use of context information, assessed the dialogue with the robot as more pleasant.
- In the third study, we used EMF to implement different robot behaviours. Here, human and robot were again equal partners and both had an error-free building plan. In one version of the system the robot took the role of an instructor and told the participants which pieces to pick up; in the other version, the robot took the role of an assistant, handed over pieces to the human when needed, and only gave instructions if necessary. The experiment participants did not have a clear preference in one of the two robot behaviours. However, we found that the users adjusted their behaviour to the robot's actions and took the counterpart to the robot's behaviour. We also found that in this kind of interaction, talking with the robot was not preferred by the users.

6.2 Discussion

The most prominent differences in CMF and EMF are their different ways to represent data and their planning horizons. In this section, we will compare these two properties for both approaches and then discuss how CMF and EMF differ in robot behaviour, application area, fault tolerance, and expandability.

6.2.1 Representation

CMF and EMF have different ways to represent and handle data from the robot's input channels. Representation is a widely discussed topic in the AI and embodiment communities. Classical AI is based on knowledge representations for objects, properties, categories and relations between objects. These representations need to be expressive enough to completely map a given domain, but at the same time logical inference on the represented facts needs to be computable. In contrast to that, most embodiment researchers support the idea that representation is not needed for an embodied agent to show intelligent behaviour. Brooks writes in [18] that AI researchers reduce the problems that they want to solve until they can represent them in an adequate way. This way, in Brooks' opinion, an established scientific approach—dividing complex problems in smaller problems—is abused to give the illusion of generating artificial intelligent systems. Harvey argues in [43] that artificial intelligence can only be generated by a physical system that has inputs and outputs, but no explicit representations. This system uses physical impulses to transport information from the input channels to the appropriate output channels to generate an intelligent reaction to the inputs. He also proposes an evolutionary approach to learn such a system.

CMF and EMF both use explicit representations to represent and process the data from the robot's input channels, even *embodied* multimodal fusion. When it comes to representations, we have a quite practical attitude, which has two reasons: (1) multimodal fusion for human-robot interaction is a domain that has to handle spoken language and planning. To the best of our knowledge, there are no methods from embodiment that have solved these high-level cognitive skills. Thus, we have to use classical approaches for speech processing and the considerate reader may also have noticed that even embodied multimodal fusion uses a combinatory categorial grammar to parse the structure of a given input sentence. (2) Even if one is not convinced that internal representations of external realities do not exist, we still need to represent data in the computers that we are using to process the robot's input data. Harvey arguments in [43] that evolutionary systems do not explicitly develop representations, which is true, but in the end the computer still has to represent data with binary patterns. Thus, our position in this discussion is that as long as we are not developing computers that can reproduce the properties of a human brain, we are forced to work with representations.

Coming back to representations in CMF and EMF, we see that CMF has a human-centred data representation. Specifically, CMF has a speech-centred representation, because in this approach the spoken language of the human is the central element around which CMF build its

fusion hypotheses. This representation is oriented on the data representation of classical state-based dialogue managers. In contrast to this, EMF implements a robot-centred representation of the robot's input data. Here, the robot is seen as an autonomous agent that has defined goals and a set of actions. Thus, the internal representation of this agent is based on actions in combination with objects in the robot's environment. EMF uses the data from the robot's input channels to calculate which of the robot's actions should be executed to reach the given goals.

6.2.2 Planning Horizon

Depending on if the robot needs to reach longterm or immediate goals, it has to plan more or fewer actions in advance to reach these goals. Furthermore, when human and robot have to define new goals together, the robot needs to be able to plan its actions to reach the new goals and compute their reachability. CMF and EMF are essentially different in their planning horizons: in CMF, we are able to integrate planning components that compute the robot's actions for long-term goals.

The rigid structure of the rules that CMF uses for interpreting multimodal input, implement an implicit prediction of the effect of robot actions to the world state, given a specific input. This property is indispensable for long-term goal planning. Furthermore, CMF uses a dialogue manager to communicate with the human, which enables the robot to discuss future plans with its interaction partner. In contrast to that, EMF can only process immediate goals, because it directly incorporates input data to find the next actions for the robot to execute. Thus, a discussion about hypothetical future goals is impossible. However, the robot is able to react fast and robust to short term changes of immediate goals when using EMF.

6.2.3 Robot Behaviour

The behaviour of the robot changes significantly, due to the different views on the input data in both approaches as well as their varying planning horizons. In CMF, verbal utterances of the human are the central information element and the fusion process is built around speech. The robot reacts to these utterances in a predictable and deterministic way, just as it is programmed into the rules in the rule engine. That means, in CMF we have a similar situation as in classical user-initiated dialogue systems in which the system handles orders or information requests by the user. However, this also means that the robot is not able to react to situations in which the human does not act as encoded in the rules by the system developer or in which one of the

modalities of the robot fails, which might lead to situations where none of the preconditions of the rules can be fulfilled.

In the EMF approach, the robot reacts not only to the verbal utterances of the human, but also to the data from other input channels. For example, the robot can react to non-verbal actions of the human, such as picking up a building piece from the human's workspace. That means, the robot shows a behaviour that is similar to mixed-initiative dialogue systems, where it can follow the human's orders as well as decide by itself which action to execute next. However, it has to be noted that using EMF the behaviour of the robot cannot be controlled as easily as in the CMF approach. This is due to the parallel processing of the input data and the fact that the robot works in an unstructured environment and reacts to unstructured input.

6.2.4 Application Area

CMF is applicable in domains in which the robot has a clearly defined task, the actions of the robots can be defined in a precise way, and the environment of the robot can be completely monitored with sensors. Furthermore, the application of CMF is reasonable in domains in which the robot should strictly follow the orders by a human and should rather omit executing an action before doing it wrong. Thus, CMF is best used in applications, in which the robot supports a human who has been instructed how to handle the robot. This would be for example an industrial scenario in which the robot collaborates with a worker or a medical scenario in which a doctor controls the robot to assist him/her in an operation.

EMF is applicable in scenarios, in which the robot cannot completely monitor its environment, but still has to execute a set of actions, and in which it is important that the robot finishes its task, while it is not important in which order it executes subtasks. Here, the robot should follow the orders by the human user, but also be able to autonomously fulfil its assigned tasks. An example for such a scenario would be a household robot that gets the order to clean a table from dirty dishes. Here, the robot can clear the dishes in any order, it is only relevant that it achieves its task.

The different planning horizons of CMF and EMF also influence the application areas of the two approaches. CMF is well-suited for applications that require planning of complex goals and backtracking in case of errors. That makes CMF the right approach for multimodal fusion in industrial scenarios, in which the robot needs to reach long-term goals that need accurate planning and precise execution of intermediate steps. EMF is applicable in areas in which a short-term planning horizon is sufficient and a quick reaction to changes in the environment

is more important. A typical example for such an application would be a robot that socially interacts with humans in public spaces. Here, the robot needs to be able to meet certain ground rules of social interaction, such as greeting the human.

6.2.5 Fault Tolerance

A fault tolerant system can react robustly to errors. The types of errors that we are dealing with in multimodal fusion are of two kinds: on the one hand, we have unexpected situations in the interaction, for example the user can pick up a wrong building piece or say a sentence that does not fit into the current context. On the other hand, there can be errors in the system itself, for example modalities can deliver false data or completely break down.

In the current implementation, CMF is not able to react to errors in the interaction or in the system. User input errors could be caught by appropriately implemented rules, but if one of the modalities breaks, CMF is not able to work properly. In EMF, the robot is working, even if one of its modalities breaks down. In that case, it will use the information from its remaining modalities to solve its assigned task. However, EMF is—in the implementation presented here—not able to detect errors in the interaction. The system uses every input by the human, also faulty inputs, and interprets them in its calculation to determine its next actions.

6.2.6 Expandability and Implementation

In this section, we look at the expandability of the two approaches, for example to add a new modality to the robot's input sensors or extend the capabilities of one of the existing modalities. For that, we enumerate the system parts that need to be changed to extend both approaches. Additionally, we analyse the process of implementing CMF or EMF on a different robot. We highlight the duration to implement a first working version, the steps to analyse a given scenario and adapt the approaches, and the integrability of the two approaches in already existing systems.

To extend CMF by a new modality, a developer has to write a new set of rules that handle the events of the new modality as well as change the old rule set in order to fuse the events from the new modality with the already existing information. However, the general processing with a rule engine is not affected by adding or changing a modality. Due to its modular structure, implementing the CMF approach on a new system is relatively easy. The implementation here can be done in a step-by-step manner. Here, the programmer has to analyse the scenario in which the new robot works in full detail before starting to implement the CMF approach,

which is time-consuming. Also, it will take a long time to get to an initial running version of the system, because many components have to work in order for the system to run in CMF. In return, the CMF approach is well-suited to be implemented in already existing systems, again due to its modular structure.

EMF can be easier extended than CMF. Due to the parallel processing of the input data, every modality works independently from other input channels. Thus, existing and new modalities can be added or removed to the processing system at all times. The generation and evaluation of OACIets from each modality does not change when another modality is added to or removed from the robot. The developer has to analyse the action-generating and action-evaluating properties of a new modality that should be added to the robot. Furthermore, he/she has to make sure that the calculation of relevance values in the new modality is normalised with the relevance calculation of the already existing modalities. Furthermore, the developer has to control the thresholds and action hierarchy in the action selection mechanism, when a new modality is added.

One main disadvantage of EMF is that it is not as easily portable to new robots as the CMF approach. As it was described by Pfeifer in [68] as a design principle for embodied systems, EMF uses synthetic methodology, which means that the developer understands the robot while building the system. In EMF, we have to understand and analyse a robot's input channels to find out whether they are action-generating or action-evaluating channels, while we are implementing the fusion process. However, EMF has the advantage that with its methodology the developer can rather quickly produce a working prototype of a new robot system, even if not all of the input channels are already working. In contrast to CMF however, the EMF developer will have to invest more time into fine-tuning the final integrated system, which is again due to the parallel processing of input data in EMF that can lead to unwanted side effects.

6.2.7 Take Home Messages

After this discussion of the advantages and disadvantages of CMF and EMF, we believe that embodied multimodal fusion is the approach that is better suited for human-robot interaction. This is mainly due to the fact that in the near future we will not be able to completely predict the behaviour of the humans the robot is interacting with as well as to completely monitor the unstructured environment of the robot with sensors, two preconditions that classical multimodal fusion needs in order to function properly. Using EMF, a robot can react robustly to errors or missing input, also EMF is tailored to the available input modalities of the robot and makes the

best use of the given input data. However, classical multimodal fusion also has its justification in certain application areas, in which the environment can be completely monitored, the input data is holistically known, and the robot needs to precisely follow a given plan and the human's instructions. Thus, we can learn some important lessons from this work:

1. *If you need a robot that can be precisely controlled by human orders using language and other modalities, use classical multimodal fusion.* The CMF approach demonstrates that the robot can be programmed to exactly follow the input by the human and a given construction task that is defined by a plan.

2. *If you need a robot that can solve given tasks autonomously and in a robust way, use embodied multimodal fusion.* The EMF approach showed that the robot can process input by a human and also adjust its actions accordingly, but it mainly works on its assigned tasks until the task is finished.

3. *Embodiment can be the basis for robust data processing. Decision making should be based on artificial intelligence.* Both of the introduced approaches have their advantages and we believe that a combination of aspects of CMF and EMF will lead to an approach for multimodal fusion that can overcome the deficiencies of the two methods.

6.3 Future Work

We believe that in the future a combination of methods from embodiment and AI will be a powerful combination for a multimodal fusion, which on the one hand processes input data robustly and fault tolerant and on the other hand uses logical calculus to select the robot's next actions. This approach will enable the robot to solve challenges such as revising actions that have led to an error or replanning actions in case of new situations in the interaction. Furthermore, the robot could use a more active approach to fill its knowledge about the environment with missing data by asking the user questions in case of uncertainties about the current state of the environment.

Concretely, we want to develop an approach for multimodal fusion, which uses the parallel data processing of EMF to generate representations of user inputs, task information, and environment status. These representations will then be processed with CMF. For this, we need to solve the following challenges: (1) The human-centred data view of CMF and the robot-centred data view of EMF need to be combined into one representation format. (2) The robot needs to

keep a dialogue history in which it stores the information about the interaction with its human partner. This can be used to recognise and solve errors in the interaction. (3) We need to model the timing of the robot. For example, we need to consider when the robot should react to the human's utterances and actions to make the interaction more natural for the human, and we need to empirically study, how the robot can influence the interaction by strategically altering the speed in which it executes actions.

Appendix A

Appendix

A.1 CMF Rules

```
package de.tum.in.jast.interpretation

import de.tum.in.jast.interpretation.DisappearedObject;
import de.tum.in.jast.interpretation.Gesture;
import de.tum.in.jast.interpretation.GoalInference;
import de.tum.in.jast.interpretation.Speech;
import de.tum.in.jast.interpretation.Utterance;
import de.tum.in.jast.interpretation.WorldModelObject;
import jast.common.InstantiatedObject;
import jast.common.Location;
import jast.common.LocationType;
import jast.common.Timestamp;
import jast.listener.GoalInferenceOutput;
import jast.reasoning.HypothesisType;
import jast.reasoning.ObjectLink;
import java.util.Collections;

import function de.tum.in.jast.interpretation.Helper.getEmptyDocument;
import function de.tum.in.jast.interpretation.Helper.getEmptyObjectlinks;
import function de.tum.in.jast.interpretation.Helper.getHypothesisType;
import function de.tum.in.jast.interpretation.Helper.generateObjectLinks;
import function de.tum.in.jast.interpretation.Helper.generateObjectLinksSpeech;
import function de.tum.in.jast.interpretation.Helper.getNumberOfObjects;
import function de.tum.in.jast.interpretation.Helper.idsResolved;
import function de.tum.in.jast.interpretation.Helper.unifyHashtables;
import function de.tum.in.jast.interpretation.Helper.getEmptyGoalInferenceOutput;
import function de.tum.in.jast.interpretation.Helper.getEmptyInstantiatedObject;

# global variables
global java.lang.Long TIMEOUT;
global java.lang.Long ATTENTIONSPAN;
global java.lang.Long WAITINGFORGOALINFERENCE;
global java.lang.Long SINGLEGESTURE;

//**************//
// only gesture //
//**************//
rule "pointing_gesture"
  when
    gesture : Gesture( type == "Pointing", active == true , singleGesture == false )
    timer : Timestamp()

    eval ( Math.abs(timer.msec - gesture.getStartTime().msec) > SINGLEGESTURE.longValue() )
  then
    System.out.println("RuleBase: rule 'pointing_gesture'");

    gesture.setActive( false );
    gesture.setSingleGesture( true );
    update( gesture );
end
```

```
//****************************//
// deictic speech and gesture //
//****************************//
rule "deictic speech and pointing gesture, resolved"
  when
    speech : Speech( hasDeictic == true, active == true )
    gesture : Gesture( type == "Pointing", active == true )
    timer : Timestamp()

    // test if objects talked about and pointed at match
    eval( getHypothesisType(speech.getIdsAndObjects(), gesture.getPointedAtIds()) == HypothesisType.Resolved )
  then
    System.out.println("RuleBase: rule 'deictic speech and pointing gesture, resolved'");

    insert( new FusionHypothesis(HypothesisType.Resolved,
                  speech.getLogicalForm(),
                  speech.getDocument(),
                  unifyHashtables(speech.getIdsAndObjects(), gesture.getPointedAtIds()),
                            getEmptyInstantiatedObject(),
                            getEmptyGoalInferenceOutput()
                           )
            );

    speech.setActive(false);
    update( speech );
    gesture.setActive(false);
    update( gesture );
end

rule "deictic speech and pointing gesture, unresolved"
  when
    speech : Speech( hasDeictic == true, active == true )
    gesture : Gesture( type == "Pointing", active == true )
    timer : Timestamp()

    // test if objects talked about and pointed at match
    eval( getHypothesisType(speech.getIdsAndObjects(), gesture.getPointedAtIds()) == HypothesisType.Unresolved )
  then
    System.out.println("RuleBase: rule 'deictic speech and pointing gesture, unresolved'");

    insert( new FusionHypothesis(HypothesisType.Unresolved,
                  speech.getLogicalForm(),
                  speech.getDocument(),
                  unifyHashtables(speech.getIdsAndObjects(), gesture.getPointedAtIds()),
                            getEmptyInstantiatedObject(),
                            getEmptyGoalInferenceOutput()
                           )
            );

    speech.setActive( false );
    update( speech );
    gesture.setActive( false );
    update( gesture );
end

rule "deictic speech and pointing gesture, conflict"
  when
    speech : Speech( hasDeictic == true, active == true )
    gesture : Gesture( type == "Pointing", active == true )
    timer : Timestamp()

    eval( getHypothesisType(speech.getIdsAndObjects(), gesture.getPointedAtIds()) == HypothesisType.Conflict )
  then
    System.out.println("RuleBase: rule 'deictic speech and pointing gesture, conflict'");

    insert( new FusionHypothesis(HypothesisType.Conflict,
                  speech.getLogicalForm(),
                  speech.getDocument(),
                  unifyHashtables(speech.getIdsAndObjects(), gesture.getPointedAtIds()),
                            getEmptyInstantiatedObject(),
                            getEmptyGoalInferenceOutput()
                           )
            );

    speech.setActive( false );
    update( speech );
    gesture.setActive( false );
    update( gesture );
end
```

```
rule "deictic speech and pointing gesture, ambiguous"
  when
    speech : Speech( hasDeictic == true, active == true )
    gesture : Gesture( type == "Pointing", active == true )
    timer : Timestamp()

    eval( getHypothesisType(speech.getIdsAndObjects(), gesture.getPointedAtIds()) == HypothesisType.Ambiguous )
  then
    System.out.println("RuleBase: rule 'deictic speech and pointing gesture, ambiguous'");

    insert( new FusionHypothesis(HypothesisType.Ambiguous,
                  speech.getLogicalForm(),
                  speech.getDocument(),
                  unifyHashtables(speech.getIdsAndObjects(), gesture.getPointedAtIds()),
                            getEmptyInstantiatedObject(),
                            getEmptyGoalInferenceOutput()
                           )
            );

    speech.setActive( false );
    update( speech );
    gesture.setActive( false );
    update( gesture );
end


//******************************//
// deictic speech but no gesture //
//******************************//
rule "speech, deictic expression, no gesture"
  when
    speech : Speech( hasDeictic == true, active == true )
    timer : Timestamp()

    eval( Math.abs(speech.getStartTime().msec - timer.msec) > ATTENTIONSPAN )
  then
    System.out.println("RuleBase: rule 'speech, deictic expression, no gesture'");

    insert( new FusionHypothesis(HypothesisType.Unresolved,
                            speech.getLogicalForm(),
                            speech.getDocument(),
                            generateObjectLinksSpeech(speech.getIdsAndObjects(), false),
                            getEmptyInstantiatedObject(),
                            getEmptyGoalInferenceOutput()
                           )
            );

    speech.setActive( false );
    update( speech );
end

rule "speech, no deictic expression, definite determiner, ambiguous"
  when
    speech : Speech( hasDeictic == false, hasDefDet == true, active == true )
    timer : Timestamp()

    eval ( idsResolved(speech.getIdsAndObjects()) == HypothesisType.Ambiguous )
  then
    System.out.println("RuleBase: rule 'speech, no deictic expression, definit determiner, ambiguous'");

    //insert( new FusionHypothesis(HypothesisType.Ambiguous,
    insert( new FusionHypothesis(HypothesisType.Resolved,
                            speech.getLogicalForm(),
                            speech.getDocument(),
                            generateObjectLinksSpeech(speech.getIdsAndObjects(), true),
                            getEmptyInstantiatedObject(),
                            getEmptyGoalInferenceOutput()
                           )
            );

    speech.setActive( false );
    update( speech );
end

rule "speech, no deictic expression, no definit determiner, resolved"
  when
    speech : Speech( hasDeictic == false, hasDefDet == false, active == true )
    timer : Timestamp()

    eval ( idsResolved(speech.getIdsAndObjects()) != HypothesisType.Unresolved )
  then
```

```
        System.out.println("RuleBase: rule 'speech, no deictic expression, no definit determiner, resolved'");

        insert( new FusionHypothesis(HypothesisType.Resolved,
                                     speech.getLogicalForm(),
                                     speech.getDocument(),
                                     generateObjectLinksSpeech(speech.getIdsAndObjects(), true),
                                     getEmptyInstantiatedObject(),
                                     getEmptyGoalInferenceOutput()
                                    )
                );

        speech.setActive( false );
        update( speech );
    end

    rule "speech, no deictic expression, definit determiner, resolved"
      when
        speech : Speech( hasDeictic == false, hasDefDet == true, active == true )
        timer : Timestamp()

        eval ( idsResolved(speech.getIdsAndObjects()) == HypothesisType.Resolved )
      then
        System.out.println("RuleBase: rule 'speech, no deictic expression, definit determiner, resolved'");

        insert( new FusionHypothesis(HypothesisType.Resolved,
                                     speech.getLogicalForm(),
                                     speech.getDocument(),
                                     generateObjectLinksSpeech(speech.getIdsAndObjects(), true),
                                     getEmptyInstantiatedObject(),
                                     getEmptyGoalInferenceOutput()
                                    )
                );

        speech.setActive( false );
        update( speech );
    end

    rule "speech, no deictic expression, no definit determiner, unresolved"
      when
        speech : Speech( hasDeictic == false, hasDefDet == false, active == true )
        timer : Timestamp()

        eval ( idsResolved(speech.getIdsAndObjects()) == HypothesisType.Unresolved )
      then
        System.out.println("RuleBase: rule 'speech, no deictic expression, no definit determiner, unresolved'");

        insert( new FusionHypothesis(HypothesisType.Unresolved,
                                     speech.getLogicalForm(),
                                     speech.getDocument(),
                                     generateObjectLinksSpeech(speech.getIdsAndObjects(), true),
                                     getEmptyInstantiatedObject(),
                                     getEmptyGoalInferenceOutput()
                                    )
                );

        speech.setActive( false );
        update( speech );
    end

    rule "speech, no deictic expression, definit determiner, unresolved"
      when
        speech : Speech( hasDeictic == false, hasDefDet == true, active == true )
        timer : Timestamp()

        eval ( idsResolved(speech.getIdsAndObjects()) == HypothesisType.Unresolved )
      then
        System.out.println("RuleBase: rule 'speech, no deictic expression, no definit determiner, unresolved'");

        insert( new FusionHypothesis(HypothesisType.Unresolved,
                                     speech.getLogicalForm(),
                                     speech.getDocument(),
                                     generateObjectLinksSpeech(speech.getIdsAndObjects(), true),
                                     getEmptyInstantiatedObject(),
                                     getEmptyGoalInferenceOutput()
                                    )
                );

        speech.setActive( false );
        update( speech );
    end
```

```
295
    //*************//
    // other rules //
    //*************//
    rule "set_old_utterances_inactive"
300   when
        utterance : Utterance( active == true )
        timer : Timestamp()

        eval( Math.abs(utterance.getStartTime().msec - timer.msec) > TIMEOUT.longValue() )
305   then
        System.out.println("RuleBase: rule 'set_old_utterances_inactive'");

        utterance.setActive(false);
        update( utterance );
310 end

    rule "nothing_happened_for_TIMEOUT_msecs"
      when
        timer : Timestamp()
315     utterance : Utterance( active == false )
        eval( Math.abs(utterance.getStartTime().msec - timer.msec) > TIMEOUT.longValue() )
      then
    //    System.out.println("RuleBase: rule 'nothing happened for TIMEOUT msecs'");

320     retract( utterance );
    end

    // some queries for the outside world to get data from the working memory
    query "all_hypotheses"
325     hypothesis : FusionHypothesis()
    end

    query "resolved_hypotheses"
        hypothesis : FusionHypothesis( type == "Resolved" )
330 end

    query "all_speech"
        speech : Speech()
    end
335
    query "single_gestures"
        gesture : Gesture( singleGesture == true )
    end

340 query "disappeared_objects"
        disobject : DisappearedObject( active == false )
    end
```

A.2 Table Layouts

In Figures A.1 and A.2 we show the two initial table layouts that were used for the JAST user evaluations. The Baufix pieces were chosen in a way so that the robot had to hand over pieces to the human, the user had to ask the robot for one of the pieces at one point. Furthermore, there were enough duplicate or similar pieces on the table to generate ambiguous situations so that human and robot had to either use pointing gestures or produce elaborated referring expressions to refer to certain objects.

A.3 Target Object Building Plans

A.3.1 Regular Plans

A.3.2 Plans with Errors

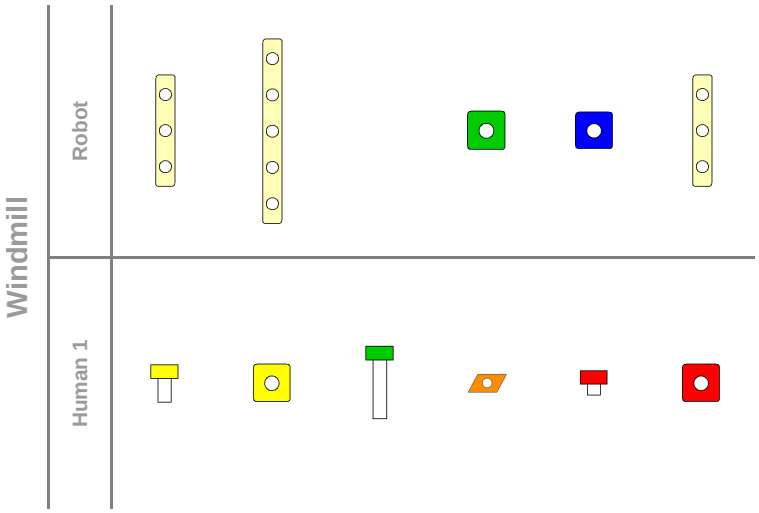


Figure A.1: Initial table layout of Baufix pieces for building a windmill.

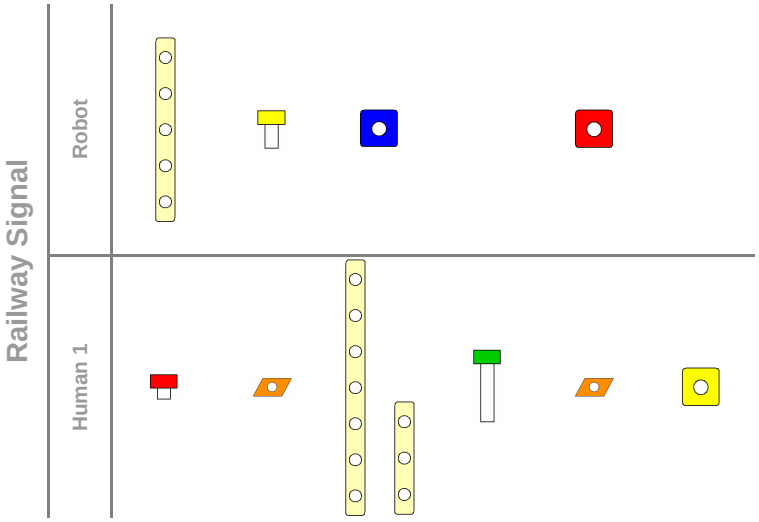


Figure A.2: Initial table layout of Baufix pieces for building a railway signal.

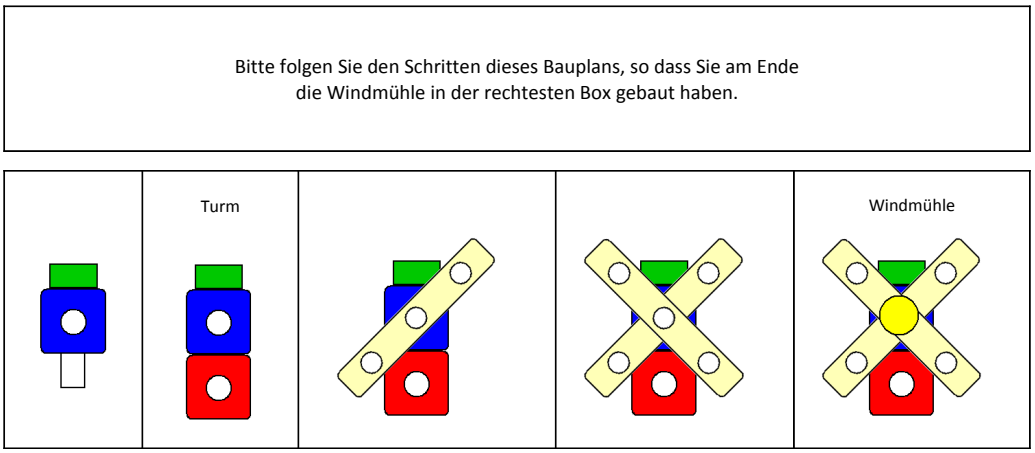


Figure A.3: Regular building plan for the windmill.

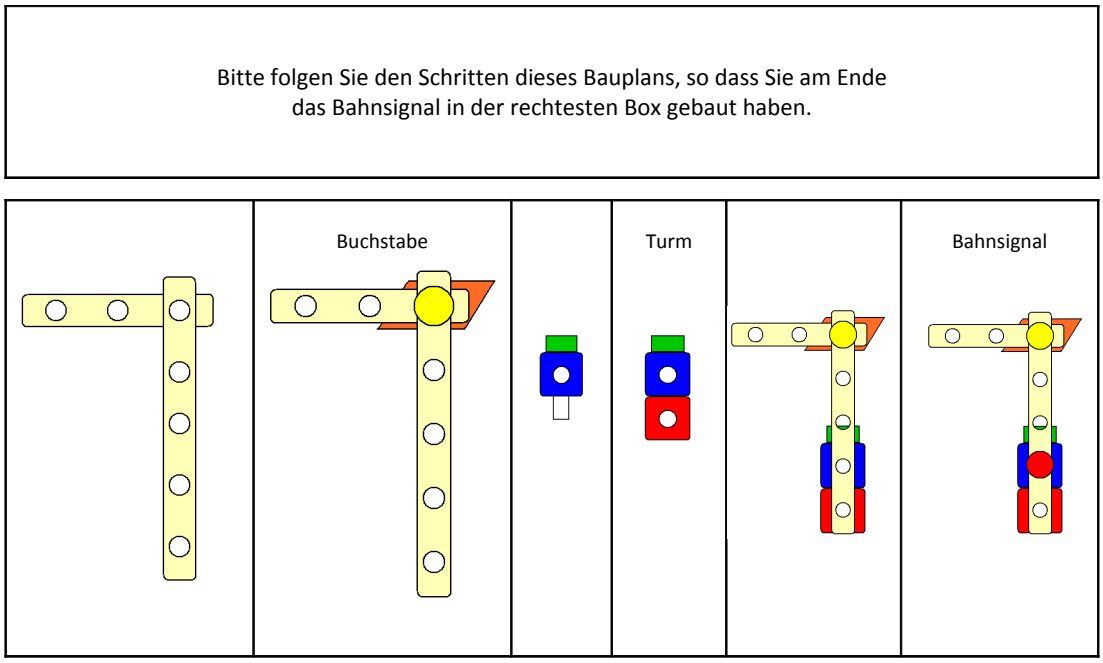


Figure A.4: Regular building plan for the railway signal.

A.4 User Questionnaires

A.4.1 Evaluation 1

Task Success

1. Ich habe mit dem Roboter erfolgreich zusammengearbeitet.
   I was able to work with the robot successfully.

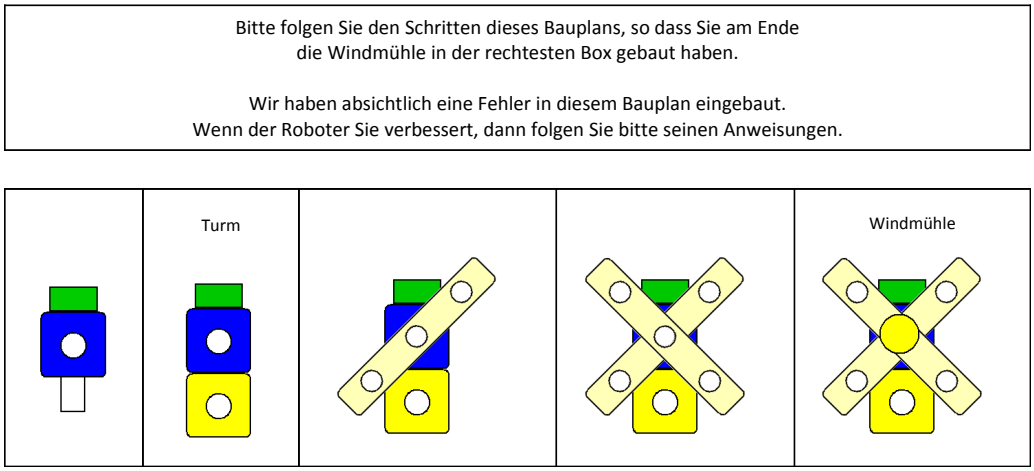


Figure A.5: Building plan for the windmill that contains an error.

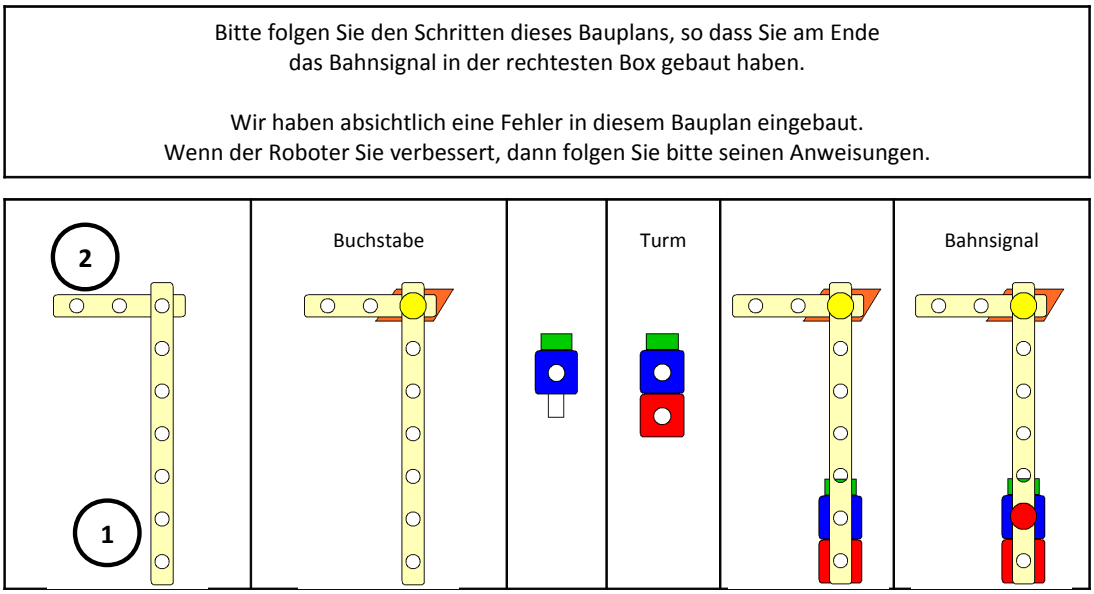


Figure A.6: Building plan for the railway signal that contains an error.

2. Ich habe mit dem Roboter verschiedene Objekte zusammengebaut.
   I was able to build several objects with the robot.
3. Der Roboter hat mir beigebracht, wie man bestimmte Objekte zusammenbaut.
   The robot instructed me how to build certain objects.
4. Ich weiß jetzt wie man einen “Schneemann” baut.
   I know how to build a “snowman”.

5. Ich weiß jetzt wie man eine “Windmühle” baut.
   I know how to build a “windmill”.
6. Ich weiß jetzt wie man einen “Buchstaben L” baut.
   I know how to build an “l-shape”.
7. Ich weiß jetzt wie man eine “Bahnsignal” baut.
   I know how to build a “railway signal”.

Usability

1. Es war einfach mit dem Roboter zusammenzuarbeiten.
   It was easy to work with the robot.
2. Es war einfach zu verstehen welche Bauteile ich benutzen sollte.
   It was easy to understand which assembly pieces I had to use.
3. Es war einfach zu verstehen wie die Bauteile zusammengefügt werden mussten.
   It was easy to understand how the pieces needed to be assembled.
4. Es war einfach sich zu erinnern wie die verschiedenen Objekte zusammengebaut werden.
   It was easy to remember how to assemble the different target objects.
5. Die verschiedenen Aufgaben waren zu schwierig.
   The different tasks were to difficult.
6. Die Namen für die Objekte waren eingängig.
   The target object names were plausible.

Opinion About the Robot

1. Der Roboter konnte verstehen was ich sage.
   The robot understood what I said to it.
2. Der Roboter hat schnell auf meine Äußerungen reagiert.
   The robot responded quickly to my requests.
3. Wenn der Roboter Bauteile aufgenommen hat, dann schien mir das sinnvoll.
   It seemed reasonable to me when the robot picked up assembly pieces.
4. Ich fand es hilfreich, dass mir der Roboter Bauteile gereicht hat.
   I found it useful when the robot handed over assembly pieces to me.
5. Die Bewegungen der Roboterarme sahen natürlich aus.
   The robot movements looked natural to me.
6. Die Bewegungen des Roboterkopfs sahen natürlich aus.
   The robot head movements looked natural to me.
7. Ich fand die Bewegungen des Roboterkopfs hilfreich.
   The robot head movements were helpful.
8. Ich fand die Stimme des Roboters leicht zu verstehen.
   I found the robot’s voice easy to understand.
9. Der Roboter gab mir nützliche Anweisungen.
   The robot gave helpful instructions.

10. Es war einfach den Anweisungen des Roboters zu folgen.
    It was easy to follow the instructions by the robot.
11. Der Roboter gab zu viele Anweisungen auf einmal.
    The robot gave too many instructions at once.
12. Die Anweisungen des Roboters waren zu ausführlich.
    The robot's instructions were to detailed.
13. Immer wenn der Roboter über Bauteile gesprochen hat, wusste ich genau, von welchem Bauteil er spricht.
    When the robot talked about objects, I always knew which objects it meant.

Interaction

1. Der Roboter hat sich so verhalten wie ich es erwarten würde.
   The robot worked the way I expected it to.
2. Ich habe zu jedem Zeitpunkt in der Konversation gewusst was ich tun oder sagen kann.
   I knew what I could say or do at each point in the conversation.
3. Ich habe immer gewusst wann ich zu sprechen anfangen musste.
   I knew when to begin speaking.
4. Wenn der Roboter mich nicht verstand, dann war mir klar was ich tun musste.
   It was clear what to do when the robot did not understand me.
5. Der Roboter war kooperativ während wir zusammengearbeitet haben.
   The robot was cooperative during our collaboration.
6. Der Roboter war flexibel während wir zusammengearbeitet haben.
   The robot was flexible during our collaboration.
7. Ich hatte das Gefühl, dass ich den Roboter kontrollieren kann.
   I had the feeling, I could control the robot.
8. Ich war verwirrt als ich den Roboter benutzte.
   I felt confused when using the robot.
9. Ich war verwirrt als ich den Roboter benutzte.
   I felt frustrated when using the robot.
10. Ich fand, der Roboter war einfach zu benutzen.
    I found the robot easy to use.

Dialogue

1. Ich fand die Konversation einnehmend.
   I found the conversation engaging.
2. Ich fand es aufregend mit dem Roboter zu interagieren.
   I found it exciting to interact with the robot.
3. Ich war so in der Interaktion mit dem Roboter versunken, dass ich ganz die Zeit vergessen habe.
   I was so engaged in the interactions that I lost track of time.

4. Ich war angespannt als ich mit dem Roboter zusammengearbeitet habe.
   I felt tense when using the robot.
5. Ich musste mich wirklich konzentrieren, während ich den Roboter benutzte.
   I really had to concentrate to use the robot.
6. Ich fand die Konversation mit dem Roboter langweilig.
   I found the conversation boring.

General Questions

1. I habe gerne mit dem Roboter zusammengearbeitet.
   I liked using the robot.
2. Ich fand der Roboter war freundlich.
   I found the robot was friendly.
3. Der Roboter schien über die Aufgabe Bescheid zu wissen.
   I found the robot to be knowledgeable.
4. Der Roboter erschien mir intelligent.
   The robot appeared to be intelligent.
5. Der Roboter gab mir gute Anweisungen.
   The robot gave me good instructions.

A.4.2 Evaluation 2

Task Success

1. Ich habe mit dem Roboter erfolgreich zusammengearbeitet.
   I was able to work with the robot successfully.
2. Ich habe mit dem Roboter verschiedene Objekte zusammengebaut.
   I was able to build several objects with the robot.
3. Die verschiedenen Aufgaben waren zu schwierig.
   The assembly tasks were too difficult.
4. Es war einfach zu verstehen wie die Bauteile zusammengefi¿$\frac{1}{2}$gt werden mussten.
   It was easy to understand how to put the pieces together.
5. Es war einfach zu verstehen welche Bauteile ich benutzen sollte.
   It was easy to understand which pieces to use

Goal Inference

1. Wenn der Roboter Bauteile aufgenommen hat, dann schien mir das sinnvoll.
   It seemed natural when the robot picked up objects from the table.
2. Ich fand es hilfreich, dass mir der Roboter Bauteile gereicht hat.
   I found it helpful when the robot picked up objects from the table.
3. Der Roboter schien zu verstehen was ich wollte.
   The robot seemed to understand what I wanted.

4. Der Roboter schien zu verstehen warum, wenn ich Bauteile vom Tisch aufnahm.
   When I picked pieces up the robot seemed to know why.
5. Wenn ich ein falsches Bauteil aufnahm, dann war der Roboter in der Lage mich zu korrigieren.
   When I picked up a wrong piece the robot was able to correct that.

Feeldings Towards The Robot

1. Ich war verwirrt als ich den Roboter benutzte.
   I felt confused when using the robot.
2. Es war einfach mit dem Roboter zusammenzuarbeiten.
   It was easy to work with the robot.
3. Ich fand, der Roboter war einfach zu benutzen.
   I found the robot easy to use.
4. Ich fand die Konversation einnehmend.
   I found the conversation engaging.
5. Ich fand es aufregend mit dem Roboter zu interagieren.
   I found it exciting to interact with the robot.
6. Ich war angespannt als ich mit dem Roboter zusammengearbeitet habe.
   I felt tense when using the robot.
7. Ich musste mich wirklich konzentrieren, während ich den Roboter benutzte.
   I really had to concentrate to use the robot.
8. Ich fand die Konversation mit dem Roboter langweilig.
   I found the conversation boring.
9. I habe gerne mit dem Roboter zusammengearbeitet.
   I liked using the robot.
10. Der Roboter erschien mir intelligent.
    The robot appeared to be intelligent.

Interaction

1. Der Roboter konnte verstehen was ich sage.
   The robot understood what I said to it.
2. Der Roboter hat schnell auf meine Äußerungen reagiert.
   The robot responded quickly to my requests.
3. Ich fand die Stimme des Roboters leicht zu verstehen.
   I found the voice of the robot easy to understand.
4. Immer wenn der Roboter über Bauteile gesprochen hat, wusste ich genau, von welchem Bauteil er spricht.
   When the robot talked about objects, I always knew which objects it meant.
5. Ich habe zu jedem Zeitpunkt in der Konversation gewusst was ich tun oder sagen kann.
   I knew what I could say or do at each point in the conversation.

6. Ich habe immer gewusst wann ich zu sprechen anfangen musste.
   I knew when to begin speaking.
7. Wenn der Roboter mich nicht verstand, dann war mir klar was ich tun musste.
   It was clear what to do when the robot did not understand me.
8. Der Roboter hat sich so verhalten wie ich es erwarten würde.
   The robot worked the way I expected it to.
9. Der Roboter schien über die Aufgabe Bescheid zu wissen.
   I found the robot to be knowledgeable.

A.4.3 Evaluation 3

Task Success

1. Ich habe mit dem Roboter erfolgreich zusammengearbeitet.
   I was able to work with the robot successfully.
2. Der Roboter hat mir beim Zusammenbauen der Windmühle gut geholfen.
   The robot helped me well to build the windmill.
3. Der Roboter hat mir beim Zusammenbauen des Bahnsignals gut geholfen.
   The robot helped me well to build the railway signal.
4. Die verschiedenen Aufgaben waren zu schwierig.
   The assembly tasks were too difficult.
5. Es war einfach zu verstehen wie die Bauteile zusammengefügt werden mussten.
   It was easy to understand how to put the pieces together.
6. Es war einfach zu verstehen welche Bauteile ich benutzen sollte.
   It was easy to understand which pieces to use

Feeldings Towards The Robot

1. Ich war verwirrt als ich den Roboter benutzte.
   I felt confused when using the robot.
2. Es war einfach mit dem Roboter zusammenzuarbeiten.
   It was easy to work with the robot.
3. Ich fand, der Roboter war einfach zu benutzen.
   I found the robot easy to use.
4. Ich fand die Konversation einnehmend.
   I found the conversation engaging.
5. Ich fand es aufregend mit dem Roboter zu interagieren.
   I found it exciting to interact with the robot.
6. Ich war angespannt als ich mit dem Roboter zusammengearbeitet habe.
   I felt tense when using the robot.
7. Ich musste mich wirklich konzentrieren, während ich den Roboter benutzte.
   I really had to concentrate to use the robot.

8. Ich fand die Konversation mit dem Roboter langweilig.
   I found the conversation boring.
9. I habe gerne mit dem Roboter zusammengearbeitet.
   I liked using the robot.
10. Der Roboter erschien mir intelligent.
    The robot appeared to be intelligent.

Interaction

1. Der Roboter konnte verstehen was ich sage.
   The robot understood what I said to it.
2. Der Roboter hat schnell auf meine Äußerungen reagiert.
   The robot responded quickly to my requests.
3. Ich fand die Stimme des Roboters leicht zu verstehen.
   I found the voice of the robot easy to understand.
4. Immer wenn der Roboter über Bauteile gesprochen hat, wusste ich genau, von welchem Bauteil er spricht.
   When the robot talked about objects, I always knew which objects it meant.
5. Ich habe zu jedem Zeitpunkt in der Konversation gewusst was ich tun oder sagen kann.
   I knew what I could say or do at each point in the conversation.
6. Wenn der Roboter mich nicht verstand, dann war mir klar was ich tun musste.
   It was clear what to do when the robot did not understand me.
7. Der Roboter hat sich so verhalten wie ich es erwarten würde.
   The robot worked the way I expected it to.
8. Der Roboter schien über die Aufgabe Bescheid zu wissen.
   I found the robot to be knowledgeable.

Robot Behaviour

1. Wenn der Roboter Bauteile aufgenommen hat, dann schien mir das sinnvoll.
   When the robot handed over pieces, it seemed useful to me.
2. Ich fand es hilfreich, dass mir der Roboter Bauteile gereicht hat.
   I found it helpful, when the robot handed over pieces to me.
3. Der Roboter hat ein eher aktives Verhalten gezeigt.
   The robot showed an active behaviour.
4. Der Roboter hat ein eher passives Verhalten gezeigt.
   The robot showed an passive behaviour.
5. Der Roboter gab zu viele Anweisungen.
   The robot gave too many instructions.
6. Die Anweisungen des Roboters waren ausreichend.
   The instructions by the robot were sufficient.

References

[1] A. E. Ades and M. J. Steedman. On the order of words. *Linguistics and philosophy*, 4:517–558, 1982. 25

[2] K. Ajdukiewicz. Die syntaktische konnexität. *Studia Philosophica*, 1:1–27, 1935. 25

[3] J. F. Allen and G. Ferguson. Actions and events in interval temporal logic. *Journal of Logic and Computation*, 4:531–579, 1994. 40

[4] J. R. Anderson, D. Bothell, M. D. Byrne, S. Douglass, C. Lebiere, and Y. Qin. An integrated theory of the mind. *Psychological Review*, 111(4):1036–1060, 2004. 13

[5] M. Argyle and J. A. Graham. The Central Europe experiment: Looking at persons and looking at objects. *Environmental Psychology and Nonverbal Behavior*, 1(1):6–16, 1976. 85

[6] J. Baldridge and G.-J. Kruijff. Coupling ccg and hybrid logic dependency semantics. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 02)*, Philadelphia, PA: University of Pennsylania, 2002. 25

[7] Y. Bar-Hillel. A quasi-arithmetic notation for syntactic description. *Language*, 29:47–58, 1953. 25

[8] E. G. Bard, R. Hill, and M. E. Foster. What tunes accessibility of referring expressions in task-related dialogue? In *Proceedings of the 30th Annual Meeting of the Cognitive Science Society (CogSci 2008)*, Chicago, July 2008. 94

[9] E. G. Bard, R. L. Hill, M. E. Foster, and M. Arai. How do we tune accessibility in joint tasks: Roles and regulations, In prep. 100

[10] E. Bicho, L. Louro, and W. Erlhagen. Integrating verbal and non-verbal communication in a dynamic neural field architecture for human-robot interaction. *Frontiers in Neurorobotics*, 4, May 2010. 94

[11] E. Bicho, L. Louro, N. Hipolito, and W. Erlhagen. A dynamic field approach to goal inference and error monitoring for human-robot interaction. In *Proceedings of the Symposium on "New Frontiers in Human-Robot Interaction", AISB 2009 Convention*, Heriot-Watt University Edinburgh, 2009. 9, 94

[12] R. A. Bolt. "put-that-there": Voice and gesture at the graphics interface. In *SIGGRAPH '80: Proceedings of the 7th annual conference on Computer graphics and interactive techniques*, pages 262–270, New York, NY, USA, 1980. ACM Press. 12

[13] L. Boves, A. Neumann, L. Vuurpijl, L. Bosch, S. Rossignol, R. Engel, and N. Pfleger. Multimodal interaction in architectural design applications. *Lecture Notes In Computer Science*, pages 384–390, 2004. 12

[14] C. Breazeal. Toward sociable robots. *Robotics and Autonomous Systems*, 42(3-4):167–175, 2003. 16

[15] C. Breazeal. Socially intelligent robots. *Interactions*, 12(2):19–22, 2005. 16

[16] T. Brick and M. Scheutz. Incremental natural language processing for hri. In *Proceedings of the ACM/IEEE international conference on Human-robot interaction*, pages 263–270. ACM New York, NY, USA, 2007. 17

[17] R. Brooks. Elephants don't play chess. *Robotics and autonomous systems*, 6(1-2):3–15, 1990. 18

[18] R. Brooks. Intelligence without representation. *Artificial intelligence*, 47(1-3):139–159, 1991. 115

[19] R. A. Brooks. From earwigs to humans. *Robotics and Autonomous Systems*, 20(2-4):291–304, 1997. 20

[20] P. Browne. *JBoss Drools Business Rules*. Birmingham, UK: Packt Publishing, 2009. 64

[21] C. Burghart, R. Mikut, R. Stiefelhagen, T. Asfour, H. Holzapfel, P. Steinhaus, and R. Dillmann. A cognitive architecture for a humanoid robot: A first approach. In *Proceedings*

of 2005 5th IEEE-RAS International Conference on Humanoid Robots, pages 357–362, Tsukuba, Japan, 2005. IEEE Institute of Electrical and Electronics Engineers. 16

[22] M. D. Byrne. Cognitive architectures in hci: Present work and future directions. In *Proceedings of the 11th International Conference on Human Computer Interaction*, 2005. 14

[23] G. Castellano, R. Aylett, A. Paiva, and P. McOwan. Affect recognition for interactive companions. In *Workshop on Affective Interaction in Natural Environments (AFFINE), ACM International Conference on Multimodal Interfaces (ICMI 08)*, 2008. 16

[24] H. Christensen, A. Sloman, G.-J. Kruijff, and J. Wyatt. *Cognitive Systems*. Available for download at http://www.cognitivesystems.org/cosybook/, 2009. 15

[25] H. Clark. Coordinating with each other in a material world. *Discourse studies*, 7(4-5):507, 2005. 34, 49

[26] P. R. Cohen, M. Johnston, D. McGee, S. Oviatt, J. Pittman, I. Smith, L. Chen, and J. Clow. Quickset: multimodal interaction for distributed applications. In *MULTIMEDIA '97: Proceedings of the fifth ACM international conference on Multimedia*, pages 31–40, New York, NY, USA, 1997. ACM Press. 12, 33

[27] R. Dale and E. Reiter. Computational interpretations of the Gricean maxims in the generation of referring expressions. *Cognitive Science*, 19(2):233–263, 1995. 81, 93

[28] K. Dautenhahn. Socially intelligent robots: dimensions of human–robot interaction. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 362(1480):679, 2007. 16

[29] J. P. de Ruiter, A. Bangerter, and P. Dings. The Interplay between Gesture and Speech in the Production of Referring Expressions: Investigating the Tradeoff Hypothesis. *Topics in Cognitive Science*, 2010. in press. 51

[30] W. Erlhagen and E. Bicho. The dynamic neural field approach to cognitive robotics. *Journal of Neural Engineering*, 3(3):R36–R54, 2006. 9, 94

[31] P. Fitzpatrick, G. Metta, and L. Natale. Towards long-lived robot genes. *Robotics and Autonomous Systems*, 56(1):29–45, 2008. 15

[32] M. E. Foster. The iCat in the JAST multimodal dialogue system. In *Proceedings of the First iCat Workshop*, Eindhoven, Mar. 2006. 10

[33] M. E. Foster, E. G. Bard, R. L. Hill, M. Guhe, J. Oberlander, and A. Knoll. The roles of haptic-ostensive referring expressions in cooperative, task-based human-robot dialogue. In *Proceedings of the 3rd ACM/IEEE International Conference on Human Robot Interaction (HRI 2008)*, pages 295–302, Amsterdam, Mar. 2008. 51, 81

[34] M. E. Foster, M. Giuliani, A. Isard, C. Matheson, J. Oberlander, and A. Knoll. Evaluating description and reference strategies in a cooperative human-robot dialogue system. In *Proceedings of the Twenty-first International Joint Conference on Artificial Intelligence (IJCAI-09)*, Pasadena, California, July 2009. 10, 23, 45, 81

[35] M. E. Foster, M. Giuliani, and A. Knoll. Comparing objective and subjective measures of usability in a human-robot dialogue system. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing (ACL-IJCNLP 2009)*, Singapore, Aug. 2009. 10, 23, 45, 81

[36] M. E. Foster, M. Giuliani, T. Müller, M. Rickert, A. Knoll, W. Erlhagen, E. Bicho, N. Hipólito, and L. Louro. Combining goal inference and natural-language dialogue for human-robot joint action. In *Proceedings of the 1st International Workshop on Combinations of Intelligent Methods and Applications, 18th European Conference on Artificial Intelligence*, Patras, Greece, July 2008. 8

[37] C. Geib, K. Mourao, R. Petrick, N. Pugeault, M. Steedman, N. Krueger, and F. Wörgötter. Object action complexes as an interface for planning and robot control. In *Proceedings of the IEEERAS International Conference on Humanoid Robots*, 2006. 19, 55

[38] B. Gerkey, R. Vaughan, and A. Howard. The player/stage project: Tools for multi-robot and distributed sensor systems. In *Proceedings of the 11th international conference on advanced robotics*, pages 317–323, 2003. 15

[39] J. Gibson. *The ecological approach to visual perception.* Lawrence Erlbaum, 1986. 19

[40] M. Giuliani. A basic system for interpretation of utterances in natural language, based on the combinatory categorial grammar. Master's thesis, Technische Universität München, 2006. 26, 62

[41] M. Giuliani, M. E. Foster, A. Isard, C. Matheson, J. Oberlander, and A. Knoll. Situated reference in a hybrid human-robot interaction system. In *Proceedings of the 6th International Natural Language Generation Conference (INLG 2010)*, Dublin, Ireland, July 2010. 10, 23, 93

[42] M. Giuliani, C. Lenz, T. Müller, M. Rickert, and A. Knoll. Design principles for safety in human-robot interaction. *International Journal of Social Robotics*, Mar. 2010. 72

[43] I. Harvey. Untimed and misrepresented: Connectionism and the computer metaphor. 1992. 115

[44] N. Hawes, J. Wyatt, and A. Sloman. An architecture schema for embodied cognitive systems. Technical Report CSR-06-12, University of Birmingham, School of Computer Science, Nov. 2006. 16, 41, 45

[45] M. Henning. A new approach to object-oriented middleware. *IEEE Internet Computing*, 8(1):66–75, 2004. 8, 15, 57

[46] H. Holzapfel, K. Nickel, and R. Stiefelhagen. Implementation and evaluation of a constraint-based multimodal fusion system for speech and 3d pointing gestures. In *ICMI '04: Proceedings of the 6th international conference on multimodal interfaces*, pages 175–182, New York, NY, USA, 2004. ACM Press. 16

[47] M. Johnston, S. Bangalore, G. Vasireddy, A. Stent, P. Ehlen, M. Walker, S. Whittaker, and P. Maloor. Match: An architecture for multimodal dialogue systems. *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 376–383, 2002. 12, 13

[48] M. Johnston, P. Cohen, D. McGee, S. Oviatt, J. Pittman, and I. Smith. Unification-based multimodal integration. In *Proceedings of the eighth conference on European chapter of the Association for Computational Linguistics*, pages 281–288. Association for Computational Linguistics, 1997. 22

[49] M. Kipp. Multimedia annotation, querying and analysis in ANVIL. *Multimedia Information Extraction*, 2010. 107

[50] D. Kirsh. Today the earwig, tomorrow man? *Artificial intelligence*, 47(1-3):161–184, 1991. 20

[51] N. Krüger, J. Piater, F. Wörgötter, C. Geib, R. Petrick, M. Steedman, A. Ude, T. Asfour, D. Kraft, D. Omrcen, et al. A Formal Definition of Object-Action Complexes and Examples at Different Levels of the Processing Hierarchy. *PACO+ Technical Report, available from http://www.paco-plus.org*, 2009. 19, 37, 112

[52] G.-J. M. Kruijff, P. Lison, T. Benjamin, H. Jacobsson, and N. Hawes. Incremental, multi-level processing for comprehending situated dialogue in human-robot interaction. In L. S. Lopes, T. Belpaeme, and S. J. Cowley, editors, *Symposium on Language and Robots (LangRo 2007)*, Aveiro, Portugal, Dec. 2007. 17

[53] T. K. Landauer, D. Laham, B. Rehder, and M. E. Schreiner. How well can passage meaning be derived without using word order? a comparison of latent semantic analysis and humans. In *Proceedings of the Nineteenth Annual Conference of the Cognitive Science Society: August 7-10, 1997, Stanford University*, page 412. Lawrence Erlbaum Associates, 1997. 17

[54] F. Landragin, A. Denis, A. Ricci, and L. Romary. Multimodal meaning representation for generic dialogue systems architectures. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC 2004)*, pages 521–524, 2004. 13

[55] S. Larsson and D. Traum. Information state and dialogue management in the TRINDI dialogue move engine toolkit. *Natural Language Engineering*, 6(3&4):323–340, Sept. 2000. 9, 22

[56] C. Lenz, M. Rickert, G. Panin, and A. Knoll. Constraint task-based control in industrial settings. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3058–3063, St. Louis, MO, USA, Oct. 2009. 72

[57] D. J. Litman and S. Pan. Designing and evaluating an adaptive spoken dialogue system. *User Modeling and User-Adapted Interaction*, 12(2–3):111–137, 2002. 80, 85, 92, 93, 100, 108

[58] P. Ljunglöf, G. Amores, R. Cooper, D. Hjelm, O. Lemon, P. Manchón, G. Pérez, and A. Ranta. Talk: Multimodal grammar library. *Deliverable D1. 2b, TALK Project*, 2006. 13

[59] N. Mohamed, J. Al-Jaroodi, and I. Jawhar. Middleware for robotics: A survey. In *2008 IEEE Conference on Robotics, Automation and Mechatronics*, pages 736–742, 2008. 15

[60] S. Möller, K.-P. Engelbrecht, and R. Schleicher. Predicting the quality and usability of spoken dialogue systems. *Speech Communication*, 50:730–744, 2008. 80, 93

[61] T. Müller and A. Knoll. Attention driven visual processing for an interactive dialog robot. In *Proceedings of the 24th ACM Symposium on Applied Computing*, Honolulu, Hawaii, USA, Mar. 2009. 8

[62] T. Müller, P. Ziaie, and A. Knoll. A wait-free realtime system for optimal distribution of vision tasks on multicore architectures. In *Proc. 5th International Conference on Informatics in Control, Automation and Robotics*, May 2008. 8

[63] M. Namoshe, N. Tlale, C. Kumile, and G. Bright. Open middleware for robotics. In *15th International Conference on Mechatronics and Machine Vision in Practice, Massey University, Auckland, New Zealand*, pages 2–4. Massey University, 2008. 15

[64] S. Oviatt. Ten myths of multimodal interaction. *Communications of the ACM*, 42(11):74–81, 1999. 13

[65] S. Oviatt. Multimodal interfaces. *The human-computer interaction handbook: fundamentals, evolving technologies and emerging applications*, 1:286–304, 2003. 13

[66] R. Petrick, D. Kraft, K. Mourao, C. Geib, N. Pugeault, N. Krüger, and M. Steedman. Representation and integration: Combining robot control, high-level planning, and action learning. In *Proceedings of the International Cognitive Robotics Workshop (CogRob 2008) at ECAI*, 2008. 19, 55

[67] R. Pfeifer, J. Bongard, and S. Grand. *How the body shapes the way we think: a new view of intelligence*. The MIT Press, 2007. 18, 48

[68] R. Pfeifer, F. Iida, and J. Bongard. New robotics: Design principles for intelligent systems. *Artificial Life*, 11(1-2):99–120, 2005. 119

[69] N. Pfleger. *Context-based multimodal interpretation: an integrated approach to multimodal fusion and discourse processing*. PhD thesis, Universität des Saarlandes, 2007. 13

[70] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng. ROS: an open-source Robot Operating System. In *International Conference on Robotics and Automation*, 2009. 15

[71] D. Schlangen and G. Skantze. A general, abstract model of incremental dialogue processing. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics (EACL-09)*, 2009. 17

[72] N. Sebanz, H. Bekkering, and G. Knoblich. Joint action: bodies and minds moving together. *Trends in Cognitive Sciences*, 10(2):70–76, 2006. 109

[73] L. Sentis and O. Khatib. Task-oriented control of humanoid robots through prioritization. In *Proceedings of the IEEE-RAS/RSJ International Conference on Humanoid Robots*, 2004. 54, 72

[74] R. Sharma, V. I. Pavlovic, and T. S. Huang. Toward multimodal human-computer interface. *Proceedings of the IEEE*, 86(5):853–869, 1998. 13

[75] C. L. Sidner, C. Lee, C. D. Kidd, N. Lesh, and C. Rich. Explorations in engagement for humans and robots. *Artificial Intelligence*, 166(1–2):140–164, 2005. 92

[76] A. Sloman. Some Requirements for Human-like Robots: Why the recent over-emphasis on embodiment has held up progress. *Creating Brain-like Intelligence*, pages 248–277, 2009. 18, 21

[77] M. Steedman. *The syntactic process.* MIT Press, Cambridge, MA, USA, 2000. 17, 25, 112

[78] A. J. N. van Breemen. iCat: Experimenting with animabotics. In *Proceedings of AISB 2005 Creative Robotics Symposium*, 2005. 7

[79] D. Vernon, G. Metta, and G. Sandini. A survey of artificial cognitive systems: Implications for the autonomous development of mental capabilities in computational agents. *IEEE Transactions on Evolutionary Computation*, 11(2):151–180, 2007. 14

[80] W. Wahlster. *SmartKom: Foundations of Multimodal Dialogue Systems.* Springer, 2006. 12

[81] M. Walker, C. Kamm, and D. Litman. Towards developing general models of usability with PARADISE. *Natural Language Engineering*, 6(3–4):363–377, 2000. 80, 93, 98, 100, 108

[82] M. A. Walker, J. Fromer, G. D. Fabbrizio, C. Mestel, and D. Hindle. What can I say?: Evaluating a spoken language interface to email. In *Proceedings of CHI 1998*, 1998. 92

[83] M. A. Walker, D. J. Litman, C. A. Kamm, and A. Abella. PARADISE: A framework for evaluating spoken dialogue agents. In *Proceedings of ACL/EACL 1997*, 1997. 80, 85

[84] M. White. Efficient realization of coordinate structures in combinatory categorial grammar. *Research on Language & Computation*, 4(1):39–75, 2006. 25

[85] M. White, M. E. Foster, J. Oberlander, and A. Brown. Using facial feedback to enhance turn-taking in a multimodal dialogue system. In *Proceedings of HCI International 2005*, 2005. 86

[86] F. Wörgötter, A. Agostini, N. Krüger, N. Shylo, and B. Porr. Cognitive agents – a procedural perspective relying on "predictability". *Robotics and Autonomous Systems*, 57:420–432, 2009. 19

[87] L. Wu, S. L. Oviatt, and P. R. Cohen. Multimodal integration - a statistical view. *IEEE Transactions on Multimedia*, 1(4):334–341, 1999. 12

[88] J. Wyatt and N. Hawes. Multiple workspaces as an architecture for cognition. In *Proceedings of AAAI 2008 Fall Symposium on Biologically Inspired Cognitive Architectures*, 2008. 15

[89] P. Ziaie, T. Müller, M. E. Foster, and A. Knoll. A naïve bayes classifier with distance weighting for hand-gesture recognition. In *Proceedings of the 13th International CSI Computer Conference (CSICC 2008)*, Kish Island, Iran, Mar. 2008. 8, 27

[90] P. Ziaie, T. Müller, and A. Knoll. A novel approach to hand-gesture recognition in a human-robot dialog system. In *Proceedings of the First Intl. Workshop on Image Processing Theory, Tools & Applications*, Sousse, Tunesia, Nov. 2008. 8, 27