Situated Interaction on Spatial Topics

Dissertation
zur Erlangung des Grades
Doktor der Ingenieurwissenschaften (Dr.-Ing.)
der Naturwissenschaftlich-Technischen Fakultät I der Universität des Saarlandes

vorgelegt von

Christian Kray

Saarbrücken
May 30, 2003

Datum des Kolloquiums:
8. Juli. 2003
Dekan:
Prof. Dr.-Ing. Philipp Slusallek
Vorsitzender:
Prof. Dr.-Ing. Philipp Slusallek
Gutachter:
Prof. Dr. Dr. hc. mult Wolfgang Wahlster
Prof. Christian Freksa, Ph.D.

Eidesstattliche Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus anderen Quellen oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet.

Die Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder ähnlicher Form in anderen Prüfungsverfahren vorgelegt.

Saarbrücken, den 1. Mai 2003

Danksagung

Diese Arbeit verdankt ihre Entstehung einer Reihe von Personen und Institutionen, denen ich an dieser Stelle meinen Dank aussprechen möchte. Meinem Doktorvater Herrn Professor Wahlster danke ich für die Möglichkeit an seinem Lehrstuhl zu promovieren und dafür, dass er trotz seiner vielen anderen Tätigkeiten Zeit fand, diese Arbeit zu betreuen. Herrn Prof. Freksa danke ich für seine Bereitschaft, als Zweitkorrektor die Begutachtung zu übernehmen.

Fast über die gesamte Dauer meiner Promotion wurde ich von der Klaus Tschira Stiftung mittelbar oder unmittelbar gefördert. Für die Förderung im Rahmen beider mit meiner Promotion zusammenhängenden Projekte, MapTalk und SISTO, möchte ich mich auch bei Herrn Tschira persönlich bedanken. Diese Arbeit wurde zu einem Teil auch von der Deutschen Forschungsgemeinschaft (DFG) im Rahmen des Sonderforschungsbereiches 378 gefördert.

Eine Arbeit wie die vorliegende entsteht nie im leeren Raum. Für die gute Zusammenarbeit danke ich den Mitarbeitern am European Media Lab und an anderen Instituten, die mit mir Deep Map verwirklicht haben. Ebenso möchte ich mich bei meinen Kollegen am Deutschen Forschungszentrum für Künstliche Intelligenz sowie am Lehrstuhl von Professor Wahlster bedanken. Darüber hinaus bin ich den Koautoren der Veröffentlichungen, an denen ich beteiligt war, dankbar für inspirierende Gespräche und die Ausarbeitung vieler Ideen. Besonders danke ich den Lesern der vielen Vorversionen dieser Arbeit für ihre konstruktive Kritik: Christian Müller, Jörg Baus, Antonio Krüger, Michael Schillo und Jochen Könemann.

Meinen Eltern danke ich dafür, dass ich überhaupt zu dem Punkt gelangen konnte, eine Promotion anzugehen. Der größte Dank gebührt jedoch Andrea, Lucien und Emma, für ihre Liebe, Geduld und Unterstützung. Ohne Euch wäre dies alles nicht möglich gewesen. Danke.

Christian Kray im April 2003

Acknowledgements

Several people and institutions were involved in process of creating this work, and I would like to express my gratefulness towards them here. Professor Wahlster provided me with the chance to work on my PhD at his chair, and I would like to thank him for finding the time to advise me in spite of the tremendous amount of his other duties. Professor Freaks agreed to become my second advisor, for which I am very grateful.

The Klaus Tschira Foundation (KTS) supported me – either directly or indirectly – throughout nearly the entire time I was working on my PhD. I would like to express my gratitude for providing me with two project grants (MapTalk and SISTO), not only towards the KTS but also towards Klaus Tschira personally. The work has also been partially supported by the Deutsche Forschungsgemeinschaft (DFG) within the Collaborative Research Center 378.

A thesis such as this one is never realized in the void. I would therefore like to thank my colleagues at the European Media Lab as well as those at other institutes, who worked hard with me to realize Deep Map, for the good cooperation. The same applies to my coworkers at the German Research Center for Artificial Intelligence and at the chair of professor Wahlster. I also want to thank the coauthors of the publications that I was involved in for inspiring conversations and for jointly working out many ideas. I am especially thankful for the constructive criticism from the readers of the many draft versions of this thesis: Christian Müller, Jörg Baus, Antonio Krüger, Michael Schillo, and Jochen Könemann.

Finally, I owe much to my parents for ever getting to the point of undertaking a PhD. However, most of all I want to thank Andrea, Emma, and Lucien for their love, patience, and support. Without you, all of this would have never been possible. Thank you.

Christian Kray in April 2003

Kurzzusammenfassung

In dieser Arbeit stellen wir ein Modell zur Verarbeitung situierter Interaktionen über raumbezogene Sachverhalte und seine Implementation vor. Außerdem präsentieren wir verschiedene Strategien zum Umgang mit häufigen Problemen, die im Zusammenhang mit dem (mobilen) Einsatz von Systemen im realen Umfeld auftreten. Das zu Grunde liegende Modell bezieht situationsbezogene Faktoren auf unterster Ebene mit ein und erleichtert durch den modularen Aufbau seinen Einsatz im Rahmen verschiedener Anwendungen. Die entsprechende Implementation spiegelt die Struktur des Modells wider und wurde im Rahmen eines mobilen Touristenführers getestet. Die ebenfalls vorgestellten Adaptionsstrategien dienen unter anderem zur Behandlung von Informationsmangel und von Ressourcenbeschränkungen sowie zum Umgang mit dem Problem variierender Verfügbarkeit und Qualität von Positionsinformation.

Short abstract

In this thesis, we present a model and an implementation to handle situational interactions on spatial topics as well as several adaptation strategies to cope with common problems in real-world applications. The model is designed to incorporate situational factors in spatial reasoning processes at the basic level and to facilitate its use in a wide range of applications. The implementation realizing the model corresponds very closely to the structure of the model, and was put to test in a scenario of a mobile tourist guide. The adaptation strategies address the lack of information, resource restrictions as well as the problem of varying availability and quality of positional information.

Zusammenfassung

Der Interaktion zu raumbezogenen Sachverhalten kommt nicht nur im Kontext mobiler bzw. situierter Systeme eine große Bedeutung zu. Auch in anderen Bereichen wie z. B. der natürlichsprachlichen Steuerung von Karten oder Benutzerschnittstellen spielt diese eine wichtige Rolle. Ein Ziel dieser Arbeit ist daher der Entwurf eines praxistauglichen und breit anwendbaren Modells für die situierte Interaktion zu raumbezogenen Sachverhalten. Darüber hinaus untersuchen wir typische Probleme, die im mobilen Realeinsatz auftreten, und zeigen Strategien auf, um mit diesen Problemen umzugehen.

Um den Hintergrund und Kontext zur Einordnung der vorgelegten Arbeit zu erhellen, stellen wir zunächst einige grundlegende Konzepte und Definitionen aus dem Gebiet der Raumkognition sowie der Situations- und Aufgabenmodellierung vor und diskutieren deren Zusammenhang. Die hierbei erörterten Punkte umfassen unter anderem den Raumbegriff, räumliche Referenzsysteme und Relationen aber auch Benutzer- und Kontextmodelle. Danach geben wir einen Überblick über verwandte Arbeiten vor allem aus dem Gebiet der Navigationsunterstützung und erörtern deren Vor- und Nachteile im Hinblick auf die Ziele dieser Arbeit. Dieser Vergleich schließt auch das im Rahmen dieser Arbeit entwickelte prototypische System (SISTO) mit ein, das das im Folgenden vorgestellte Modell implementiert.

Basierend darauf präsentieren wir im Hauptteil einen Ansatz für die Modellierung situierter Interaktion über raumbezogene Aspekte. Das resultierende Modell besitzt verschiedene Eigenschaften, die in mehrerlei Hinsicht über bisherige Ansätze hinausgehen. Zum einen werden dort situationsbezogene Faktoren – wie kontextuelle und benutzerspezifische Faktoren – auf unterster Ebene und in Abhängigkeit der zu bearbeitenden Aufgabe mit einbezogen. Das Modell wurde so entworfen, dass die Integration weiterer Faktoren sowie die Anpassung ihres jeweiligen Einflusses einfach zu realisieren ist und somit die Einbeziehung neuer empirischer Erkenntnisse vereinfacht. Ein weiterer erwähnenswerter Aspekt in diesem Zusammenhang ist die systematische Analyse induzierter Referenzsysteme. Zum anderen erlaubt der modulare Aufbau unseres Modells die einfache Modellierung komplexer raumbezogener Aufgaben aus Teilmodellen für grundlegende Aufgaben bzw. aus denen anderer komplexer Aufgaben. So beinhaltet das Modell Komponenten zur Evaluation von Objekten im Kontext einer spezifischen Aufgabe und Situation, zur Etablierung eines Referenzsystems, zur Berechnung von Zwei- und N-Punkt-Relationen sowie zur Segmentierung von komplexen Trajektorien. Zudem umfasst der vorgestellte Ansatz ein weitgehend sprach- und modalitätsunabhängiges Format zur Repräsentation raumbezogener Interaktionen, mit Hilfe dessen wir alle Interaktionen zwischen Benutzer und System enkodieren konnten.

Ein zweiter wesentliche Beitrag dieser Arbeit besteht in der Vorstellung verschiedener Adaptionsstrategien für den Realeinsatz eines Systems, das die situierte Interaktion über raumbezogene Sachverhalte erlaubt. Dabei erläutern wir zunächst den Umgang mit fehlender Information innerhalb unseres Modells und zeigen Möglichkeiten auf, wie das Problem beschränkter Ressourcen angegangen werden kann. Aufgrund der großen Bedeutung von Positionsinformation im Kontext situationssensitiver Systeme stellen wir dann einen umfassenden Ansatz zur deren Bestimmung und Verarbeitung vor. Die dazu vorgestellten Strategien umfassen Verfahren zur Inferenz, Unsicherheitsreduzierung und Exploration von Positionsinformation sowie Adaptionsmöglichkeiten im Rahmen verschiedener zuvor untersuchten Aufgaben. Das dazu neu entwickelte Verfahren zur dynamisch optimierenden Generierung von Fragen zur Bestimmung der Benutzerposition erlaubt erstmals die schnelle Lokalisierung des Benutzers selbst in Abwesenheit jeglicher Messwerte. Dabei werden die Fragen nicht nur im Hinblick auf den Informationsgewinn optimiert sondern auch im Hinblick auf die Minimierung der Interaktionsdauer. Dieser Prozess wird nach Erhalt einer Antwort neu angestoßen, um die Fragen hinsichtlich des aktuellen Wissensstands zu optimieren.

Die Praxistauglichkeit unseres Ansatzes illustrieren wir im Folgenden mit Hilfe der prototypischen Implementierung, die in zwei unterschiedlichen Gastsystemen zum Einsatz kam. Das von uns realisierte System, SISTO, basiert auf einem Multiagentensystem, entspricht in struktureller Hinsicht sehr stark dem zu Grunde liegenden Modell und erlaubt die einfache Erweiterung und Einbindung in andere Gastsysteme. Neben seinem modularen Aufbau und seiner einfachen Erweiterbarkeit und Integrierbarkeit realisiert die Implementierung erstmalig das oben erwähnte interaktive Verfahren zur Positionsbestimmung. Außerdem unterstützt es die Generierung optimierter Lokalisierungsphrasen, die auf induzierte Referenzsysteme zurückgreifen. Darüber hinaus stellen wir eine generische Architektur zur Behandlung von Positionsinformation vor, die auf einem dreischichtigen Ansatz beruht und Messwerte, eine Positionshistorie sowie Inferenz- und Interaktionsmechanismen integriert. Anhand einer beispielhaften Interaktion mit dem System zeigen wir dessen Fähigkeiten auf.

Zum Abschluss der Arbeit fassen wir die hier erzielten Ergebnisse noch einmal stichpunktartig zusammen. Im Anschluss daran erörtern wir kurz, welche zukünftige Forschungsrichtungen sich daraus ergeben können.

Abstract

The interaction on spatial topics is highly important no only in the context of mobile and situated systems but also in other fields such as natural language access to maps or user interfaces. Therefore, one goal of this thesis is to develop a generic model for situated interaction on spatial topics that can be used to build real world applications. In addition, we analyze typical problems that arise in the context of mobile real world applications, and point out strategies for coping with them.

In order to provide background information, we first review basic concepts and definitions from the field of spatial reasoning as well as from the modeling of situations and tasks, and highlight their relationship. This discussions includes topics such as scale, spatial frames of reference and relations as well as user and context models. Then, we review a selection of related work in the context of navigational assistance, and analyze their advantages and shortcomings with respect to the goal of this thesis. This comparison also includes the prototypical implementation of the approach presented hereafter.

Based on this analysis, we present an approach for the modeling of situated interaction on spatial topics in the main part of this work. This model incorporates several features that go beyond previous approaches: Situational factors – such as context- or user-related factors – are taken into account at the lowest level with respect to the current task. We designed the model in a way that supports the inclusion of further factors and the adaptation of their impact in context of a specific task. An additional noteworthy feature is the underlying systematic analysis of induced frames of reference. Furthermore, the model is highly modular and therefore facilitates the modeling of complex task by combining partial models for basic processes and/or other complex tasks. It incorporates components for the evaluation of objects in the context of a specific task and situation, for the establishment of spatial frames of reference, for the computation of two- and n-point relations as well as for the segmentation of complex trajectories. In addition, our approach relies on a representation for spatial interaction that is largely independent of the target language or modality. We used this format to encode all interactions between the user and the system.

The second major contribution of this thesis consists of a set of adaptation strategies for systems that allow for situated interaction on spatial topics in a the real world setting. We first introduce means to address the lack of information as well as resource restrictions within the previously presented model. Due to the general relevance of positional information in the context of systems that are aware of the current situation, we then present a comprehensive approach to determine and process the positional information. The corresponding strategies include approaches based

on inference, reduction of uncertainty and exploration as well as ways to adapt the complex tasks that we reviewed earlier. We developed a new algorithm to determine the user's current position through the dynamically optimizing generation of questions aimed at quickly localizing her – even in the absence of any sensor data. The questions are not only optimized towards the potential information gain but also in terms of minimizing the duration of the corresponding interaction. In order to achieve this goal, the generation process is triggered after each reply from the user.

In order to illustrate the suitability of our model for real world applications, we provide a detailed description of a prototypical implementation in the following. SISTO – the system we realized – has been used within two different host systems. It is based on a multi-agent system, and its structure corresponds very closely to the underlying model. Due to this architecture, it is a straightforward task to extend SISTO or to embed it into further host systems. The implementation not only the first one to realizes the above mentioned algorithm for the interactive determination of the use's current position but also the first one to generate localizations based on induced frames of reference. In addition, we present a generic approach for handling positional information that relies on a three level approach, which integrates sensor data, a position history, and inferential and interactive mechanisms. In order to illustrate the features of SISTO, we then follow a person using the system on an example journey through the city of Heidelberg.

The thesis on concludes on a summary of the results and achievements, and on an outlook of future research opportunities.

Contents

List of Figures

List of Tables

1 Introduction

... burning with curiosity, she ran across the field after that strange rabbit and followed it into the rabbit-hole. Tumbling down she fell, surprised by the depth of the hole. While she was falling, she noticed cupboards and book-shelves at the sides of the well. On one of them, she saw a small device, which lit up as she passed:

"TAKE ME!"

it said in a pleasant voice. So, she took it in a blink of an eye. Down, down, down she went in what seemed to be an endless fall.

"I wonder how many miles I've fallen by this time?"

she said to herself. To her surprise she heard the device in her hand answer her question:

"Oh, already tens of miles. In less than a minute, you will land safely."

Somewhat relieved now, Alice was glad as she indeed landed on some dry leafs shortly thereafter. She caught a glimpse of the rabbit and hurried not to loose it. Then she came to big hall with doors all around it. She did not see the rabbit anymore, and she did not know which way it had gone. As she was standing there wondering what to do next, the little device suddenly spoke to her again:

"If you want to get out of this hall, there is a key on that glass table..."

"But where am I?", she interrupted.

"You're in Tomorrowland." the device replied.

(a slightly modified excerpt from 'Alice in Wonderland' [Carroll, 1946])

This modified little excerpt from Carroll's famous fairy tale illustrates several central points about the topic of this thesis. On the one hand, it shows that such an intelligent, knowledgeable and adaptive device still belongs to the realm of (science) fiction. On the other hand, we can use this story to demonstrate several key issues and features of such a device, and a subset of these will be part of the model and implementation presented in later chapters. Either way, the device described in the fairy tale seems to perform quite well when interacting about spatial topics in various situations.

Therefore, let us analyze the story in terms of what it takes for a device to behave in the way it does. When Alice passes the shelf on which the assistant is located, its screen lights up and displays a short message. This is arguably a better way to draw Alice's attention to the device than spoken output, as it stands out visually while audio output is not as easy to locate if there are other objects on the shelf. Especially, if we assume that Alice is 'traveling' at a considerable speed, we have to take into account factors such as time restrictions both on Alice's and on the devices side as well as her viewing direction.

Similarly, only a multi-factorial reasoning process can produce an answer such as the one Alice asks herself. Note that the device not only recognizes the implicit question of when Alice will land, but also replies to the explicit question. It does so in a way a young girl can understand: it produces a rough description instead of giving the exact speed and/or distance to the surface in meters. That kind of adaptation requires, for example, the inclusion of Alice's age and (true) intention in the process generating the reply.

This applies even more to the third interaction where the device proactively informs Alice about her current options. And when Alice interrupts the output by asking where she is, the corresponding reply can only be generated if we take into account her familiarity with the current environment as well as the level of granularity in terms of spatial descriptions.

These few examples illustrate the impact of situational factors in the interaction on spatial topics. While the field of spatial reasoning and the systems developed therein have come a long way [Knauff et al., 2002], most models and systems neglect the relevance of situational factors by either ignoring them altogether, or by incorporating only a small number of them. The inclusion of these factors into spatial reasoning processes consequently constitutes a main goal of the work presented in this thesis.

1.1 Aims and methods

The story presented in the previous section and its analysis already hint at the interdisciplinary nature of this thesis. In order to attain our goal of modeling situated interaction on spatial topics, we combine findings and methods from several disciplines including artificial intelligence, computer science, (cognitive) psychology, and (computational) linguistics. Hence, we can classify this work as belonging to the field of *cognitive science.* The interdisciplinary and highly interrelated nature of situated interaction on spatial topics may be one reason why most models and systems handling spatial information focus on well-defined subsets of spatial problems, e. g. spatial relations (as we will point out in chapter 2 and 3). Furthermore, only a few models take into account situational factors such as weather conditions or the user's properties and abilities. Even fewer (if any) approaches provide means to cope with issues arising in real world applications, e. g. missing or unavailable information.

However, most if not all of these features are of central importance not only for stationary applications such as intuitive natural language access to geographic information systems (GIS) but even more so in a mobile setting. The quality of location based services (LBS) – for example, navigational assistance or localized recommendation services – depends on its robustness against outages of various types (e. g. network connection, access to databases), its adaptation to the situation of the user (e. g. her current position), and its flexibility to cope with complex interactions (such as incremental guidance). From these considerations, we can derive several questions and issues that we have to address in order to model and support situated interaction on spatial topics. Hence, the goal of this thesis is to answer the following questions:

- **What situational factors can have an impact on spatial reasoning?**
  The example presented in the previous section already hints at the great influence of the situation on the interaction on spatial topics, and we will present further evidence throughout this thesis. Relevant factors are not only related to the current user of a system but also include contextual aspects such as the current means of transportation, which can be shared by several people.

- **How can we integrate spatial reasoning and situational factors into a coherent and extensible model?**
  Due to the omnipresent nature of space, the number of tasks and processes related to space is too large to allow for a detailed description and modeling of all of them. Hence, a useful model should be modular and provide means for easy extension both in terms of additional factors and further tasks and processes.

- **How can we explicitly model and encode interactions?**
  Since it is our goal to handle situated *interaction* on spatial topics, we obviously have to investigate how to represent the interaction between a user and a system. A suitable encoding scheme should provide means to represent interactions independent of the natural language (or modality) used to communicate. In order to facilitate the extension of the model that we intend to design, the interactions between different parts of the model should be made explicit as well.

- **What are key problems in situated interaction on spatial topics, and what strategies can help to adapt to them?**
  If we intend to design a model for building real-world applications, it is not sufficient to develop an approach that integrates situational factors and interaction on spatial topics. We also have to investigate, which problems may arise in a real-world setting, and how we can cope with it. Hence, we have to design adaptation strategies to handle these issues within the model.

- **How can we best deal with positional information?**
  Mobile applications such as location based services constitute main areas, where situated interaction on spatial topics may occur. In this context, knowing the user's current position is a key factor for determining the situation. Therefore, it makes sense to investigate how to best deal with positional information, both on a conceptual (e. g. using various inferential algorithms) and a practical level (e. g. compensating sensor deficiencies).

- **What is a suitable way to implement the proposed approach?**
  While the design of a comprehensive model for situated interaction on spatial topics constitutes a major step forward towards intelligent user interfaces, it is also important to consider how we can realize a corresponding implementation. Furthermore, a working implementation will add support for the validity of our approach.

The following section presents an overview over the structure, which we will follow in presenting our results for the points listed above, and it will also give a general overview over this thesis.

1.2 Outline

In addition to the scientific aims presented in the previous section, there is another important goal we pursued in writing this thesis: We wanted to provide the reader with a clear structure that facilitates various entry points and browsing. Therefore, the following chapters (except the final one) adhere to the same basic layout: A short introduction presents an overview on the content of the chapter and included sections. These sections discuss the main points of the chapter in detail. At the end of each chapter, there is a short summary that reviews the most relevant points of the chapter. Figure 1.1 depicts the basic structure underlying this thesis.

Therefore, we recommend several possible tracks for reading depending on the main intention of the reader. If the main goal is to obtain a more detailed overview than the one provided in the conclusion chapter, we suggest the reading of the summary sections. Readers familiar with spatial reasoning can skip the first sections of chapter 2 and possibly chapter 3 to proceed directly to the main chapters 4 and 5. For all other 'selective' readers, the summary sections should provide a good starting point for deciding whether or not the contents of the corresponding chapters match their respective interests. The short index at the end of this thesis should also help in locating topics of specific interest.

In the chapter following this introduction, we review some basic concepts that are vital for the understanding of the central points, and we define the terms used throughout this thesis. Topics

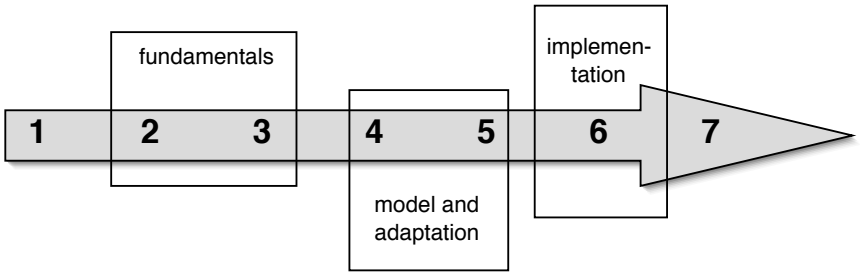


Figure 1.1: Structure of this thesis

discussed here include spatial concepts, cognitive and technical resources, situational, user-related, and contextual factors. Furthermore, we identify several typical tasks related to space, analyze the underlying processes, and then determine a set of common subtasks. These subtasks serve as the starting point for the model proposed in the main part of this thesis.

Chapter 3 presents work related to this thesis, including systems that provide navigational assistance and various approaches to different problems in spatial reasoning. We then compare these systems and approaches along several dimensions and analyze their respective advantages and shortcomings. The comparison also includes the implementation, which realizes the model presented in the following chapter. This concludes the discussion of the fundamentals.

Chapter 4 and 5 present the main contributions of this thesis. We introduce a modular approach to situation-aware spatial processes based on multi-attribute utility theory, and a hybrid model for spatial concepts that takes into account situational factors. Furthermore, we propose a hierarchical process-oriented method to model complex tasks related to space, and we present an encoding schema that allows for language- and media-independent interaction. We then present several adaptation strategies, which address various problems arising in real-world applications such as missing information and resource restrictions. Another key point is the handling of positional information, for which we propose an approach based on measurements, inference, and interaction. We conclude the chapter with an analysis of the potential for adaptation with respect to the interaction with the user.

In chapter 6, we describe a prototypical implementation (SISTO) of the approach presented previously. After reviewing some general requirements, we shortly introduce the two host systems (Deep Map and SmartKom), which were used to test SISTO. We then give a detailed description of the internal architecture and the interactions taking place when processing various tasks, before we propose a generic approach for handling positional information and its propagation. We conclude the chapter by following a tourist on a trip through Heidelberg in order to demonstrate the features and capabilities of SISTO. Chapter 7 summarizes the achievements of this thesis and highlights some possibilities for future research. In the appendix, we present a series of experiments aimed at the identification of the semantics of two path prepositions, and we provide detailed information on the encoding of the interactions between SISTO and its user.

2 Basic concepts

Before we review related work and before we present the main results of this thesis, we need to introduce some basic concepts that we will use throughout this work. The title of this thesis – situated interaction on spatial topics – already highlights the two key areas of interest: spatial concepts and situational factors. Consequently, we will first review the former one in section 2.1, and then the latter one in section 2.2. However, since the user's current position and the available resource are complex and highly relevant situational factors, we analyze them separately in section 2.3 and 2.4.

One goal of the work presented in this thesis lies in the design of a modular model that facilitates extension and that is applicable to different domains. In order to choose the 'right' building blocks, we then analyze some typical tasks related to space in section 2.5. The chapter concludes on a discussion of the interrelations between spatial concepts and situational factors as well as problems arising during the integration of these (section 2.6).

2.1 Spatial concepts

Although (or maybe because) spatial concepts play an important role in everyday life as well as in various sciences (e. g. mathematics, physics, geography, or psychology), there is no single definition of what exactly is meant when talking about *space* or about important concepts such as spatial relations. Therefore, we need to define the terms used throughout this thesis (cf. 2.1.1) and to review relevant approaches related to space (cf. 2.1.2, 2.1.3).

2.1.1 Space

Space (and time) are very fundamental and omnipresent for almost all human behavior and reasoning [Freksa, 1997]. Yet, it is difficult (or impossible) to find a single definition that covers all aspects of space. For example, while in mathematics the term *space* is frequently used to describe the dimensionality of sets or vectors [Bronstein and Semendjajew, 1979], this has little to no relevance in the context of human behavior. However, since computers heavily rely on mathematical concepts, oftentimes spatial knowledge is stored in cartesian coordinates (e. g. in geographical information systems (GIS)). This somewhat contradicts the naive conception most people have of space as being the physical environment, in which we live and act.

In psychology and other disciplines such as architecture, (everyday) space is often seen as being structured and hence being perceived differently according to its *scale* (see, for example, [Lynch, 1960]). After initial approaches which introduced a binary portioning (e. g. [Ittelson, 1973, Downs and Stea, 1977]), the distinction between *small- and large-scale spaces* has been further refined [Freundschuh and Egenhofer, 1997]. Montello [Montello, 1993], for example, distinguishes four main categories:

- **figural space**
  This encompasses the space within the direct reach of a person, which is smaller than the body of the observer. Another term that is frequently used to describe this kind of space, is *table-top space* [Ittelson, 1973].

- **vista space**
  The space that can be perceived visually from a single location without locomotion falls in this category. For example, the room a person is located in lies in vista space.

- **environmental space**
  If a portion of space cannot be perceived from a single location without the observer moving around, it can be classified as belonging to environmental space. A city is an example for an entity existing in this type of space.

- **geographical space**
  Montello defines geographical spaces as spaces that cannot be apprehended even with extensive knowledge but have to be reduced to figural or vista space in order to do so. This is the space of countries or continents.

These different types of spaces are closely related to how humans encode and memorize spatial information such as constellations or routes, which we review in the following section.

2.1.2 Spatial knowledge

When humans explore space they not only perceive it but they build up a mental representation of it (cf. [Tversky, 1993]). Similarly to other cognitive processes, there is no unique representation format that is used all the time but rather several of them: Different ones are used on different occasions.[1] They can complement one another neatly, but can sometimes also encode contradicting information. Generally, we can distinguish three classes of spatial knowledge: *landmark knowledge, route knowledge* and *survey knowledge* [Werner et al., 1997].

Landmarks are objects, which are embedded in the environment and which differ from other objects in their vicinity in one or more respects such as visual salience and/or conceptual salience (see, for example, [Sorrows and Hirtle, 1999]). Since they 'stand out' from their environment they are not only easy to remember but also easy to recognize. Therefore, they are highly relevant in a number of spatial processes such as object localization [Gapp, 1995] or wayfinding [Lynch, 1960, Raubal and Worboys, 1999]. Landmark knowledge actually links specific landmarks to other knowledge. For example, by associating a turn instruction with a landmark at a decision point, a person can decide which path to follow in order to get to her target location.

Route knowledge (also known as *procedural knowledge*) is most frequently gained from actively exploring the environment. (Alternatively, people can acquire route knowledge indirectly, e. g. by listening to route instructions.) Route knowledge consists of a series of spatial actions such as turning or following a road, which together form a route from one location to another.

Survey knowledge encodes information about the topology and/or spatial constellations in an area. People mainly acquire survey knowledge by extensively exploring a region of space, which enables them to establish multiple relationships between various locations within that area. Maps also represent survey knowledge, and hence, support the acquisition of survey knowledge. The main difference between survey knowledge and the two other categories lies in the way in which knowledge is organized: survey knowledge abstracts from single experiences and observations to form an integrated model.

The amalgamation of spatial knowledge that is encoded in different ways forms the basis for human spatial reasoning, and is often defined as a *cognitive map* [Tolman, 1948] or *cognitive collage* [Tversky, 1993]. It is important to highlight that these cognitive maps do not result from a homomorphic mapping of the real world to a representation, but that they are a conglomeration of possibly contradicting pieces of information. Nevertheless, they enable humans to efficiently store spatial information and to interact with space in a meaningful way most of the time.

2.1.3 Spatial relations

Not only do humans act within space they also *talk* about it or *refer* to it verbally or by other means such as gestures. A frequent means to realize spatial references consists of *spatial relations* [Herrman and Grabowski, 1994]. A spatial relational expression consists of three main parts. The relation itself constitutes the first part, and in the following paragraphs we will review what different types there are. The second part is the *anchor object*, which is also called reference object or

[1] It is possible that there are also gender-specific differences in terms how information is encoded (see, for example, [Fontaine and Denis, 1999]).

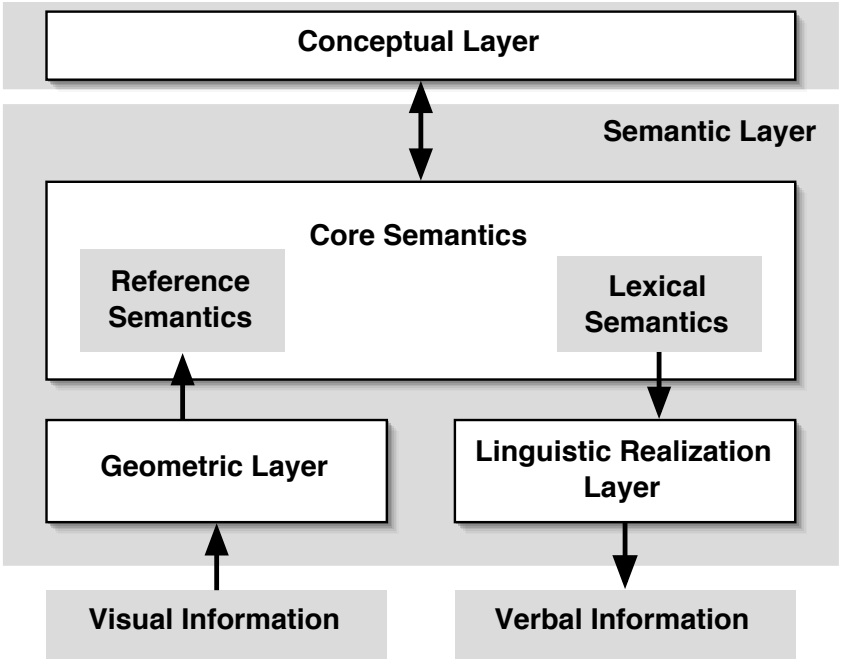


Figure 2.1: Gapp's three-level semantics for spatial descriptions (adapted from [Gapp, 1997]).

relatum.[2] This entity defines the origin (or ground) for the relation. The *target object* (also called 'object to be localized') is the third part of a spatial relational expression; this is the object that is localized or spatially related to the anchor object. Some spatial relations such as `in-front-of` additionally require a *frame of reference*, a structuring of space, which we will review in detail in 4.1.2. The whole expression can also be graded with a *degree of applicability* (DA) [Schirra, 1994] that captures how well a relation applies to the situation it describes.

A relevant distinction in this context is the one between a *spatial relation* as a semantic concept and its corresponding *lexical instantiation* (see, for example, [Herskovits, 1986, Gapp, 1997]). While a spatial relation encodes a specific spatial meaning, it is independent of the target language that it is expressed in. For example, a spatial relation such as `near` can be realized in German using "nahe", "bei", "an" or "neben", or in English using "nearby", "close to", or "at". These are just a few examples for possible realizations; further alternatives include adjectives (e. g. "neighboring") or phrases (e. g. "which border on it"). Gapp [Gapp, 1997] does account for this distinction by introducing the semantic layers in his computational model for spatial descriptions (see figure 2.1). He defines realization layer (consisting of a geometric and and a linguistic layer), on which the core semantics layer is based. The latter has two components: the reference semantics rooted in the geometric layer and the lexical semantics relying on the linguistic realization layer. Contextual and world knowledge on the conceptual layer informs the translation processes on the semantic layer.

[2] Some relations such as `inbetween` require more than one anchor object [Habel, 1989].

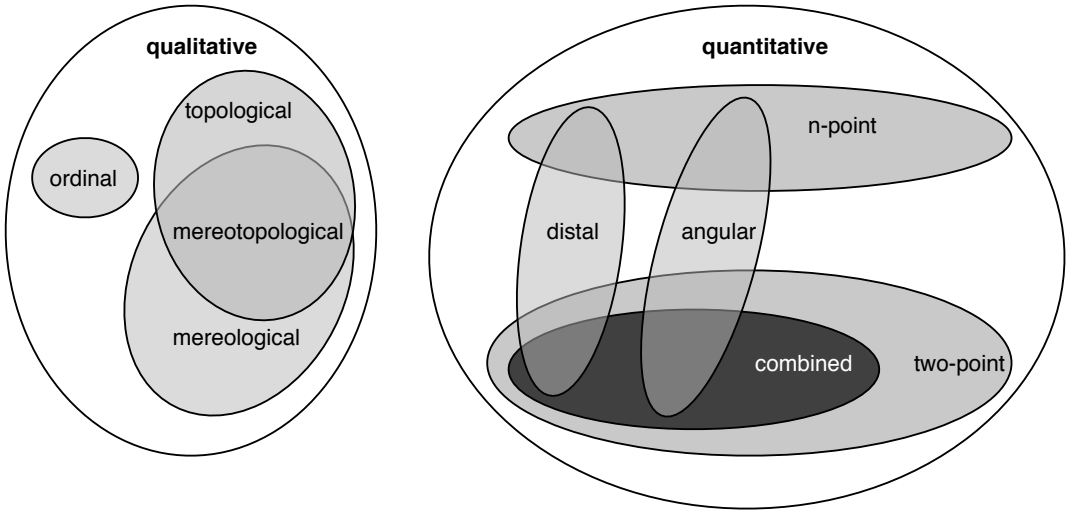


Figure 2.2: Classifications for spatial relations.

Several (partially overlapping) classifications have been proposed for spatial relations (e. g. [Gapp, 1997, Mukerjee, 1997, Masolo and Vieu, 1999, Kray and Blocher, 1999]) and figure 2.2 tries to summarize these different classifications.[3] A very fundamental distinction in this context is that of *qualitative* and *quantitative* relations. While the latter ones are inherently graded concepts relying on continuous or discreet measures, the former abstract away from those measure *by collapsing 'indistinguishable' values into an equivalence class* [Cohn, 1996].

We can further subdivide the category of qualitative relations. The group of topological relations [Egenhofer and Franzosa, 1991] (such as `inside` or `overlaps`) includes spatial relations that are unaffected by elastic deformations of the two related entities. Mereological relations such as `part-of` were introduced by Lesniewski [Leśniewski, 1931] (qtd. in [Masolo and Vieu, 1999]) and capture relations between parts. Mereotopological relations [Masolo and Vieu, 1999] combine topological and mereological notions to introduce relations such as `being-connected-with`. *Ordinal relations* are another group of qualitative relations that is based on partially ordered sets (see [Kainz et al., 1993] for an overview). An influential theory in qualitative spatial reasoning, the Region Connection Calculus, and the subset of eight jointly exhaustive and pairwise disjoint relations called RCC8 rely on topological concepts (see, for example, [Randell et al., 1992, Cohn, 1996, Renz and Nebel, 1999]). These relations are shown in figure 2.3 as well as the direct transitions between them (which define their 'conceptual neighborhood' [Freksa, 1992]).

The group of quantitative relations can also be divided into several (partially overlapping) subclasses. Gapp [Gapp, 1997] has proposed a subdivision into three main categories: *distal relations*, *angular relation* and *special relations*. Distal relations capture spatial concepts related to the distance between anchor object and target object such as `next-to` or `far-from`.[4] Angular rela-

[3] The diagram in the figure is a qualitative representation, i. e. the size of overlapping regions is not significant.

[4] Gapp does not distinguish between topological and distal relations.

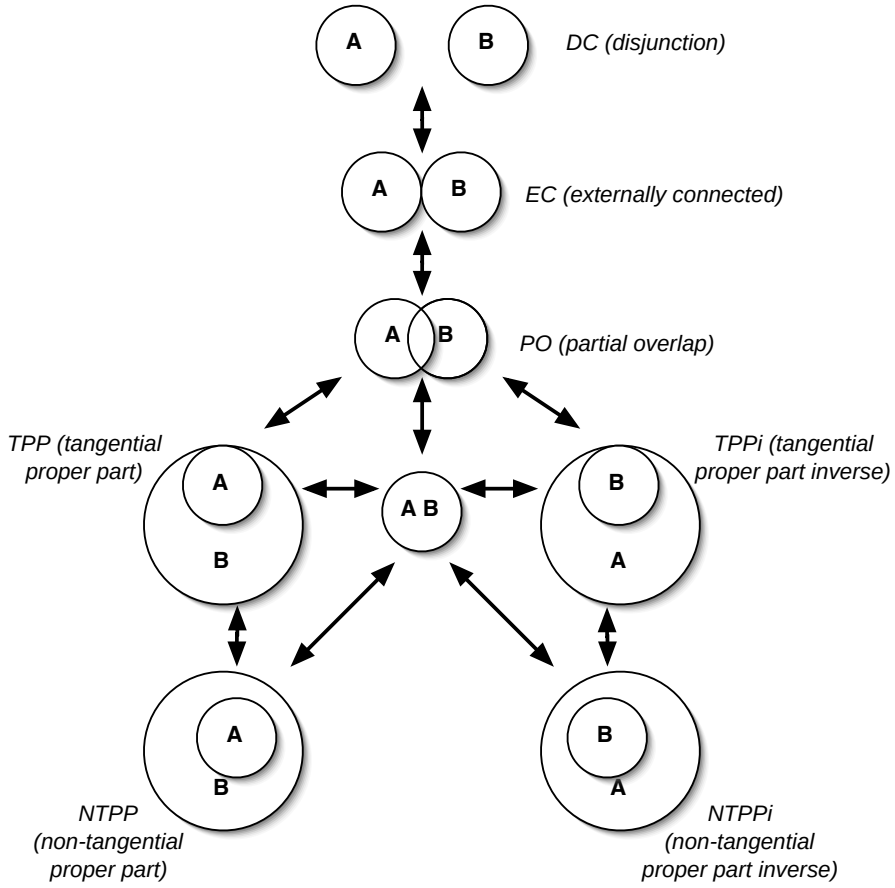


Figure 2.3: RCC-8 relations and transitions – arrows indicate direct transitions between states.

tions express the angular disparity of anchor and target object with respect to a frame of reference, for example `left-of` or `in-front-of`. Gapp further subdivides angular relations into projective relations and geographical relations (such as `north-of`). He also lists some special relations (e. g. `next-to` or `inbetween`) that do not fit in any other category. In addition, he defines 'combined' relations, i. e. combinations of angular and distal relations such as `behind-left`.[5]

Kray and Blocher [Kray and Blocher, 1999] proposed a classification based on the same two essential parameters – angle and distance – and categorized them according to the minimum dimension that the evaluation of a relation requires. They suggested a distinction between *two-point relations* and *n-point relations* (also known as 'path relation'). Examples for the first group include relations such as `behind` or `next-to`; `follow` and `depart` are examples for the latter group, which could be realized using "along" or "moving away from it".

[5] In some languages, the corresponding lexical realizations such as the German "links hinter" are used frequently, while in other languages such as English, they usually require a longer and more complicated description, e. g. a subordinate clause.

2.2 Situational concepts

Throughout the different research communities and disciplines, there are various definitions of what exactly is contained in the context model [McCarthy and Buvač, 1998], the user model [Dey and Abowd, 1999], and the situation model [Jameson, 2001]. Therefore, it is necessary to define how those terms will be used in this thesis. As we see it, a *situation* consists of two parts (if we assume a single user of interest): On the one hand, there are user-related factors, which are intrinsically tied to a specific user, her abilities, goals, personal traits, etc. These factors are captured in a *user model* [Kobsa and Wahlster, 1989]. On the other hand, the user perceives and acts in a certain environment, which also has distinctive properties, and which offers specific possibilities for action. In contrast to user-related factors, these factors are independent of an individual user (i. e., they equally affect all users) and they are determined by the environment. They define the *context model*. Together, these two models form the *situation model*.[6] The term *situational factors* will be used as a superordinate concept for user-related and contextual factors.

2.2.1 User-related factors

It is a well known fact that the abilities and properties of individual people strongly influence their performance and preferences in many tasks, even though most fundamental cognitive processes are assumed to follow the same underlying principles for all unimpaired people. This corresponds to the everyday experience that there are significant interpersonal differences when it comes to solving real-world problems such as navigating in an unknown environment. Therefore, it certainly makes sense to identify factors that have an impact on spatial cognition and that might enable a system to provide better services in that domain, especially in a mobile setting [Specht and Oppermann, 1999].

The *age* of a user is such a factor. There is strong evidence that the performance of adults in different cognitive tasks decreases with increasing age [Kirasic, 2000, Kray et al., 2002]. It does not only affect recall but also spatial cognition per se [Jenkins et al., 2000]. Consequently, a truly adaptive system should adjust its behavior to the age of its user, e. g., by preferring relational expressions that are easier to remember in case of older users, or by increasing the frequency of incremental navigational instructions for children.

Especially if a (spatial) task involves physical interaction with the environment or motion therein, the *physical constitution* of a user becomes an important factor. Not only may it impose restrictions on which terrains are accessible, respectively passable by her (e. g., wheelchair users cannot easily overcome staircases), it can also affect the speed she can attain, and her field of vision, i. e. which objects are visible to her. Furthermore, it has an impact on how often navigational instructions should be given.

A third influential factor in terms of spatial tasks is the user's *familiarity with the environment*. If she does not know anything about the environment, she probably expects more detailed and fine-grained route instructions than in the opposite case. Furthermore, candidates for possible anchor objects in relational expressions may be rated differently depending on whether the user is unfamiliar with the environment: visual salience might be of higher relevance in case of little

[6] If we want to consider multiple users, the situation model would consist of several user models and the context model.

knowledge while conceptual salience (e. g. whether the object is a historic sight) might be more important in the opposite case.

The *intention* or the current goal of the user may also influence the outcome of spatial reasoning processes. We can discover great differences when we compare, for example, the needs of a user who wants to reach the train station as fast as possible to catch her train, to those of a user, who wants to go on a sightseeing tour. While in the first case the system should frequently give short instructions that are easy to understand, it can be desirable to have more elaborate instructions that also highlight potentially interesting sights along the way. Certainly, very frequent instructions would not be welcome in the latter case.

Knowing about the *interests* of the user, e. g. whether she is interested in certain architectural styles or historic periods, enables a system to select objects (among a number of candidates) that are interesting to that specific user. These objects are probably better anchor objects in relational expressions and better landmarks for orientation than others that do not match the user's interests (assuming all other factors being equal).

The *cognitive resources* that the user currently has available to understand and perceive the output of a system also play an important role in the context of navigational assistance. For example, when a user is highly stressed – e. g. driving on her own in a foreign city at rush hour to get to certain location – she expects short instructions that are easily understood, which is less important when she is concentrating on the interaction in order to plan, which museum to visit next.

Furthermore, there are processes such as the determination of a suitable anchor object, where the *dialog history* makes a difference. If, for example, an object has already been mentioned in the dialog between system and user, then this object (or its name) are probably known by the user. This certainly contributes to its quality as an anchor object because it is more likely that the user is able to identify it (or its location) than in the case of unknown objects (again assuming all other factors being equal).

Additionally, the *emotional state* state of the user may severely impact her abilities and behavior [Picard, 1997]. For example, strong emotions such as horror or rage can reduce the ability to decode the output of an assistive system. Closely related to this factor is the *biological state* of a user, e. g. whether she is ill, intoxicated, or exhausted. Obviously, this may not only influence the user's interaction but also restrict the set of available actions. This list of user-related factors influencing spatial reasoning processes is not complete. There are most certainly additional further factors, even in the restricted domain of navigational assistance. However, not only user-related factors do have an effect, but also factors that originate mainly in the current environment. These are reviewed in the following section.

2.2.2 Contextual factors

In addition to the user-related factors described in the previous section, contextual factors have an impact on various spatial reasoning processes. The latter ones are not intrinsically tied to the user but are determined by the environment. One important example for these factors is the *means of transportation* that is currently used. Not only is route planning heavily influenced by this factor, but also the segmentation of a route, which is necessary to give incremental navigational instructions. The higher speed attained when driving a car, for example, may imply longer segments

than in the case of pedestrian. The current *weather conditions* – i. e., whether or not there is precipitation – does affect route planning as well, but may also impact the selection of relational expressions, as a pedestrian will probably prefer concise descriptions over more fancy ones if it is raining hard. A further factor of importance is the *granularity* of the current conversation, i. e. the scale of the space (e. g., large-scale vs. small-scale [Montello, 1993]) that the conversation is currently focusing on. This may impact the selection of landmarks and reference objects as well as which relations are feasible (e. g., geographic relations vs. those derived from body-axes).

An addition to these factors it makes sense to take into account properties related to the system that is providing navigational assistance. For example, the *output quality of sensors* can influence the selection of relations: if the orientation or exact position of the user cannot be determined precisely, certain relations and frames of reference (e. g. egocentric perspectives) are not feasible. The *amount of information* that is stored within in the system is another relevant factor when evaluating an object. Additionally, the *computational resources* that are available to a system have a great impact on all reasoning processes, as they impose hard constraints on the size of problems that can be handled in a timely fashion [Blocher, 1999].

Further factors include the *olfactory and acoustic state* of the environment, e. g. whether a user is exposed to an unpleasant smell or whether there is a lot of noise. The list of contextual factors presented in this section is as much incomplete as the one presented for user-related factors in the previous section. However, even these few factors can help a system to adapt to the situation when providing navigational assistance. While the identification of situational factors is a step towards more adaptive systems, we also have to consider whether it is actually possible to obtain information about them in a real world scenario. Therefore, the following section discusses the measurability of the factors identified so far.

2.2.3 Measurability

While situational factors play an important role in the designing user-friendly and adaptive systems, there are several problems associated with their inclusion into computer systems. Obviously, they have to be captured in a representation format that can be processed by a computer. This may entail a thorough analysis not only of what actually defines the factor and what facets are relevant in the application context but also what values to represent in the computer. For example, when modeling the age of a person, one has to decide whether this relates to her actual physical age, to her perceived age, or to another concept of age such as a composition of several biological measurements. Furthermore, we have to decide whether to present age numerically (e. g. the number of years since a person was born) or by symbolical values (such as “old” or “very young”).

An additional problem lies in the measurement of situational factors. For many factors such as the aforementioned age there is no single sensor that can reliably measure it. In many cases, we have to combine a number of sensors and employ sophisticated reasoning in order to determine a factor. This is especially true for user-related factors such as interests, abilities, and properties. Most of the time, these either either have to be inferred from observations that the system collects over an extended period of time, or we have to directly ask the user. Both of these approaches have some drawbacks: On the one hand, a long observation period has to pass before reliable information about the corresponding factor is available. On the other hand, the user may not be willing to answer a questionnaire prior to using the system. Additionally, even if there is a single

sensor that measures a situational factor such as the current noise level of the environment, it may return false readings or fail altogether. If a system is unable to cope with this type of problem, it is only of limited use in real-world applications.

Another reason that may restrict the measurability of situational factors is of less technical nature. Some factors may affect the security and/or privacy of a user and therefore cannot be obtained even though there may be sensors that can reliable measure it. Consider, for example, biological sensors such as heart-rate sensors or sensors that measure the tension of the skin. While these can provide information about the emotional state of the user, there are certainly many people that would object the measurement of this type of information.

One important consequence of these considerations is the need to cope with missing information (see also section 5.1). As we have seen, there are many reasons why certain information is not available either temporally or permanently. The following section will examine a further factor – the user's current position – that is highly relevant but also prone to this kind of problem. Therefore, a model (and its implementation) designed for adaptation to the situational context has to provide means to handle missing information.

2.3 Positional concepts

The current position of a user is another key factor defining her current situation. Many of the situational factors (such as available means of transportation) depend on it, but it also influences factors such as what bandwidth is available. Because of its high importance especially in a mobile setting, we present various means to cope with positional information in this thesis. However, we first have to define what concepts we subsume under the term 'positional information' (2.3.1) as well as what techniques currently are available to measure positional information (2.3.2).

2.3.1 Terms and definitions

Our notion of positional information does not only include the absolute or relative *location* of the user, but also his *viewing direction* and *body orientation*. Additional relevant positional parameters are the *speed* and *acceleration* of the user's movements. We assume that for each of these parameters a *sensor* exists that produces a permanent stream of data. One important observation is that currently, there is no technology that works perfectly in all cases and situations. Applications with a more general purpose, e. g. an electronic tourist guide, therefore have to rely on multiple sensors to determine the positional information. It is also advisable to include an explicit error measure for each type of positional information in order to be able to account for it during computation. In the following section, we will hence present means to address this as well as various techniques to measure positional information.

2.3.2 Positioning techniques

To detect the current location several different technical approaches exist. The most common system is the satellite-based *global positioning system* (GPS) that uses the runtime difference of satellite signals. The accuracy of GPS depends on the number of satellites, whose signals are received simultaneously. Since the system needs to "see" the satellites, GPS does rarely function

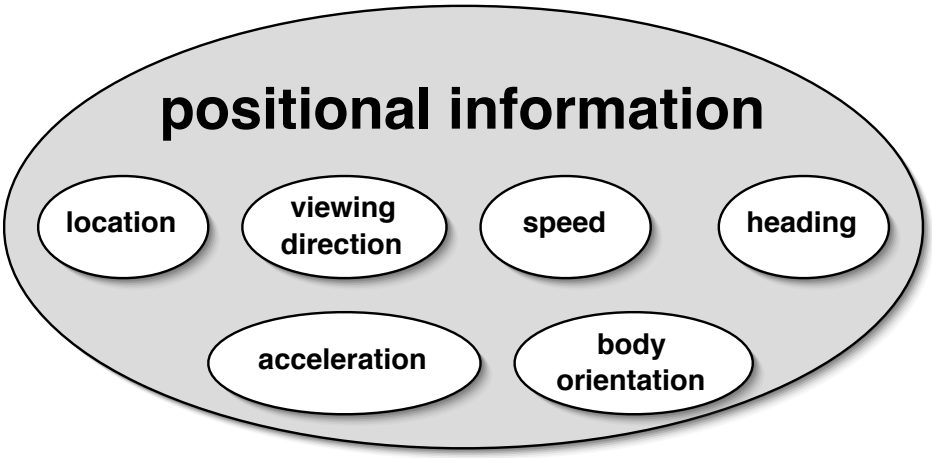


Figure 2.4: Facets of positional information.

inside buildings, and can be problematic during bad weather or in dense vegetation. A similar approach uses the network cells of cellular phone companies, and does usually also reach indoor areas. The problem here is that precision depends on the network cell size, which may vary from 500 meters to several kilometers.[7] Radio technology like *Wavelan* and more recently *Bluetooth* have much smaller cell sizes resulting in a higher accuracy, but do of course need a fully developed infrastructure of senders to work properly.

Other sensors for tracking the (indoor) position of the user are based on *infrared transmitters* ([Harter and Hopper, 1994, Want et al., 1995, Butz et al., 2001]) or *devices emitting and scanning laser beams* ([Lankenau and Röfer, 2002]). Since light does not pass through walls, it is possible to distinguish different rooms or parts of a room with relatively high accuracy. Infrared cell sizes may vary from a few decimeters to several meters. A means to increase accuracy consists of combining different techniques such as ultrasound with radio signals. This approach was applied, for example, by the Cricket system [Priyantha et al., 2000]). Unfortunately, both radio and ultrasound suffer from multi-path problems,[8] which require complicated algorithms to correct the resulting errors.

Imaging techniques can be used to detect landmarks in the surroundings that help to determine the actual location from video images (e. g. based on Marr's procedural model for visual perception [Marr, 1982]). The drawback of this approach is its high cost in terms of computational power and memory footprint. Other direct tracking approaches rely on electromagnetic scanning of tags or transponders such as the ones used in most theft prevention systems for stores. This allows, for example, to detect a person entering or leaving a certain room or area.

Electromagnetic devices such as electronic compasses are often used to determine the viewing direction and body orientation. A key problem with these devices is their sensitivity to metallic

[7] The third generation cell phone networks (the Universal Mobile Telecommunications System), which are currently in the process of being installed, promise to allow for a much more precise localization [The UMTS Forum, 2003].

[8] (partial) reflections of certain signals, which depend on a variety of factors such as the topology of the environment and the physical properties of nearby objects

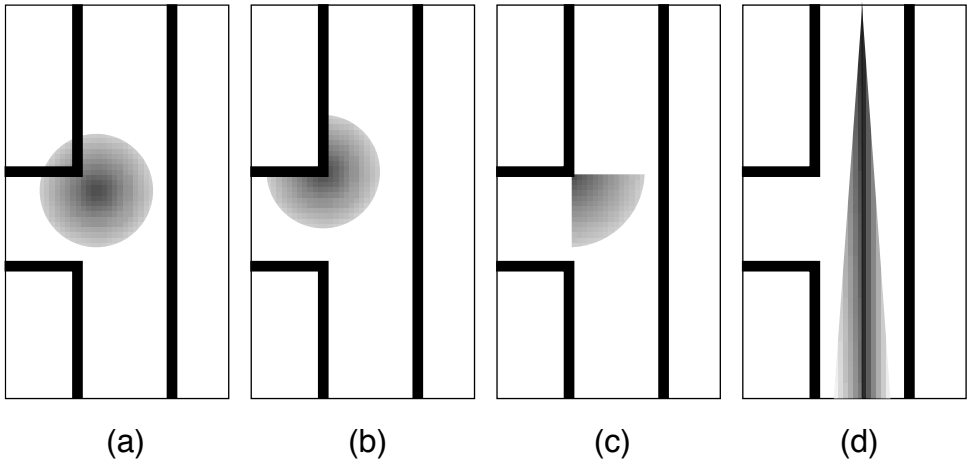


Figure 2.5: Error regions for different sensors: (a) GPS, (b) short range wireless cell, (c) infrared beacon, and (d) electronic compass.

objects in their vicinity as well as the strong interference and distortion that electromagnetic fields can cause. Some approaches therefore use infrared to determine both location and orientation (e. g. [Butz et al., 2001]) in different granularities. The viewing direction can also be tracked indirectly using *accelerometers* by measuring the changes in acceleration of the user. This technique is often applied in conjunction with other tracking devices to improve the overall quality of the results. Accelerometers also facilitate the determination of the actual location by *dead reckoning*, a method that extrapolates the location from the velocity and traveling direction (see also 5.3.1).

A further electromagnetic positioning technology relies on radio signals and small tags – so-called RF or RFID tags, where 'RF' is an abbreviation for 'radio frequency' [Estrin et al., 2001]: These tags emit a (static) signal upon entering the sending area of specialized antennas, which then allows for the localization of the tagged entity or person. The same principle is used, for example, by anti-theft systems in shopping malls.

In order to handle incomplete and missing information from the sensors we assume that every sensor delivers the positional information (e. g., x-,y-, and z-coordinates of the location), and an *error measure*. The error measure is a region in space that constraints the actual user location or viewing direction. The smaller the region, the more precise is the measuring. The region itself can be mathematically described as a circle, ellipse, cone or as a polygon,[9] depending on the characteristics of the sensor in use. Figure 2.5 shows some example regions for GPS (a), short range wireless cell (b), an infrared beacon (c), and an electronic compass (d).

[9]For the sake of simplicity we use 2D-representations but the approach can be extended to 3D.

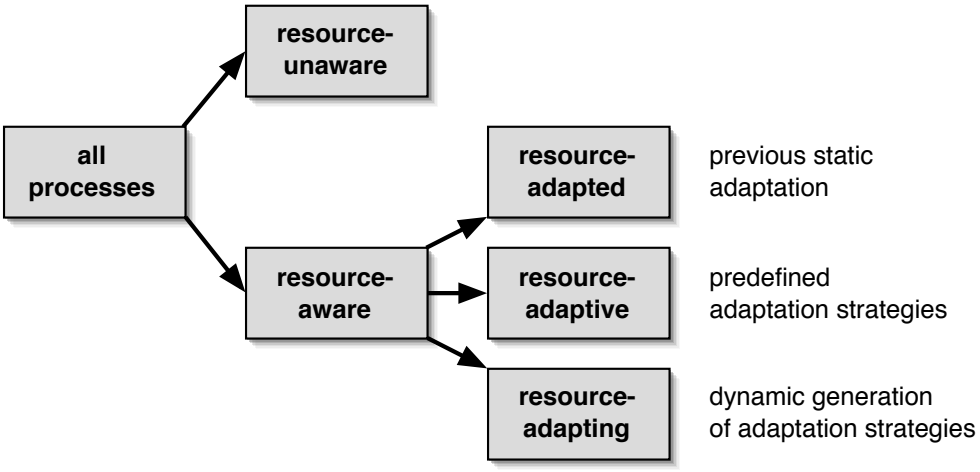


Figure 2.6: Taxonomy for resource-awareness (from [Wahlster and Tack, 1997]).

2.4 Resources

The term ‘resource’ is a very general and abstract concept that is usually defined as ‘a source of supply or support’ or ‘an available means’ [Merriam Webster, 2002]. In the context of human-computer interaction, we can define it more precisely as the ‘available means to solve a task’ (cf. [Jameson and Buchholz, 1998]), and we can distinguish two main types of relevant resources: *cognitive resources* and *technical resources*, which we will examine in the following two sections. *Resource-aware systems* are artificial (and natural) systems that are aware of what resources are available to them at any given time. Evidently, such systems should dispose of mechanisms to react to changes in the current resource situation: possible strategies in this context range from simply refusing to process the task at hand (e. g. if a key resource is not available) to more gradual and sophisticated approaches such as lowering the quality of the provided service.

Wahlster and Tack [Wahlster and Tack, 1997] distinguish three types of resource-aware processes: *resource-adapted*, *resource-adaptive* and *resource-adapting processes* (see figure 2.6). Resource-adapted processes have been adapted to resource restrictions that are not only previously known but also static. Consequently, their behavior is deterministic so that the quality of results is directly determined by the input. However, this kind of resource adaptation does not cope well with new and/or varying resource restrictions. Resource-adaptive processes employ a predefined adaptation strategy to perform well when faced with such a situation of varying resource availability. This results in a less deterministic output quality, which is mainly determined by what resources are available during computation. The main drawback of this approach consists in its inflexibility on the strategic level. Resource-adapting processes address this issue by dynamically generating adaptation strategies or by switching between several ones. They may also allow for adaptation and/or learning on the meta-level by analyzing past resource restrictions and evaluating the success of various adaptation strategies (see, for example, [Blocher, 1999]).

2.4.1 Cognitive resources

Cognitive resources are all types of resources that influence the cognitive processes a human performs. Since there is an ongoing discussion about how human cognition works and how it is structured, there is no single model of cognition. Consequently, it makes sense to focus on *resource restrictions* that have been documented as affecting cognition. Since cognitive resources are inherently tied to a user, it is hard to distinguish between what we called 'user-related factors' (see 2.2.1) and 'cognitive resources'. In this thesis, we assume that cognitive resources are a subclass of user-related factors that impact cognition for any task the user is performing (unlike, for example, familiarity with the environment, which only influences some tasks) and that are not intrinsically tied to the user's body functions (such as age or physical condition). However, this distinction is not a crisp one: Consider, for example, the emotional state of a person. While emotions are not necessarily required for all tasks a human is performing, it has been argued that they are nevertheless involved in most if not all cognitive processes [Picard, 1997].

Nevertheless, we can list several cognitive resources that influence a person's ability to reason, judge, and perceive – and thus her cognition in general. A first relevant aspect concerns the memory of a person. The amount of space available in the *working memory* [Baddeley, 1986] restricts the number of items that can be stored while performing the current task. Evidently, this has a major impact on all kinds of reasoning. A user of a mobile navigational assistant, for example, who wants to remember a complicated route description will certainly perform much worse if she has to remember a long unknown phone number as well or is listening to a public announcement simultaneously [Jameson, 2002].

Similarly, the *attention* of a person is a very relevant and limited resource that has a strong impact on cognition. Humans can split their attention only between very few different tasks, and the addition of a *secondary task* is a common means in psychological studies to increase the difficulty of another task. Attention or its sharing between several tasks is also a key issue when building assistive systems as these should help their user in a way that does not interfere with her primary task.

A further resource that impacts all cognitive processes is the *amount of time* which is available in order to perform the necessary steps. As an example, let us consider the navigational assistant scenario again: When the user is riding a fast car, she obviously has much less time to decide which direction to turn at the next crossing than she would have as a pedestrian. Time restrictions certainly also influence the perception of the output of such a system: a complicated presentation requires a certain amount of time to *decode*, and if the user cannot invest that minimum time, she is unable to understand it entirely. In addition to the cognitive resources of a human user, we also have to take into account technical resources in the context of human-computer interaction.

2.4.2 Technical resources

In analogy to a human being, an artificial system disposes of certain resources that impact its performance in various tasks. This concept does not only include factors that are directly related to the software and hardware constituting the system but also some factors that are influenced by the current context. Therefore, the context model can subsume technical resources. However, analogous considerations apply as in the case of cognitive resources and user-related factors.

One important technical resource is the *computational power* that is available to solve a given task. In many cases, this resource directly determines if a task can be performed at all, and how long the process will take. The same applies to the available *memory* as well as to the *bandwidth* that can be used, for example, in the communication between different components.

Furthermore, the means by which a system can interact with the user and perceive its environment play an important role. This does, for example, not only include the size, resolution, and color-ability of a screen but also the means for generating audio output. Similarly, on the input side, technical resources include keyboards, pointing devices, and microphones. Additionally, the means by which a system can perceive its environment fall into the category of technical resources. These include sensors such as microphones, cameras, or positioning devices (e. g. GPS, compass, infrared sensor).

Technical as well as cognitive resources are defined relative to the task that a human user (or artificial agent) is performing at a specific moment in time. Therefore, the next section reviews different tasks related to space.

2.5 Tasks

The Merriam Webster online dictionary [Merriam Webster, 2002] gives several definitions for the term ‘task’, one of which defines it as “a usually assigned piece of work often to be finished within a certain time”. The handling of tasks and their fulfillment and persecution is a central component of human cognition. While some tasks can be performed simultaneously without interference, others are competing for the same limited resources that are available at any given moment [Jameson, 2002]. Due to this fact, it is important to closely analyze what tasks a human typically performs in a domain before proposing a model that should cover that domain [Casner, 1991]. Therefore, we will review various tasks related to space in this section (2.5.1) and identify some common subtasks involved in these (2.5.2). In doing so, we will concentrate on those tasks that can benefit from assistance by a computer

2.5.1 Tasks related to space

Since we live and act in a three-dimensional world, many tasks that humans perform are related to space. Examples for such tasks include locomotion, the physical interaction with real world objects, and communicating with others about these tasks. Not all of these activities are easily simulated by an artificial systems, and not all of these benefit from assistance by such a system. For example, only in very special cases (such as the compensation of certain severe motion disabilities) does it make sense to obtain assistance from a computer systems for grabbing objects. Consequently, we will focus on those tasks that can be supported by an artificial system.

Objects of the real world have some spatial properties that form the target of spatial tasks and actions. One of these properties is the *location* of an object, the place it occupies in the real world. A common task related to space, therefore, consists of determining or describing the location of an object. This *localization* can either be performed on an external object, or reflexively (on the human observer herself). The latter case is a special case of localization, which we refer to as *self-localization.*

Localizations are also a means to *refer* to objects, e. g. by describing a building as being "close to" another. This points us to another task related to space, the *identification* of objects in the real world. In addition to relational expressions such as the one presented above, it is also possible to employ other means. Anaphoric expressions – for example "What's *this*?" – and deictic gestures (e. g. pointing to an object) are further ways.

Another very common task related to space consist of *finding and following a way* to a target, which is often 'the reason' behind a persons desire to learn the location of an object. Way finding is a complex tasks that involves several steps that need to be taken in order to succeed [Maaß, 1999]. Firstly, the location of the origin and the target have to be known. Secondly, a route leading from origin to target has to be planned, and thirdly, this route has to be followed. Obviously, this is a task that can greatly benefit from assistance (we present several such systems in the following chapter), but its decomposability also hints at the existence of certain sub-tasks, which we will identify in the following section.

This short list of tasks related to space is certainly not complete, but those tasks we presented have several things in common. They are fundamental to many other tasks, and thus have to be performed quite frequently. In addition, they are not restricted to the real world but can also be applied to representations thereof (such as maps) and (abstract) entities in figural space (see 2.1.1). For example, we can localize an object on a map and interact with it using spatial metaphors ("Pan left."), or we can refer to controls in graphical user interface using spatial expressions ("What is the function of the icon in the upper left corner of the screen?"). Furthermore, the task reviewed above do also rely on common sub-tasks, which we will identify in the following.

2.5.2 Common sub-tasks

When we look more closely at localization, identification, and way finding, we can further dissect these tasks into sub-tasks – in a similar way in which we analyzed way finding in the last subsection. For example, all these tasks involve the *evaluation of objects* – we have to 'rate' several objects according to certain spatial (and non-spatial) criteria. In the identification task, the selection of the intended object from a set of potential candidates is the key process to perform. The same is true for the localization task. Here, not only the intended object (target object) has to be identified, but it is often also necessary to select a suitable anchor object for a relational expression. Again, this requires the evaluation of real world objects according to a number of criteria – as does the identification of origin and target of a route in the case of way descriptions.

Since way descriptions often include relational expressions such as "left of" or "in front of", we can identify another sub-task shared with the localization sub-task. As we have seen in 2.1.3, such angular or projective relations require the establishment of a *frame of reference.* The selection of a suitable frame of reference also consists of a selection problem, where we have to choose the one that suits best the current situation from a (potentially large) number of candidates. These candidates include, for example, the user's current position, the position of relevant objects, and previously used frames of reference. In 4.1.2, we will present a thorough analysis of various frames of reference.

The establishment of spatial relations is a further common sub-task, which is part of localizations and way descriptions. Here, we can distinguish between different types of spatial relations (see 2.1.3 for details) so that we can define separate tasks for the evaluation of two-point and n-

point relations. It is important to note that this sub-task does strongly interact with the other ones presented above. For example, the currently selected frame of reference can be influenced by the object evaluation – i. e. the currently favored anchor object – and it has a great impact on which spatial relations are applicable in the current situation. However, the selection of a spatial relation may also influence the choice of a frame of reference or anchor object.

A final sub-task, which we can identify in the context of way description and finding, is the *segmentation* of longer trajectories into smaller ones. For example, if we want to generate a verbal path description of a longer route, we have to produce a sequence of instructions for successive chunks of the route. This requires the selection of appropriate segments, i. e. the decision of where to 'cut' the trajectory into smaller parts.

All the sub-tasks described above reoccur in the context of different (complex) tasks related to space, and they are strongly interrelated and influenced by various situational factors. In the following section, we therefore discuss these interactions and dependencies in more detail.

2.6 Discussion

In this chapter we reviewed several aspects and factors related to situated interaction on spatial topics: These included spatial concepts (such as scale, relations, and frames of reference), situational concepts (consisting of user- and context-related factors), the user's position as well as cognitive and technical resources. Since our goal is to design (and implement) a computational model that allows for the situated interaction on spatial topics, we have to integrate all these factors. In this section, we want to give some examples for the impact of situational factors and point out several problems arising in the context of situated interaction in real-world use. Figure 2.7 gives an overview of the relationships and interactions, which we will discuss in the following paragraphs.

The influence of user-related factors on almost any task a human performs is rather obvious. Consider, for example, an old user who's physical condition is not too good. If she wants to be guided from one place to another, it stands to expect that she will walk at a slower speed than the average user. Consequently, the segmentation of the route and the resulting descriptions should be different from the one computed for the average user as it takes her longer to follow a segment and thus requires her to remember instructions for a longer period of time. This example can also serve to illustrate the impact of contextual factors such as the current means of transportation: whether someone is walking or driving a car makes also a great difference in terms of segmentation and way descriptions.

The influence of the user's current position (and the quality of positional information) is also easy to demonstrate: It is often a prime candidate for being used as a frame of reference (egocentric frame of reference, see 4.1.2). Not only does such a use facilitate the understanding of relational expressions – the user does not have to perform any mental rotations or translations in order to 'imagine' the current frame of reference – but it also does not require any knowledge of the environment. Furthermore, the user's current position plays an important role in giving route instructions: it has to be closely monitored to give directions at the right location. Otherwise, a mismatch between the direction and the location where it is given may result in false and misleading instructions.

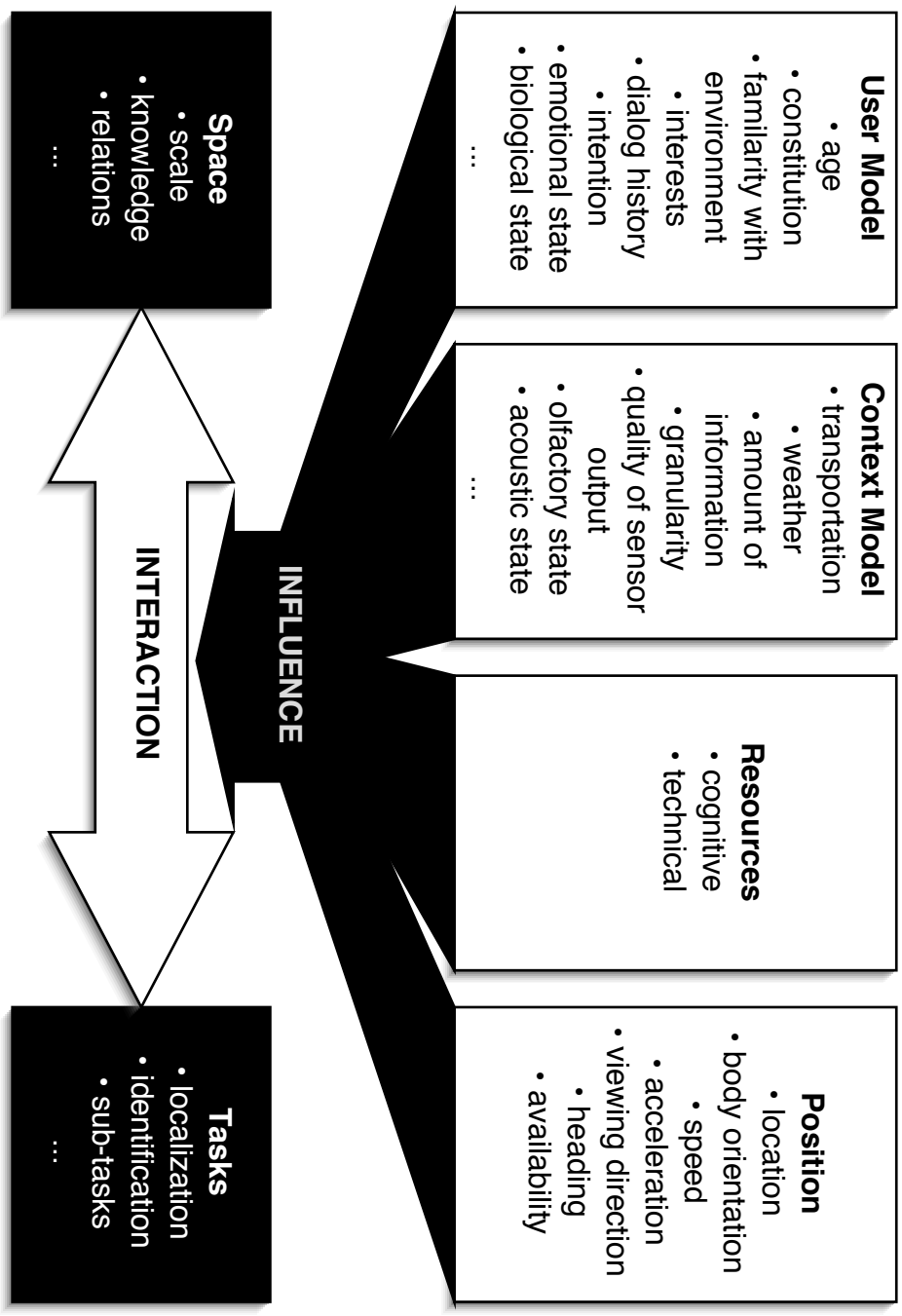


Figure 2.7: Influences and interactions in situated interaction on spatial topics.

The user's current position is also a good point to illustrate typical problems that arise in real-world use: Although positional information is only a facet of what constitutes the current situation [Schmidt et al., 1999], it nevertheless plays a key role since the current location and viewing direction of a user determine, for example, to a large degree what she can see and do. As we pointed out in 2.3.2, none of the available techniques is always reliably returns the precise position. Consequently, we have to deal with imprecise, missing, or even false readings for positional information.

A similar problem consists of the resources available to perform a certain task. Anybody who has ever interacted with a person while she was performing another demanding task, will confirm that the cognitive resources that person can put into the interaction were severely limited. Obviously, this has consequences for interaction on spatial topics: depending on the amount of cognitive resources that a human user can afford to put into the interaction with a system, the complexity and type of the content has to vary. For example, a distracted user may be unable to grasp elaborate way descriptions which precisely describe the way to follow, but she may very well understand simple turn instructions.

Equally important are the technical resources that a system can use for the computation and presentation in the context of situated interaction on spatial topics. For example, if only audio output is available, graphical means are simply unfeasible. Similarly, if the computation required for an interaction takes too long, it is possible that it becomes obsolete altogether such as a turning instruction given after the addressee has already passed the corresponding decision point.

The problems and examples we described above highlight the impact that situational factors such as user properties, positional information and resource availability may have on situated interaction on spatial topics. Consequently, a computational model has to take into account the impact of these factors as does a system that aims at providing the user with assistance in this area. Furthermore, the addition of further factors should be easy to realize. In the following chapter, we review a selection of related systems in terms of which factors they take into account, and we also compare them along several other dimensions.

2.7 Summary

In this chapter we presented the basic terms and concepts that we will use throughout this thesis. We first introduced fundamental concepts related to space such as the classification based on the scale of a space (figural, vista, environmental and geographical space). In this context, we also reviewed spatial knowledge, which can be classified into landmark knowledge, route knowledge, and survey knowledge. A further relevant field is that of spatial relations: We introduced basic terms, e. g. anchor and target object as well as frames of reference, and proposed a classification according to several criteria (qualitative vs. quantitative, two-point vs. n-point). We also shortly reviewed Gapp's 'scruffy' model based on potential fields [Gapp, 1997], and gave a brief overview over Randell et al.'s 'neat' model based on the connection relation [Randell et al., 1992].

In addition to spatial concepts, we analyzed user- and context-related factors that constitute the situation model. These factors include, for example, the age and constitution of the user as well as the current means of transportation. A further influential aspect in this context is the user's current position; we identified relevant positional information (e. g. location, speed, and heading), and reviewed means to measure these. We then presented a classification of different types of resources into cognitive and technical resources as well as a taxonomy for resource-awareness (which will also be used in the following chapter to compare related systems).

Furthermore, we collected typical tasks that arise in the context of situated interaction on spatial topics. These included the identification and localization of objects as well as finding and following a route. A closer analysis of these tasks revealed several common sub-tasks such as the evaluation of objects, the establishment of a frame of reference, or the computation of spatial relations. The chapter concluded on a discussion of the impact that situational factors have on interaction on spatial topics, and we also pointed out several common problems in this context.

3 Related work

In recent years, the interest in (mobile) navigational assistance has risen tremendously – partly because of the widespread adoption of mobile phones and PDAs. Hence, there has been a growing number of research projects on (mobile) systems that provide navigational assistance and further services related to space. These systems are highly relevant in the context of situated interaction on spatial topics. Not only does the situation of the user change frequently since she is moving around, but also do these systems offer a broad array of services related to space. Hence, in order to provide a background for the work presented in later chapters and to put it into perspective, we review a selection of prominent and/or closely related systems in this chapter.

We start by introducing the Cyberguide system (in 3.1), one of the first systems that provided mobile navigational assistance. We then review Hippie – a prototype that was developed in the HIPS project (in 3.2) – before we present the GUIDE system, which was among the first systems that were available publicly (3.3). CyberAssist is a more recent project that is carried out at the CyberAssist Research Center in Japan (3.4). We then analyze the TellMaris system, which is being developed at Nokia and being targeted at mobile phones (3.5). It shares the later property with LoL@, a tourist guide for the city of Vienna, which we present in 3.6. Unlike most other systems, REAL allows for in- and outdoor use and a seamless transition between these uses, which we review in 3.7. Finally, we introduce SmartKom, a large German research project focussing on multi-modal interaction and its mobile component (in 3.8).

After reviewing these systems, we analyze their advantages and shortcomings in section 3.9 in a number of categories such as inclusion of situational factors and adaptation capabilities. This is contrasted with the properties of our approach (see chapter 4), respectively its prototypical implementation (see chapter 6).

3.1 Cyberguide

The Cyberguide project [Long et al., 1996] was one of the first projects for mobile navigational assistance. Within this project, several prototypes for mobile assistance were developed at the Georgia Institute of Technology. The development of various Cyberguide instances was influenced by earlier work on the PARCTab [Want et al., 1995], the InfoPad [Long et al., 1995], and the Active Badge system [Want et al., 1992] as well as the Personal Shopping Assistant [Asthana et al., 1994]. Throughout the project, both indoor and outdoor systems were built that ran on a PDA (Newton MessagePad 100) as well as on a TabletPC. The underlying position technologies consisted of infrared beacons and GPS. Figure 3.1 shows two screenshots of a guidance system for open house days at the Graphics, Visualization and Usability Center at the Georgia Institute of Technology that provides visitors with information about available demos.

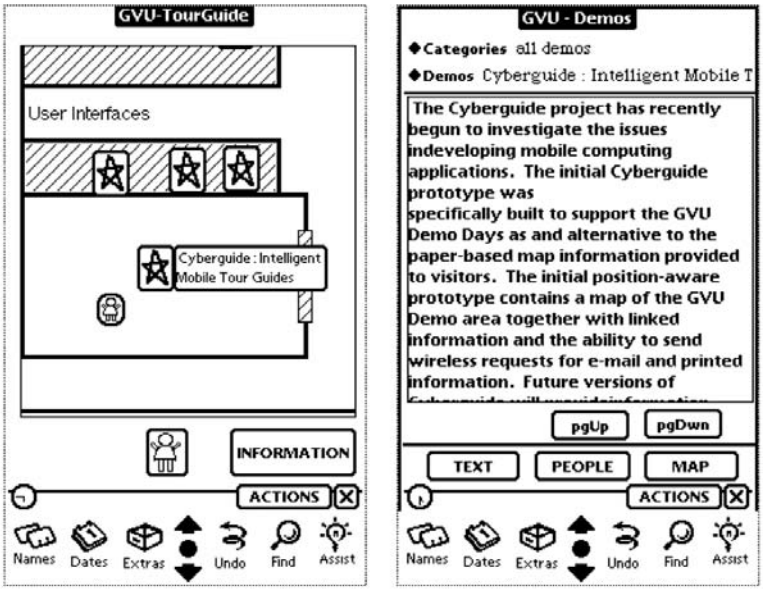


Figure 3.1: Cyberguide - a mobile context-aware tour guide (from [Long et al., 1996]).

One main goal of the Cyberguide project was to build a system that supports rapid prototyping [Long et al., 1996]. Consequently, several prototypes for various purposes were developed, which provided different services, i. e. (location-based) information, localization, and bar recommendation as well as communication [Abowd et al., 1997]. Contrary to the title of the project, there seems to be no real guidance functionality in any of those prototypes. The information service provides the user with an interactive map that displays the user's current location as well as icons for points of interest. She can then access further information by clicking on these icons, or on an 'information' button. In addition, the map is used for (self-) localization: icons on the map represent the user's current location and points of interest in her vicinity. A bar recommendation service also relies on this interface. In addition to these map-based services, an e-mail service is included that allows the user of the system to send messages to other users.

One of the central components of Cyberguide is the so-called 'Navigator', which is in charge of providing accurate information about the current position of the user. Conceptually, it seems to be independent of the underlying sensing technology – within the actual prototypes infrared beacons and GPS were used. Situational factors beyond the user's current position are not considered during the provision of services [Long et al., 1996] but are envisioned for later development. Similarly, cognitive and technical resource restrictions are almost entirely neglected. Since there exist separate versions of the system for use on a PDA and on a desktop PC, Cyberguide can be classified as a *resource-unaware* system but an integration of these versions would result in a basic resource-adapted system. Neither lack of information nor varying quality of positional information is accounted for. However, most services do not require very precise information about the user's current position so that adaptation to changing information quality is not a central issue in Cyberguide.

The interface of the system mainly consists of maps and textual information which are enriched by a few control buttons (see figure 3.1). The user interacts with the system through point and click with either a pen (PDA) or a mouse (PC), and possibly textual input (e. g. within the email component). Hence, the interface is mainly unimodal. In the literature about Cyberguide, there is no mentioning of an internal representation of what is presented to the user in which way. Consequently, the same information is apparently always presented in the same way, i. e. the same medium or modus. The same is true for verbal interaction: the main language is English, and there is no reference to multi-lingual capabilities of the system.

Since Cyberguide was conceived from the beginning to support rapid prototyping and the development of various context-aware applications, it was based on a modular architecture. The basic services were provided by four main components that interacted through proprietary interfaces. Hence, not only several interchangeable versions of each component were created but this approach also (theoretically) allowed for the flexible distribution of the components between a mobile client and server. The later property was not exploited as this was out of the scope of the project.

3.2 Hippie/HIPS

Hippie [Oppermann et al., 1999] is an adaptive exhibition guide that was developed at the German National Research Center for Information Technology (GMD-FIT[1]) within the HIPS (short for **H**yper-**I**nteraction within **P**hysical **S**pace) project [Benelli et al., 1999]. The main goal of the project is to unobtrusively enrich the visit to a museum. Hence, Hippie is intended to run on a PDA, or on a subnotebook [Oppermann and Specht, 1999]. Being based on standard Internet technology, Hippie also supports access from a standard PC.

In addition to providing guidance to the location of exhibits, Hippie also adaptively provides information about the exhibits through hierarchical and dynamically created web pages (see figure 3.2. There is also some rudimentary localization service that is able to generate simple spatial references such as "In front of ...". But it is embedded in other functions such as proactive information provision and is not accessible on its own. Similarly to the CyberGuide system (see 3.1), Hippie also allows for interpersonal communication through message sending.

[1] which is now part of the Fraunhofer Research Group

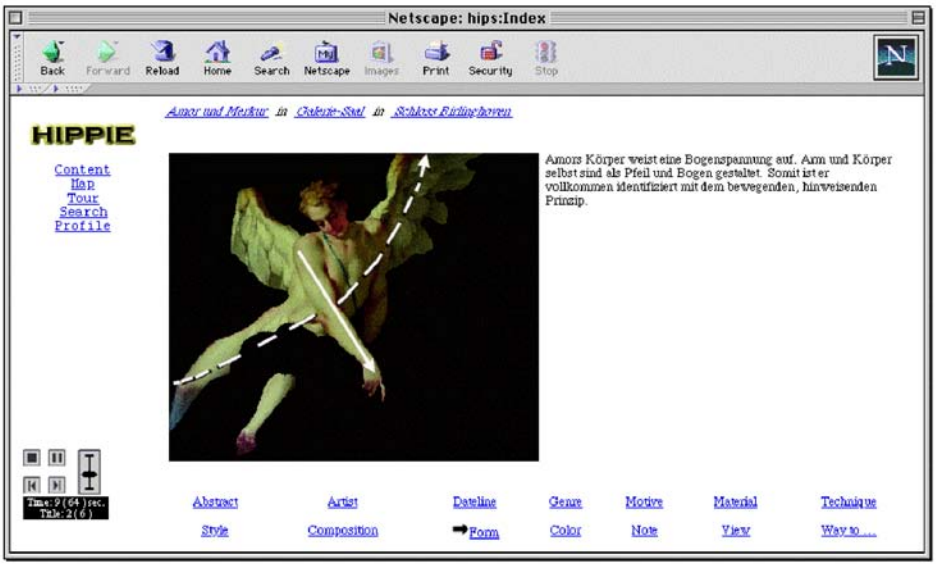


Figure 3.2: Hippie - a nomadic information system (from [Oppermann and Specht, 1999]).

Infrared beacons are the sole means by which Hippie determines the user's current location. These are not only attached at each exhibit but also at doors and passages to other rooms so that the system can keep track of the user's position on room-level granularity. An electronic compass provides information about the user's orientation. Other projects within HIPS rely on infrared for determining orientation as well [Not et al., 1998]. One key strength of Hippie lies in its sophisticated user-model and resulting adaptation capabilities in terms of information provision. It takes into account the user's interaction history (e. g. which links she clicked on and what information about an exhibit she accessed) as well as the position history. Based on these observations, Hippie adapts the content of its presentation and proactively suggests further sights that might potentially be of interest for the user. Contextual information beyond the current position are not taken into account, neither is the current task (since Hippie is designed to support only one task – the visit of an exhibition). Further prototypes developed within HIPS implement a multi-level proximity concept, where different presentations are generated depending on the distance to an exhibit and the time spent next to it [Not et al., 2000, Bianchi and Zancanaro, 1999] as well as depending on the type of visitor[2] [Levasseur and Veron, 1991].

According to the classification of resource-aware systems (cf. [Wahlster and Tack, 1997]), Hippie is a *resource-adapted* system since the presentations are adapted to either stationary or mobile use [Oppermann and Specht, 1999]. However, beyond this distinction, it does not take into account any cognitive or technical resources. In addition, there are no means to adapt to the lack of information. In terms of adaptation to varying quality of positional information, Hippie incorporates a static two-level approach: if the position is only known on room-level granularity (trough

[2]Levasseur and Veron [Levasseur and Veron, 1991] found that most visitors to a museum behave according to four distinct spatial patterns. They labeled these types depending on the underlying motion patterns after animals: the 'ant', the 'butterfly', the 'fish', and the 'grasshopper'.

the beacons mounted at doors and passages) information about all exhibits in the room is provided. If Hippie knows the position more precisely (i. e. on exhibit-level), it provides information on the specific exhibit the user is facing. However, the system cannot infer the user's position in absence of any positional information.

The verbal interface of Hippie employs English and there is no mentioning of a language-independent representation or multi-lingual abilities of the system. Presentations generated by Hippie are unimodal (pictorial and textual) in the stationary scenario, and multi-modal in the mobile scenario (spoken language, pictures and text). The user interacts with the system through a point-and-click metaphor. Apparently, the modus and medium of a presentation are not dynamically selected.

Hippie is based on an architecture that relies on the standard Internet approach of a web server from which a client retrieves (personalized) web pages. Additionally, there is an 'adaptive component' that is in charge of maintaining a user model [Oppermann and Specht, 1999]. From the literature, it is not clear how this part is connected to the rest of the system. Since the encoding of the presentations is standardized (HTML [World Wide Web Consortium, 2002d]), access is possible from any standard web browser.

3.3 GUIDE

The GUIDE project at Lancaster University is one of the best known systems for navigational assistance [Cheverst et al., 2000a, Cheverst et al., 2000c, Cheverst et al., 2000b]. It is one of the few systems that have been deployed in the real world with tourists visiting the city of Lancaster, UK. The GUIDE system is publicly available in Lancaster, where tourists can rent a unit at the tourist office. The system consists in a standard tablet PC which is equipped with a wireless network card. The city has been outfitted with some strategically placed base stations that provide wireless network access to those clients. Figure 3.3 shows some screenshots of the system.

The main services offered by GUIDE consist in the provision of information on sights at Lancaster and guiding the tourist to these sights, i. e. from one to another. However, there is also a means to communicate with other users and to make reservations (see buttons in lower right corner of screenshots figure 3.3). Unlike most other outdoor guidance systems, GUIDE does not rely on GPS but rather uses the network cells defined by several strategically placed wireless access points to determine the current position of the user. In addition, there is a simple interactive means to address the loss of network connection: the user is given a long list of thumbnails of all sights at the city and is then asked to select the one closest to her. Based on her answer, the system estimates the user's current position.

When a tourist starts to use GUIDE, she undergoes a short interview to build up a small static user model that is later used for adaptation in two ways. On the one hand, it guides the selection process of what information to present to the user. On the other hand, it has an impact on how the information is presented. Since the system also keeps an interaction history, it is able, for example, to change the order in which nearby sights are displayed on the screen so that the items at the top of the list are those sights that have not been visited yet. The main contextual information that GUIDE takes into account in addition to the user's current location are the opening hours of the sights at Lancaster. This does also affect the ordering of the sights displayed on the screen. No

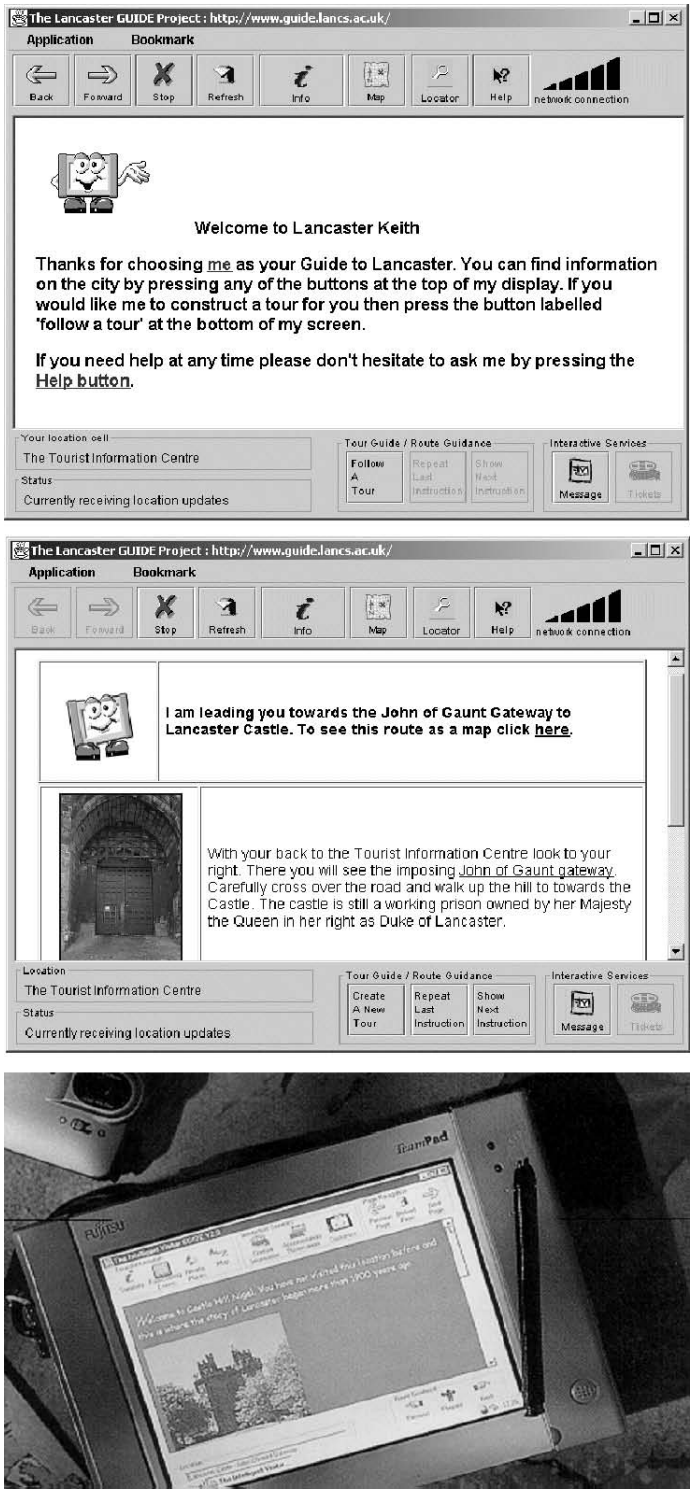


Figure 3.3: GUIDE - a mobile tourist guide for Lancaster, UK: Welcome screen (top), navigation (middle), mobile device (bottom) (from [Cheverst et al., 2000d]).

further contextual factors are mentioned in the literature on GUIDE, neither is any task-model or an influence of the user's current task on any aspect of the system.

Similarly, neither technical nor cognitive resources are currently being considered in the system.[3] In terms of addressing the problem of missing information, GUIDE relies on a proxy/cache that partially compensates the loss of network connectivity – which is equivalent to losing access to relevant information – by providing less sophisticated and less up-to-date information from the local database. We already mentioned above that GUIDE also provides a simple means to cope with missing positional information by displaying a list of thumbnail pictures of the sights at Lancaster and letting the user pick the one that is closest to her.

Since GUIDE is available publicly to tourists visiting the city, it has to offer its service in several languages. Therefore, the system disposes of (static) descriptions of the sights in several languages. From the literature, it is not clear whether the interface is translated as well. In addition, it does not mention dynamic generation capabilities of the system. GUIDE presents its services to the user using pictures and text on the tablet PC, hence can be classified as being a multi-medial and unimodal system. However, some field tests with audio-push were received enthusiastically, and hence the inclusion of further channels is planned for future versions [Cheverst et al., 2002].

The underlying architecture of the system is an enhanced client-server model consisting of a (web) server, which provides in-depth and up-to-date information on the sights of the city, and a client (in the form of the tablet PC). In order to allow for disconnected operation, the client allows for limited functionality (in terms of information quality and guidance) by relying on local cache/proxy. Technically, the system is realized as a web server with several proprietary additions, and a slightly modified web browser on the client side. The interaction of these components relies on an object model presented in [Cheverst et al., 1999]. The interface is based on a web browser, from which it also borrows the interaction metaphor (e. g. buttons for moving forward and back in interaction history). Consequently, the presentations consist of standard web pages that are enhanced by proprietary tags, which incorporate dynamic information.

3.4 CyberAssist

The CyberAssist project at the CyberAssist Research Center (CARC) [CARC, 2002] pursues the ambitious goal of designing devices and techniques to enable a human user to access a variety of location-based services through a simple and unified interface. Figure 3.4 shows a recent prototype (Cobit).In order to allow for permanent operation of the device, the battery-less device called Cobit[4] uses small solar panels in order to operate. Since information is transmitted using modulated light, the solar panels also serve the purpose of network connectivity [Nishimura et al., 2002]. The corresponding infrastructure consists of a network of small micro-servers that run a modified version of Linux (UBLinux) with custom software components (UBKit – Ubiquity Building Toolkit) that facilitates rapid development [Mori, 2003].

Several services related to space are envisioned for CyberAssist, ranging from train ticket booking (instead of using an ATM the user can access this service through the unified interface

[3] However, the system keeps track of the time during tours, and automatically adapts the tour if the user spends more time than planned at some sights, e. g. by removing future sights in order to meet the previously specified duration.

[4] which stands for 'compact battery-less interaction terminal'

Figure 3.4: CyberAssist - devices for unified access to location based services (adapted from [Nishimura et al., 2002]).

provided by a Cobit) over cooking aid (suggesting recipes based on previous shopping) to spatial reminders (which automatically remind the user of things she wanted to do at a location, e. g. to buy stamps when she passes by the post office) and disaster mitigation. Currently, a shopping assistant service is being realized, which provides information on the products on display on shelves that are located near the user. This information is transmitted using modulated light which also provides the energy to power the device. In addition, the current position is determined based on the light source (similar to infrared beacons). However, the general infrastructure of CyberAssist has specifically been designed to allow for the inclusion of various positioning techniques [Sashima and Kurumantani, 2002].

This approach has some distinct advantages over other platforms as it does not require any additional power source. Consequently, the actual devices worn be users can be very small (see figure 3.4), which improves their general acceptance and facilitates the design of intelligent clothing or fashion items. Furthermore, the use of modulated light allows for the seamless integration of the sender infrastructure into existing light sources, and thus avoids the installation of dedicated beacons or antennas.

In order to adapt to the current user, the system incorporates a user model agent that provides information about the user to other agents of the system. Beyond the user's current location no further contextual factors are taken into account. Neither is the task the user is currently performing or any resource-restrictions. Hence, we can classify CyberAssist as a *resource-unaware* system. Further issues which are not addressed in the system include the lack of information and missing or imprecise positional information. The literature on CyberAssist does not list any multi-lingual capabilities but one declared goal is multi-modal interaction, e. g. by means of natural language and device-based interaction. CyberAssist relies on a modular architecture which is based on a FIPA-compliant [The Foundation for Intelligent Physical Agents, 2002] multi-agent system with an XML-based [World Wide Web Consortium, 2002a] content encoding.

Figure 3.5: TellMaris - navigation in 3D space on a mobile phone (from [Kray et al., 2003]).

3.5 TellMaris

TellMaris is a prototype for a mobile tourist guide that was developed at Nokia Research Center [Laakso, 2002, Bosch i Creus, 2002]. It is one of the first mobile systems that combine three-dimensional graphics with two-dimensional maps and that run on a mobile device (a mobile phone). The first prototype was developped for the city of Tønsberg, Norway to help boat tourists in finding locations of interest (e. g. hotels). During summer 2002, a first exploratory field test was conducted with a small number of volunteers (see [Laakso, 2002, Kray et al., 2003]). Figure 3.5 shows a picture of the interface: On the left side of the screen, a three-dimensional rendering of the area around the user's current location is shown. The user can dynamically navigate the rendered scene using cursor keys on the phone. Her location is also highlighted in the map on the right side of the screen, and is dynamically updated to reflect the navigation of the user in the 3D scene.

The first prototype of TellMaris only provides static navigation services: the user can navigate simultaneously in the 3D scene and the map in order to explore the city of Tønsberg. Currently, the only automated assistance consists of an arrow in the 3D model that points in the direction of the target location that the user has selected previously. While positioning eventually will depend on GPS, the first prototype depends entirely on the user to manually position herself on the map or in the 3D model. Similarly, no situational factors are currently taken into account at any stage of the computation.

TellMaris has been fine-tuned for use on a mobile phone, and hence, can be classified as a *resource-adapted* system in terms of technical resources. Cognitive resources are not considered in the current prototype. While there is no means to address the lack of relevant information, the manual navigation along with the presentation of pseudo-realistic images *and* a map to enable the user to interactively align her position with the one in the system. Preserving this ability in the next prototype will enable the system to address GPS outages, i. e. missing positional information. This is one of the potential benefits of combining 3D renderings with conventional 2D maps, which was supported by a common strategy observed during the exploratory study: Several subjects used

the 3D rendering to navigate locally and to identify objects in their immediate environment while relying on the 2D map for global navigation [Kray et al., 2003].

Relying largely on graphical interaction, TellMaris uses only very little textual information. Currently, the system's interface elements are in English, which could be translated to any other language with little effort. However, since the system does not employ a language-independent representation it cannot dynamically generate textual or verbal output. The system's interface (see figure 3.5) is unimodal, and only supports textual input and button-based interaction.

The TellMaris system has been streamlined for use on a mobile phone. Hence, its underlying architecture consists of a small number of highly optimized components [Bosch i Creus, 2002], which interoperate to provide the services described above. The modular architecture is realized as a set of applications that run in parallel and that interact using a proprietary protocol and content language.

3.6 LoL@

The LoL@[5] system [Anegg et al., 2002, Pospischil et al., 2002] is a mobile tourist guide for the city of Vienna designed for the next generation of mobile phone networks, the Universal Mobile Telecommunications System (UMTS). It was developed at the Forschungszentrum Telekommunikation Wien and is currently undergoing empirical evaluation. Similarly to the GUIDE system (cf. 3.3), the main interaction metaphor is that of a web browser: user's click on links or buttons to access information and/or to trigger some actions. Figure 3.6 shows several screenshots from the system as well as the interactions required to switch between them. The target platform for the system is a PDA or a mobile phone of the third generation (3G) but due to some technical and availability problems, LoL@ currently runs on a laptop [Pospischil et al., 2002].

The system provides three main services to its user: guidance on virtual or real tours through Vienna, provision of information on sights, and a personalized semi-automated tour diary. The guidance process requires the user to manually confirm her arrival at the end of each segment in order to trigger instructions for the next one. These instructions are either transmitted via speech output or using a map that is enriched using a set of predefined sights. The user can also access multi-medial information on the objects that are visible on screen by selecting them with her pen and then clicking on an icon. The corresponding information adheres to the browser metaphor: hierarchical links allow for accessing structured information such as pictures, texts, or movie clips. When the user follows a tour, LoL@ automatically generates a tour diary consisting of a chronologically ordered list of visited sights and user additions such as pictures and notes. The diary is a web page, which includes links to the information contained within LoL@, and which can be downloaded to the user's PC after the tour.

Following the classification proposed in [Wahlster and Tack, 1997] (see 2.4), LoL@ belongs to the group of *resource adapted* systems: it has been designed a priori to account for some technical and cognitive resource restrictions in a static way. In order accommodate for the small screen size of mobile devices and a stressed user (e. g. by uncomfortable environments or time pressure), the presentation of information was streamlined using the browser metaphor. Obviously, there is no means to address the lack of situational information as this is not considered in computation.

[5] local location assistant

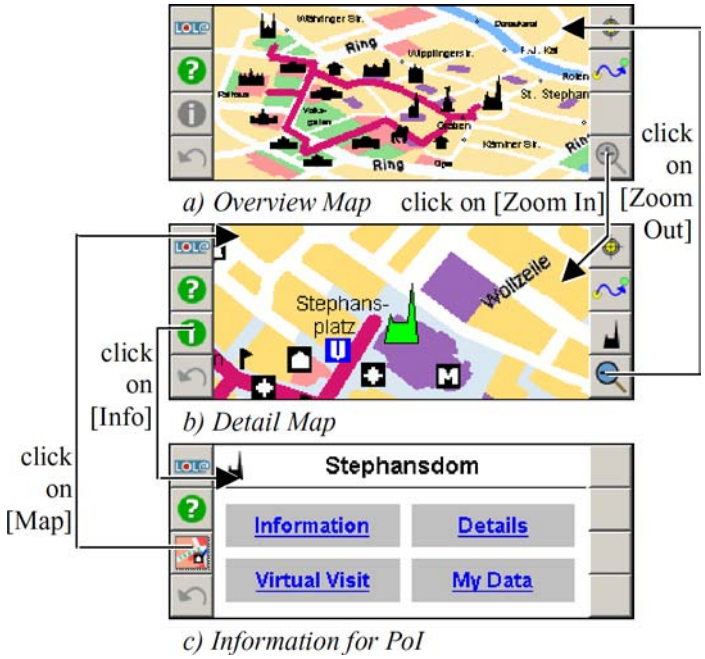


Figure 3.6: LoL@ - the local location assistant (from [Anegg et al., 2002]).

In order to determine the user's current position, the system relies on GPS and/or user interaction, and it allows the user to turn 'positioning' on and off. In the later case, LoL@ relies entirely on the user: she either has to manually tell the system that she has reached the end of the current tour segment, or to select the sight, where she is currently located. With 'positioning' turned on, the user's position is determined using GPS. Only in this case is her position displayed on the map. The user's abilities and properties as well as her current context are not taken into account during computation. Similarly, the task the user is currently performing does have no impact on system behavior beyond the selection of available options.

Concerning the adaptation to varying quality of positional information, the system provides several ways to deal with this issue. On the one hand, imprecise positions can be presented graphically by a shaded circle on the map which represents the area of uncertainty.[6] On the other hand, a simple form of interaction can be used to compensate for low precision: the system may ask the user to select her current location from a list of street names (along with house numbers). In the absence of any sensor data, the user can position herself by clicking on a sight on the map. While this set of adaptation strategies can address several issues such as the absence of positional information, there are some shortcomings: The system is unable to determine the position at a higher precision than an inherent threshold as it relies on sights and (possibly predefined) street segments. Furthermore, it is unclear whether it could deal with positions that are located far away from a point of interest). Additionally, the presentation of a (potentially long) list of street names with house numbers contradicts the intention of streamlined interaction.

[6]This technique is also applied within REAL (see 3.7).

Figure 3.6 shows the user interface of LoL@, which is based on a web metaphor and simple map interactions: The user can 'click' links, buttons, or icons with her pen, or access some commands using voice shortcuts. In addition to textual, pictorial, and animated output LoL@ can generate spoken output, and can therefore be classified as a multi-modal system. The system's main language is English, and there is no reference to any multi-lingual capabilities.

The architecture of LoL@ is based on the client-server paradigm: a 'terminal' with permanent network connection accesses a remote server where most processing takes place. The technical realization of the system is modular: several standard components such as streaming server and a database are being integrated. The application logic resides on the server as a set of Java servlets [Sun Microsystems, Inc., 2002] except for the display and direct interaction handling, which is located on the mobile device in the form of a web browser. The interaction between those components is also based on open standards such as the extensible markup language (XML) [World Wide Web Consortium, 2002a] and the extensible style sheet language (XSL) [World Wide Web Consortium, 2002b]. Information is passed using the hypertext transfer protocol (HTTP) [World Wide Web Consortium, 2002c].

3.7 REAL

The REAL project (an acronym for **Re**source-**A**daptive **L**ocalization) is a subproject of the Special Research Center 378 (Resource-adaptive cognitive processes)[7] sponsored by the German Research Foundation (DFG)[8] [Baus, 2002, Baus et al., 2002]. Within REAL two systems for assisting pedestrians in an airport scenario have been developed: an outdoor system called ARREAL (augmented reality REAL – see figure 3.7) and an indoor version called IRREAL (infra-red REAL - see figure 3.9). While the first one runs on a standard sub-notebook and is self-sufficient, the later one consists of a PDA, an infrastructure of infrared-beacons, and a server which is responsible for planing routes and generating presentations. This server also provides an info kiosk service (see figure 3.8), where the user can access rich information about the environment.

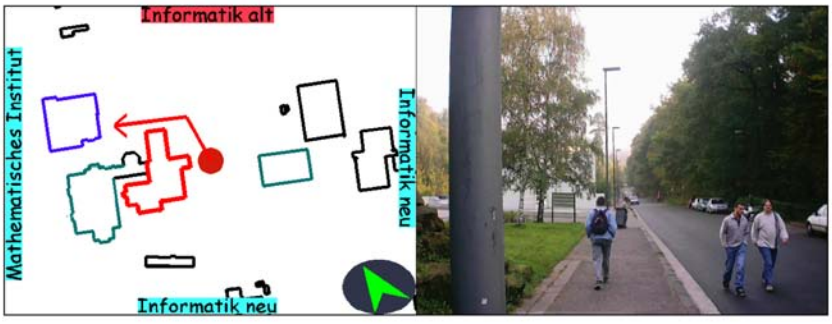


Figure 3.7: ARREAL – Augmented Reality Resource-Adaptive Localization: system output and usage scenario (from [Baus, 2002]).

[7] Sonderforschungsbereich 378: Resourcenadaptive kognitive Prozesse
[8] Deutsche Forschungsgemeinschaft

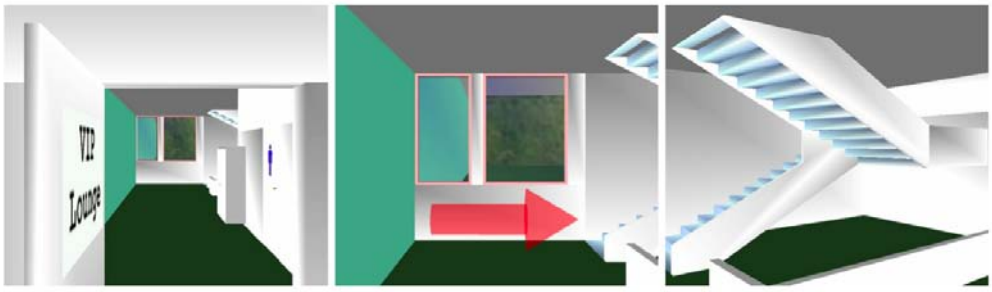


Figure 3.8: REAL – 3D directions at the information kiosk (from [Baus, 2002]).

All these sub-systems have been seamlessly integrated so that it is possible to use the solution based on the sub-notebook while entering or leaving a building without loss of service or any kind of disruption. This transparent transition from indoor to outdoor usage and vice versa is a key feature of REAL, which sets it apart from other systems.

The REAL system provides one main service: assisting a pedestrian in all stages of navigating an area. Additionally, it offers limited support for the provision of information on objects along the route. In order to determine the position of the user, the system relies on two complementary approaches. Outside buildings, GPS serves as a means to track the user. When entering a building, the system transparently switches to positioning by infrared beacons that are mounted at strategical locations. These beacons use high energy infrared emitters to transmit localized information (such as directions) to the user's device according to a sophisticated transmission schema [Baus et al., 1999]. Since the client (e. g. a PDA) only needs to filter the stream of information for an ID that was previously assigned to it, the term *passive location-awareness* was coined for this concept.[9]

Situational factors beyond the user's position are only considered during the computation of a route, which is initiated at the information kiosk (see figure 3.8). The kiosk can also display an animated virtual walkthrough for the corresponding route. When computing the route, the system takes into account various user-related and contextual factors such as the age or familiarity of the user with the environment or whether the route contains stairs. Since the system's main purpose is navigational assistance, it does not have to take into account the current task of the user.

Resource-adaptation plays a central role in REAL. Consequently, the system tries to adapt to various resource restrictions on different levels. Cognitive resources such as time-pressure or a secondary task beside navigation influence the computation of the route as well as the presentation of the corresponding directions. In this context, technical resources such as the number and location of infrared beacons along the route also have an impact. Additionally, the presentation of the route at the information kiosk (prior to actually following it) is adapted to honor these resource restrictions. The system even employs different strategies for planning the presentation according to the available resources. When the user actually follows the route, her walking speed deter-

[9] as opposed to *active location-awareness* where the client needs to compute localizations by itself, e. g. based on GPS signals

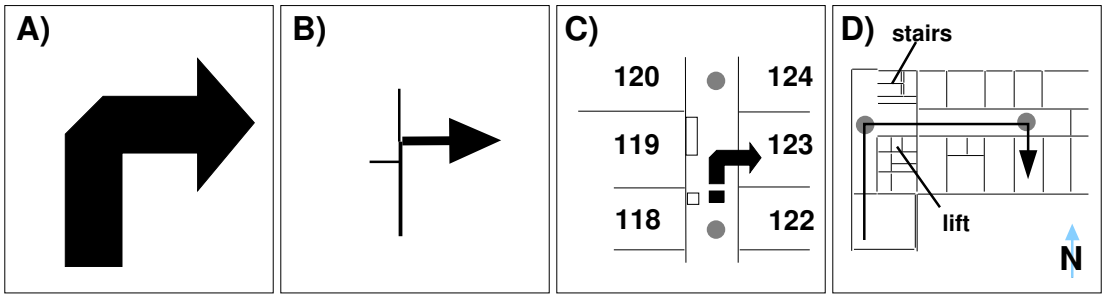


Figure 3.9: IRREAL – mobile indoor guide (from [Baus, 2002]): presentations adapted to accuracy of positional information (from A: precise information to D: no information).

mines the level of detail of the presentation. The probabilistic transmission protocol ensures that the user receives crucial information even if she follows the route with great speed. In summary, REAL belongs to the group of *resource-adaptive* systems. While the system provides no means to adapt to missing information, it has some simple means to determine the user's current position by interacting with her: by clicking marked areas on an overview map (see subfigure C) and D) of figure 3.9), she can tell the system that she is located at that position. This enables REAL to provide a more detailed map and information specifically tailored to the position indicated by the user (see subfigure A) and B) of figure 3.9).

Since the system relies almost entirely on graphical presentations (such as arrows, maps, icons, and three-dimensional animations, only very few verbal output is generated. Currently, there are two versions of the system, one in German and another one in English. Consequently, there is no internal language-independent representation. The system is multi-medial and predominantly multi-modal. The architecture of REAL is heterogeneous: On the one hand, there is a stand-alone variant that incorporates outdoor as well as indoor navigation services. On the other hand, there is the PDA-based variant that relies on a 'smart' infrastructure in order to function. All in all, several components – realized in various languages and on different operation systems – interact to provide the navigation services of REAL. Interaction between these relies on proprietary protocols and messages.

3.8 SmartKom

SmartKom [SmartKom Research Consortium, 2002] is a large research effort sponsored by the German Research Foundation (Deutsche Forschungsgemeinschaft – DFG), where several academic and industrial partners cooperate to develop a prototype for future communication devices. One sub-project focusses on improving communication support in a mobile setting. The mobile component of SmartKom [Wahlster et al., 2001] aims at providing a mobile user with easy-to-use and tightly integrated communication services such as on-the-fly booking of seats at nearby movie theaters. Starting mainly from the results obtained in the Verbmobil project [Wahlster, 2000] but also from Deep Map [Malaka and Zipf, 2000], a first prototype is currently under development. Figure 6.2 shows a vision of the final device and its interface.

Figure 3.10: SmartKom - mobile component (from [Wahlster, 2003]).

While the final version of SmartKom is envisioned to provide a broad range of location-based service, the current prototype provides three main services: guidance, tour recommendation, and location based online booking of tickets for movie theaters. Since one of the main goals of the SmartKom project is the tight integration of all services and devices, SmartKom can either use a standard GPS receiver or the sensor array of a modern car (GPS, wheel sensors, accelerometer, etc.) for positioning. In analogy to REAL, the mobile version of SmartKom supports the seamless transition from in-car to outdoor pedestrian usage, and adapts its presentation accordingly. The present prototype does not feature a user model, and contextual information is only taken into account when generating presentations. However, the task the user is currently performing does have no impact on the system's behavior or computations.

Similarly, SmartKom does not consider cognitive or technical resources except for the size of the screen when generating presentations. The lack of relevant information currently leads to the failure of the service that requires it, but does not affect the functioning of the system.[10] This robustness could be further improved by communicating the cause of the failure to the system. As SmartKom relies on a full-scale planning process, this feature could easily be realized. In terms of adapting to positional information of varying quality, there are no functions realized or planned.

Since the system employs technology from the Verbmobil project, it disposes of highly sophisticated components for natural language generation. Not only are these capable of generating output in various languages but also they can do so in a highly dynamic and adaptive way. Similarly, SmartKom is an inherently multi-modal system, both on the output and on the input side. It is one of the first systems to integrate a wide range of modalities from speech over gestures to mimic expressions, which enable it to even recognize sarcastic utterances. Within the project, a markup language was developped to capture multi-modal interactions (M3L, multi-modal markup language [SmartKom Research Consortium, 2002]).

The size of the project naturally led to a rigidly modular architecture which is based on multiple blackboards that are used to pass information between various components [Wahlster, 2001]. These components are written in different languages and run on various operation systems, which is one reason why the content of their interactions is encoded using an XML-based format.

[10] i. e. the system continues to function properly after the service failed

3.9 Discussion

Throughout this chapter, we presented several systems that provide services related to space, and we discussed their respective features and shortcomings. However, we have to state the list of systems we reviewed contains only a selection from a large number of related research projects. For example, there are several EU funded projects such as CRUMPET (creation of user-friendly mobile services personalized for tourism) [Poslad et al., 2001] or PEACH (personal experience with active cultural heritage) [PEACH, 2003] that aim at building navigational assistant systems. In addition, further projects are continuously created such as the new Special Research Center "Umgebungsmodelle für kontextbezogene Systeme" (environment models for context-aware systems) [Rothermel, 2003, Hohl et al., 1999], and virtually every car-manufacturer nowadays offers more or less sophisticated car navigation systems. Baus [Baus, 2002] provides an extensive list of location-aware systems. Our goal in deciding which systems to review here was to select a representative subset of all projects and systems concerned with situated interaction on spatial topics. Therefore, we focussed on systems that offer unique services (such as the transparent transition between indoor and outdoor usage) or that have been influential throughout the field (such as the Cyberguide project.)

Table 3.1 summarizes the discussion, and compares the systems reviewed to SISTO on Deep Map, the implementation of the approach we are going to present in subsequent chapters. In this table, entries in round brackets indicate that the corresponding functionality is either severely limited or not realized yet. (For example, CyberAssist does not yet provide guidance.) The symbols $\oplus$ and $\ominus$ stand for 'does apply' respectively 'does not apply'. SISTO, for example, takes into account the task the user is performing, while SmartKom does not.

We compared all systems along several dimensions. Since one goal of the model presented in the following chapter is easy adaptation to various tasks, we first reviewed what services the systems provide. From the table, we can see that some systems such as SmartKom or GUIDE do indeed offer a number of different services. However, since they do not rely on a modular and unified underlying model for spatial reasoning, none provides as many services as SISTO does. In addition, SISTO does also allow for easy extension due to the decompositional structure of the model presented in the next chapter.

In terms of positioning, roughly half of the systems rely on the Global Positioning System (GPS). Another large group uses light to determine the user's current position (either infrared or white light). Some systems have been designed from the beginning to support various sensors, i. e. they are adaptable with regard to the technology used to measure the user's position. Four systems (GUIDE, LOL@, REAL, and SISTO) include some means of interacting with the user to determine her position. These capabilities range from simply clicking on designated alternative positions (REAL) over static list to choose from (GUIDE) to dynamic lists based on the last known position of the user (LOL@). Only SISTO includes a sophisticated model based on position history, situational knowledge, and visibility (described in detail in 5.3) which allows for the dynamic generation of interactions tailored to the current user and situation.

The consideration of situational factors is another relevant feature for systems aimed at real-world use. We therefore compared all systems in terms of whether they take into account user- and context-related information. Additionally, we analyzed how they handle the impact of the user's current task. Roughly half of the reviewed systems includes user-related information but

they differ greatly in how this information is included. Hippie contains a sophisticated user-model, which is continuously updated, and which is used throughout the entire system. GUIDE as well as REAL rely on a static user-model that is used to adapt the generated presentations and their content in the former case. In the later case it only affects tour planning. SISTO currently also uses a static user model[11] but takes user-related information into account at all stages of computation. This comparison is nearly mirrored in the case of contextual information and its inclusion into the systems. GUIDE as well as REAL take into account contextual information for specific tasks: presentation and content selection in the former case and during tour planning in the later case. SmartKom distinguishes three 'scenarios', which are used to adapt the presentation accordingly. SISTO currently relies on a small static contextual model but considers contextual information on all stages of computation (see chapter 4). SISTO on Deep Map is also the only system that takes into account the task the user is currently performing, e. g. in object evaluation. Therefore, the same component can handle similar jobs in various contexts, which is not necessarily the case with the other systems.

A further very relevant feature set for real-world applications is their ability to adapt to changes in their physical and virtual environment. Resource limitations on the cognitive and technical side fall into that category. Unfortunately, REAL is the only system that can dynamically adapt to the varying availability of resources. Although most other systems have been optimized for mobile use, they are – at most – resource-adapted (following Wahlster and Tack's taxonomy [Wahlster and Tack, 1997]). While the model presented in chapter 4 and especially the adaptation strategies based thereon (see chapter 5) allow for resource-adaptation, the current implementation (SISTO) does not include such capabilities.

A further common problem in real-world applications lies in the lack of relevant information: often, information (such as situational factors or database entries for world objects) is only partially available or not at all. In this case, a system should gracefully degrade instead of abruptly failing. Only GUIDE, SmartKom, and SISTO incorporate this feature. GUIDE can handle network outages – which result in the unavailability of the central database – by relying on a scaled-down local version. SmartKom has been designed to be robust against such events: although it does not compensate for it, it is able to inform the user about the event and to continue to function properly. SISTO can handle the loss of network connection in a similar way as GUIDE does. While the underlying model is designed to handle the lack of relevant information, this part of it has not yet been implemented.

Since knowledge about the user's current position is a central factor in determining her current situation, it is highly important for a real-world application to be able to adapt to varying quality of positional information. Consequently, all but three of the systems reviewed in this chapter provide some means to address this issue. Cyberguide, CyberAssist, and SmartKom currently do not dispose of adaptation mechanisms. Hippie distinguishes two levels of granularity: either the room is known, which the user is currently located in, or the exhibit that she is facing. TellMaris entirely relies on manual navigation (although later versions will employ GPS), and can therefore operate independently of the user's current position. GUIDE provides a means to select the current position from a (static) list of sights. LOL@ goes one step further by dynamically generating a list

[11] but since this is an external component that SISTO queries at demand, it could easily be replaced with a more sophisticated version

of street names based on the last known position, but it can also communicate the imprecision of a position to the user. To do so, it displays the current position on the map as a circle, which grows with imprecision. REAL relies on the same metaphor, and also provides some simple means of interaction: clicking on certain highlighted spots on the map allows the user to specify her current position more precisely.

While most systems provide one or more means to address varying quality of positional information, there are some shortcomings. On the one hand, the communication of imprecision is tied to a specific modus (i. e. a 2D map). On the other hand, the interactive means provided by GUIDE, LOL@, and REAL are limited in several ways. Either there is a large static list of positions/sights to select from (GUIDE), or a list of street names with house numbers that is dynamically generated (LOL@). In both cases the interaction is not very user-friendly, since scrolling through a large list of choices is not feasible on a mobile device. Providing static spots to click on in order to specify one's position (REAL) not only restricts positioning to those spots but also inherently ties this function to a specific modus. SISTO goes beyond this in several ways: on the one hand, it provides a sophisticated algorithm for dynamic interaction that is optimized for speed and minimal interaction (see 5.3). On the other hand, these interactions are not tied to a specific modus. Furthermore, it can employ induced frames of reference to address the (partial) lack or imprecision of positional information. Finally, the underlying model allows for various adaptation strategies (not all of which have yet been implemented).

The interface of a system and the available means of interaction are the parts of the system that are most apparent to its user, and that therefore greatly her perception of the system. Hence, we included a comparison in terms of support of natural language and multi-modality in our review. From table 3.1 we can conclude that most systems are statically tied to one language. GUIDE and REAL allow for several different languages, but are still static. Only SmartKom and SISTO support dynamic multi-lingual interaction by introducing a semantic layer that encodes interactions in a way that is independent of the actual target language. While SISTO relies on the preverbal message (see 4.2), SmartKom goes a step further by employing a full plan-based mechanism and sophisticated language processing features. The same is true in terms of multi-modality, since SmartKom has been specifically designed to account for various modalities such as speech, gestures, and mimic expressions. While SISTO could (due to the preverbal message) in principle support a comparable range of modalities, its current implementation is limited to verbal/textual input and pointing (which also applies to Hippie, LOL@, and REAL).

Another point of practical importance is the architecture of a system. While not being directly apparent to the user, it has a serious impact on the system in terms of extensibility and adaptability. Hence, we compared all systems with respect to what type of architecture they are built on, and how interaction between different components is realized. From table 3.1 it is apparent that although all systems rely on a modular architecture, they do so in several ways. Hippie, GUIDE, and LOL@ are based on the client-server paradigm: a 'client' (i. e. a web browser) accesses a 'server' (i. e. a web server). While this approach allows for the easy addition of multiple clients, it highly depends on a reliable connection between client and server, which is not always a given (e. g. in wireless networks). In addition, the server is a single point of failure – making the systems relying on it less robust than less centralized approaches. Cyberguide and TellMaris are built using interacting applications. Although this approach is more decentralized, it has some drawbacks: On the one hand applications may be specifically designed for a certain device/platform,

which may hinder dynamic distribution. On the other hand, their interaction is often problematic because different programming languages may have to communicate and there is no standard on how to realize this. The architecture of REAL is a hybrid that combines a client-server approach with that of interacting applications, therefore inheriting the respective advantages and drawbacks. SmartKom relies on an approach that was originally developed in the Verbmobil project Multiple blackboards are used to enable distributed applications to interoperate. While this approach allows for easy extension, interactions between various components are implicit and hard to track. Multi-agent systems address this issue by introducing an explicit message passing mechanism, and also include standardized look-up services. This enables systems relying on this approach (CyberAssist and SISTO) to compensate for failures of certain components, to dynamically add or remove components, and to transparently relocate components to other platforms. In addition, multi-agent systems facilitate the encoding of contents using a standard language that is explicitly defined. Aside from CyberAssist and SISTO, only LOL@ and SmartKom employ such an encoding scheme, while all other systems rely on a proprietary interaction language.

	Cyber-guide	Hippie HIPS	GUIDE	Cyber-Assist	TellMaris	LOL@	REAL	SmartKom	SISTO on Deep Map
Services	information, communication	guidance, information, (localization), communication	guidance, information, reservation, communication	(shopping/ ticketing assistant), (guidance), (spatial reminder), (disaster mitigation)	navigation	guidance, information, tour diary	guidance, (information)	guidance, information, map interaction	guidance, info., localization, data collection, map interaction, (reservation)
Positioning	GPS, adaptable	infrared beacons, compass	network cells, interaction	modulated light, adaptable	manually	GPS, interaction	infrared beacons, GPS, compass, interaction	GPS, adaptable	GPS, adaptable, interaction
Situational factors									
User	⊖	⊕	presentation, content selection	(envisioned)	⊖	⊖	during tour planing	⊖	⊕
Context	⊖	⊖	presentation, content selection	(envisioned)	⊖	⊖	during tour planing	presentation	⊕
Task	⊖	⊖	⊖	⊖	⊖	⊖	⊖	⊖	⊕
Adaptation capabilities									
Cognitive Resources	⊖	⊖	⊖	⊖	⊖	⊖	⊕	⊖	(⊖)
Technical Resources	⊖	⊖	⊖	⊖	⊖	⊖	⊕	⊖	(⊖)

Table 3.1: Comparison of systems providing navigational assistance and related services (entries in brackets are not entirely realized, ⊕ and ⊖ provides the corresponding feature) - part one

	Cyber-guide	Hippie HIPS	GUIDE	Cyber-Assist	TellMaris	LOL@	REAL	SmartKom	SISTO on Deep Map
Adaptation capabilities (continued)									
Lack of information	⊖	⊖	local proxy/cache	⊖	⊖	⊖	⊖	robust	(⊕)
Position	⊖	two levels of precision (room vs. exhibit)	simple interaction	⊖	synchronized manual navigation in 3D and 2D	communication of imprecision, interaction	communication of imprecision, simple interaction	⊖	communication of imprecision, inference, interaction
Interface and user interaction									
Language	English	English	static, multilingual	(Japanese)	English	English	German, English	dynamic, multilingual	dynamic, multilingual
Multimodality	⊖	⊕	⊖	(envisioned)	⊖	⊕	⊕	⊕⊕	⊕
Architecture									
Type	interacting applications	client-server	client-server	multi-agent system	interacting applications	client-server	hybrid	multi-blackboard	multi-agent system
Interaction	proprietary	proprietary	proprietary object model	standard-based	proprietary object model	standard-based	proprietary	standard-based	standard-based

Table 3.2: Comparison of systems providing navigational assistance and related services (entries in brackets are not entirely realized, ⊕ and ⊖ provides the corresponding feature)- part two

3.10 Summary

In this chapter we reviewed a selection of recent or influential systems that incorporate spatial interaction in a complex scenario. Most of these systems stem either from the realm of outdoor tourist guides (GUIDE, TellMaris, LoL@) or (indoor) exhibition guides (CyberGuide, Hippie). While the REAL system mainly supports navigation, it stands out by allowing for the seamless transition from indoor to outdoor navigation and vice versa. Only a few systems have a broader field of application: CyberAssist and SmartKom are still under development and aim at providing a variety of services to a mobile user – ranging from shopping assistance over spatial reminders to location-based ticket reservation and seamless integration with more stationary systems. SISTO on Deep Map (the implementation of our approach) is the only system that has been completely implemented and that supports a variety of (interactive) tasks related to space.

In 3.9 we compared all the systems along a number of relevant dimensions (see also table 3.1 for an overview). In doing so, we mainly concentrated on six categories. First we reviewed the systems in terms of what services they provide and on which technology they rely for determining the user's current position. For the later dimension, two key factors were whether the system was able to easily adapt to further sensors and whether it could bring the user into the loop.

The next field of comparison was related to the inclusion of situational factors, a key advantage of SISTO on Deep Map. Here, we compared the systems in terms of whether they take into account user- and context-related factors as well as the task that the user is currently performing. A dimension that is equally important in real-world applications is the ability of a system to adapt to unforeseen events. Therefore, we reviewed a systems according to whether they can adapt to cognitive and/or technical resources, the lack of information, and positional information of varying quality and availability.

The third field of comparison we considered was the interface and the way a user can interact with the system. The two dimension we analyzed in this context were the ability to handle various natural languages and whether the system is able to cope with multi-modal input and output. Finally, we compared all systems in terms of their underlying architecture, how it was designed, what type of architecture was used, and whether the interaction between the components of a system was implicit or explicit.

4 Modeling situation aware spatial processes

In the two previous chapters, we have discussed typical tasks related to space (chapter 2) and situative factors that are important in the context of navigational assistance and situated interaction with a mobile system (chapter 3). The next step now consists in analyzing how those factors influence these tasks, and how we can model their relationship in a flexible way. We will therefore first examine the fundamental sub-tasks identified in section 2.5.2, and present computational approaches that take into account the current situation (in 4.1).

Based on these basic processes, we will then present means to realize more complex ones. Most of the time, these complex processes involve interaction between the user and the system of different complexity. In order to support different languages, media and modalities, we propose a language- and media-independent encoding of these interactions (in 4.2). This encoding is then applied within the mechanisms for modeling complex tasks related to space (in 4.3), which include the typical tasks reviewed in chapter 2 as well as several additional ones that are relevant in the context of situation-aware mobile system.

4.1 Basic spatial reasoning processes

As we have pointed out in 2.5, the analysis of common tasks related to space reveals several interacting sub-tasks or basic processes. In this section we will therefore present a modular approach to modeling those tasks in a way that facilitates interaction and emphasizes the influence of contextual factors.

4.1.1 Object evaluation

The evaluation of objects according to a number of spatial (and non-spatial) factors plays a central role in many spatial reasoning processes, e. g. when selecting an anchor object for a localization or when determining the object that was intended by the user. Consequently, it is highly important to find an extensible framework that is flexible enough to compensate missing information but expressive enough to take into account situational changes. In the following sections, we will therefore first analyze what factors are relevant in this process, before introducing the multi-attribute utility theory, which we will then use to model object evaluation in a way that satisfies the requirements mentioned above.

Relevant spatial and non-spatial factors

Although it is possible to identify several factors that are relevant in almost any of the tasks analyzed in this thesis, it is important to remember that relevance is a *relative* concept: a factor that has a high impact in the context of one task may be irrelevant in the context of another, and vice versa. In section 4.1.1 we will therefore present an approach that evaluates objects based on the task that is currently performed. While we will limit our analysis to the complex tasks listed in section 4.3, the model can easily be extended to further tasks.

Visual salience is an important factor in many tasks such as selecting an anchor object for a localization or determining landmarks. But it is also a fairly complex concept that needs to be broken down further in order to make its inclusion in the reasoning process feasible. In his PhD. thesis, Maaß [Maaß, 1999] proposed a model for object evaluation in the context of incremental route instructions that takes into account *color, height*, and *width* of an object. Height and width may be seen as a very rough approximation of the *shape* of an object, which are much easier to handle than a full analysis of the characteristics and differences in shape (see [Krüger, 2000]). Another relevant factor (especially in the context of a mobile assistant) is the *visibility* of an object from key positions such as the user's current location, the route she is currently following, or the target object of that route. The relative *proximity* of an object to those locations can also play an important role.

In addition to these spatial and perceptual factors, several non-spatial factors have to be considered as well. In analogy to the concept of visual salience, we can define a *conceptual salience*, which captures the 'uniqueness' of an object on the conceptual level. Obviously, we have to break this concept down into several measurable factors. The *function* of an object may be one of those factors. For example, if one of a set of similar buildings contains a pharmacy while the others are all residential buildings, the pharmacy certainly has a high conceptual salience. On a more technical level, we may use some meta-information to extract further knowledge from the inter-

Factor	Identification	Localization	Guidance	Translation	Data-Collection	Implementation
visual and spatial salience						
color	⊕	⊕	⊕	⊕	⊕	⊖
width	⊕	⊕	⊕	⊕	⊕	⊖
height	⊕	⊕	⊕	⊕	⊕	⊖
visibility	⊕	⊕	⊕	⊕	⊕	⊕
proximity	⊕	⊕	⊕	⊕	⊕	⊕
conceptual salience						
function	⊕	⊕	⊕	⊕	⊖	⊕
amount of data	⊕	⊕	⊕	⊕	⊖	⊕
subjective relevance						
matching user interest	⊕	⊕	⊕	⊖	⊖	⊖
matching intention	⊕	⊖	⊕	⊕	⊖	⊕
dialog history	⊕	⊕	⊕	⊕	⊖	⊕

Table 4.1: Factors relevant in complex tasks related to space: ⊖ and ⊕ indicate strong and weak impact, respectively whether a factor was realized in implementation.

nal database that a system has to incorporate in order to provide its services. Since information is rarely collected randomly, the *amount of data* available for a given object most frequently indicates that this object is 'interesting', or conceptually salient.

In a mobile tourist guide, it may also be sensible to consider the *historic period* from which an object stems. However, we may include a more general factor that subsumes this factor: the *subjective relevance* an object has for a specific user. We can then analyze facets such as the historic period (or artistic period) in terms of whether it matches the *user's interest*. In the context of the task the user and/or the system perform at a given moment it makes also sense to consider whether an object corresponds to the *current target object* or is closely *related to her intention*, e. g. the object to be localized or the destination of the current route.

An additional factor that is relevant in many tasks can be deduced from the *dialog history* that records the interaction between the system and the user: Whether or not an object has been mentioned before determines, among others, whether the user knows about this object and its identity. A detailed analysis should also take into account if the object has been mentioned by the user or by the system, once or more frequently, and how much time has passed since the last mentioning. Furthermore, we should consider how many interactions took place during that time.

Table 4.1 list all the factors that we described in the previous paragraphs as well as their relevance for the complex tasks presented in section 4.3 and whether there were included in the

prototypical implementation. While we were able to obtain promising results in terms of object evaluation/selection in the prototypical system built to take into account these factors (see chapter 6), it is clear that there is a multitude of further factors that could be relevant for the tasks we considered as well as for other tasks. Similarly, the exact impact of a factor in the context of a specific task – whether it increases or decreases the evaluation of an object and to what degree – has to be determined by empirical studies. The entries listed in the table were obtained through experimentation with the implementation and should only be seen as first reasonable estimates. However, if we can easily modify the influence of a specific factor, the adaptation to new empirical evidence is a straightforward task. Therefore, we will introduce a flexible modeling approach for object evaluation on the following pages that has been developed specifically to facilitate the addition of further factors and tasks as well as the modification of their respective influence.

Multi-attribute utility theory

The *Multi-attribute utility theory* (MAUT) [Keeney et al., 1976] is an evaluation scheme stemming from decision analysis [von Winterfeldt and Edwards, 1986]. It has been applied to a wide range of different fields such as user modeling or decision support, and it has been demonstrated that it is possible to express a number of calculi using MAUT (see, for example, [Schäfer, 2001]). In its elementary form, an evaluation function $v(x)$ is defined that assigns each object x a value, which is computed by a weighted sum over the relevant value dimensions:

$$v(x) = \sum_{i=1}^{n} w_i v_i(x) \tag{4.1}$$

Each value dimension d_i is associated with an evaluation function $v_i(x)$, which maps the value of an object x in that dimension to an evaluation value that corresponds to the *utility* of that specific characteristic to the evaluator. The impact of each individual dimension on the overall evaluation is assigned a weight w_i, which determines the *relative importance* of that dimension. Usually, these weights are normalized, i. e. their sum equals one. The definition of the evaluation function $v_i(x)$ for a dimension d_i corresponds very closely to that of the overall evaluation function:

$$v_i(x) = \sum_{a \in A_i} w_{ai} v_{ai}(l(a)) \tag{4.2}$$

Again, a weighted sum is computed, but this time over the evaluation $v_{ai}(l(a))$ of all attributes $a \in A_i$ that are relevant on dimension d_i. v_{ai} maps the actual level $l(a)$ of an attribute to an evaluation value, which is then weighted with w_{ai} according to the relative importance of the attribute on this dimension. The weights w_{ai} are usually also normalized.

This basic form of MAUT can be extended and modified in several ways. On the one hand, the evaluation functions can be modified, e. g. by using a weighted product instead of a sum. On the other hand, the nesting of evaluation functions (overall evaluation depends on dimension evaluation, which depends on attribute evaluation) can be taken further if the complexity of a domain requires additional abstractions. Furthermore, the weights can be computed dynamically, e. g. depending on the actual situation. Finally, the weighting function can be modified to take into account missing information on certain dimensions while ensuring comparability (see [von Winterfeldt and Edwards, 1986] for a thorough analysis of MAUT and its properties).

Applying MAUT to object evaluation

Its flexibility and expressional power predestine MAUT to capture complex relationships, such as those in situation-aware spatial reasoning, and to integrate contextual and user-related influences [Jameson, 2001]. Even if we assume a single-factor and purely spatial object evaluation (e. g. based solely on closeness to the listener), we can express this in terms of MAUT using an evaluation function $oe(x)$ that simply maps the object to a rating according to the spatial process used to evaluate the object. Obviously, the extension of the evaluation function $oe(x)$ to additional spatial and non-spatial factors such as visibility, proximity to speaker, size, function, etc., is a straightforward task. We can therefore define $oe(x)$ in the following way:

$$oe(x) = \sum_{i=1}^{n} w_i(t) oe_i(x) \tag{4.3}$$

In this equation, $oe_i(x)$ is the object evaluation function on dimension i of the set of length n of all dimensions, which are defined by all spatial and situational factors. The weight $w_i(t)$ of a value on dimension i depends on the task t, which triggered the object evaluation. Modeling w_i in this way is an important enhancement of the basic MAUT as it allows for a flexible weight function that is determined by the current task. Consequently, it is now possible to easily model the influence that each dimension i has in the context of a specific task t. The evaluation function for dimension i, $oe_i(x)$, can be defined analogously as

$$oe_i(x) = \sum_{a \in A_i} w_{ai}(t) oe_{ai}(l(a)) \tag{4.4}$$

where A_i is the set of all attributes relevant in dimension i, and oe_{ai} the evaluation function that maps the actual level $l(a)$ of an attribute to an evaluation. $w_{ai}(t)$ expresses the weight of attribute a for the evaluation in dimension i. Making the weight function dependent on task t again allows for an easy modeling of the influence that each attribute a has in the context of this task.

If information on a dimension is missing for one or more objects, there are several ways to address this. One easy way to cope with such a situation lays in assuming a default (median) value, which can either be deduced locally (by computing the average value for all currently available information on this dimension), or globally (by computing the average value of all possible values on this dimension). Whenever there is no information at all on a dimension i for any object, the corresponding evaluation function $oe_i(x)$ can be removed entirely from the overall evaluation function. However, if the corresponding evaluations have to be compared to previous ones that included information of this dimension, it is necessary to either adjust the weights, or to rely on default values.

$$oe_{color}(x) = w_{color}(t) sal(\vec{color}\,(x)) \tag{4.5}$$

$$oe_{height}(x) = w_{height}(t) \left| \frac{1}{|\mathfrak{O}|} \sum_{y \in \mathfrak{O}} height(y) - height(x) \right| \tag{4.6}$$

$$oe_{width}(x) = w_{width}(t) \left| \frac{1}{|\mathfrak{O}|} \sum_{y \in \mathfrak{O}} width(Y) - width(x) \right| \tag{4.7}$$

$$oe_{shape}(x) = min\left(y \in \mathfrak{O} \setminus \{x\} \left| \frac{1}{(n(x)+n(y)) * \sqrt{2}} \sum_{i=1}^{n(x)+n(y)} d_i \right.\right) \quad (4.8)$$

In the following paragraphs we will discuss the individual evaluation functions for the factors listed in 4.1.1. Equation 4.5 through 4.8 list several dimensions that are relevant for determining the visual salience of an object. These formulas stem from [Maaß, 1999], except for equation 4.8 which was developed by [Krüger, 2000], and they can easily be integrated into the evaluation process using MAUT. In order to compute the salience of an object in terms of its color [Maaß, 1999] proposes a model based on the distance in RGB-color space that does not only take into account the local distribution of color but also the 'coloration' of the entire viewing area. The complete formula is more elaborate than the one shown in equation 4.5 but can be found in [Maaß, 1999] and is based on work presented in [Maaß et al., 1995]. (Gapp [Gapp, 1997] proposes an similar approach to determine the salience of color, which can be used alternatively.) Equation 4.6 and 4.7 for evaluating the 'distinctiveness' of the height and width of an object express the average difference of an object x to all other objects y in the set $\mathfrak{O}$ of current candidates.

Maaß [Maaß, 1999] suggested this rough approximation for evaluating differences in shape. In his thesis on automatic generation of abstractions in 3D, [Krüger, 2000] presents a more precise measure based on the $n(x)$ nodes that constitute an object x in 3D space: After an aligning projection into a sphere of diameter one, the similarity of two objects x and y is computed by first adding the minimal distance of all $n(x)$ nodes of x to any node of y to the minimal distance of all $n(y)$ nodes of y to any node of y. Since the maximum distance is $\sqrt{2}$, the resulting sum can easily be normalized. Consequently, the resulting values range from 0 (x and y are identical) to 1 (x and y are two orthogonal lines of length 1). In equation 4.8, we compute this degree of similarity for all objects, and take the minimum value as an indicator on the dissimilarity of x.

$$\begin{aligned} oe_{visibility}(x) = {} & w_{user}(t)\frac{1}{2}\left(perc\_vis(user,x) + perc\_vis\_area(user,x)\right) + \\ & w_{target}(t)\frac{1}{2}\left(perc\_vis(target,x) + perc\_vis\_area(target,x)\right) + \\ & w_{route}(t)\frac{1}{2}\left(perc\_vis(route,x) + perc\_vis\_area(route,x)\right) \end{aligned} \quad (4.9)$$

As we have seen in 4.1.1, the visibility of an object is also an important factor when evaluating it, for example, in terms of its appropriateness as an anchor object in a localization. Equation 4.9 shows the corresponding evaluation function on the visibility dimension. It is composed of three distinct terms capturing the visibility of x seen from the user's current position, from the location of the current task's target object, and from the route the user may be following. The visibility itself consists of a function measuring the percentage of the object that is visible (*perc_vis*) and another one that returns the percentage of the entire viewing area that is occupied by the actual object (*perc_vis_area*). We selected this composition to take into account that it is not only important how much of an object is visible but also how much of the entire viewing area is actually covered by this object. Consider, for example, the situation when a user is standing in front of a large building: even though it covers her entire viewing area, she might only see a small portion of it. However, if she is looking at the same building from a distance, she might see it in its entirety but it may only cover a part of her viewing area. Using the approach we formulated in equation 4.9, we are able to capture both situations.

However, while we can easily compute the visibility of an object in this way, a high value does not guarantee that it is *recognizable.* For example, if an object stands out from a number of similar adjacent objects by a relatively small feature – say a fancy mailbox at one of a long row of terraced houses - and if this feature is covered, the object may not be recognizable although it is almost entirely visible. Both Maaß [Maaß, 1999] and Krüger [Krüger, 2000] point out the difficulties in determining salient and differentiating features of arbitrary objects. Consequently, a formula trying to capture recognizability would have to include a large number of factors. While we refrained from designing such a formula (instead providing a simplified version in equation 4.9, our approach allows for the integration of more elaborate modeling approaches such as the ones presented by Maaß and Krüger.

$$\begin{aligned} oe_{proximity}(x) &= w_{user}(t)proximity(user, x) + \\ & \quad w_{target}(t)proximity(target, x) + \\ & \quad w_{route}(t)proximity(route, x) \end{aligned} \tag{4.10}$$

The same approach used for capturing visibility can also benefit the modeling of the proximity of an object. Analogously, we can take into account the user's current position, the target object, and the current route as shown in equation 4.10. Aside from the geometrical and visual factors discussed in the previous paragraphs, we identified several factors contributing to the conceptual salience of an object in 4.1.1. Equation 4.11 shows the evaluation function corresponding to the dimension 'function', which grades the salience of an object in terms of how many other objects in $\mathfrak{O}$ share the same function fn: the fewer do, the higher the value returned.

However, often objects such as buildings have multiple functions (e. g. they contain a restaurant and a bookstore). We can then define $fn(x)$ as returning a set of functions $\mathfrak{F}_x$, and use $\left|\mathfrak{F}_x \cup \mathfrak{F}_y\right|$ instead of the simple comparison. However, people may perceive such objects as being composed of several distinct entities. If we want to account for this explicitly, we can model each component as well as all compositions as distinct objects and evaluate them separately, or we can introduce an inheritance hierarchy, where parts inherit properties from the containing object. We will discuss this in more detail in the context of pars-pro-toto deixis and the identification task in 4.3.1.

The amount of data available on an object is a second conceptual factor, which we can measure easily by 'counting' the entries in the database that stores (non-spatial) information associated with the object. Equation 4.12 shows a possible, relative measure, namely the number of entries $data(x)$ for object x divided by the maximum number of entries for any object $y \in \mathfrak{O}$. It is a straightforward task to model this aspect in a more detailed way by separately weighting each type of data in analogy to equation 4.10. For example, we can evaluate the length of textual information instead of just evaluating whether or not there is any text.

$$oe_{fn}(x) = w_{fn}(t)\left(1 - \frac{|\{y \in \mathfrak{O} \,|\, fn(y) = fn(x)\}|}{|\mathfrak{O}|}\right) \tag{4.11}$$

$$oe_{data}(x) = w_{data}(t)\frac{data(x)}{max(y \in \mathfrak{O}|data(y))} \tag{4.12}$$

$$\tag{4.13}$$

In addition to conceptual and visual salience we identified the subjective salience as a third influential concept in object evaluation. A first factor of this type is how well an object matches the

user's interest. In order to include it into the reasoning process, we need to have access to a user model that provides this kind of information, which is highly domain-dependent. In the context of a mobile tourist guide, historical information about the sights combined with knowledge about the user's preferred historic period may serve as a starting point to model this factor. These considerations apply to a second subjective factor as well: how closely an object is related to the user's current intention. For a mobile tourist guide with features such as guidance, object localization and information on sights, the target objects of those services can fall into this category. Multiple intentions or interests can be modeled using a weighted sum. Equation 4.14 and 4.15 list generic functions for intention and interest, which depend on external functions to evaluate an object in terms of its interest for the user, and its relationship to the user's current intention.

$$oe_{interest}(x) = w_{user}(t)interest(user, x) \tag{4.14}$$

$$oe_{intention}(x) = w_{user}(t)intention(user, x) \tag{4.15}$$

The third subjective factor we introduced in 4.1.1 is the fact, whether an object has been previously mentioned. Not only is this a strong indication whether the object is known to the user, but it has also a great impact when evaluating objects in the context of an identification: If an object has been mentioned before, its likelihood of being the target of a "What is this?" question decreases tremendously. If a system incorporates a dialog history that annotates each entry with a time and a turn and allows for time- and entry-based searching (such as the one included in the system we present in chapter 6), we can model this aspect in a very detailed way. This is advantageous as there are several aspects of being previously mentioned that should be taken into account. On the one hand, it makes a difference whether an object x was mentioned by the user or by the system. In the first case, it is very likely that the user is familiar with the object while in the second case, it is still possible that she did not pay attention to the system's output. On the other hand, we can distinguish between objects that have been mentioned recently and those, that have been mentioned long ago: While the user will probably remember the first ones readily, she might already have forgotten about the later ones.

Furthermore, the frequency of prementioning certainly influences the recall of an object: an object that has been mentioned ten times during the last five minutes is more likely to be remembered better than one that has been mentioned only once during the last couple of hours. Equation 4.16 tries to capture all those facets: there is a base value $mentioned(x)$ for being mentioned at all, to which we add increments. These increments include functions modeling whether x has been mentioned by the user ($mentioned\_by(user, x)$) or by the system ($mentioned\_by(system, x)$) as well as functions that capture the delay since the last mentioning ($delay(now, x)$) and the frequency ($times\_mentioned(x)$).

$$\begin{aligned} oe_{dialog}(x) = {} & w_{mentioned}(t)mentioned(x) + \\ & w_{user}(t)mentioned\_by(user, x) + w_{system}(t)mentioned\_by(system, x) \\ & w_{delay}(t)delay(now, x) + w_{frequency}(t)times\_mentioned(x) \end{aligned} \tag{4.16}$$

From a linguistic perspective, this formula is a simplification: a dialog most frequently is not simply linear in nature (i. e. has a flat discourse structure), but rather has a more complicated structure. For example, discussion of one topic may be interrupted to shortly clarify a subtopic

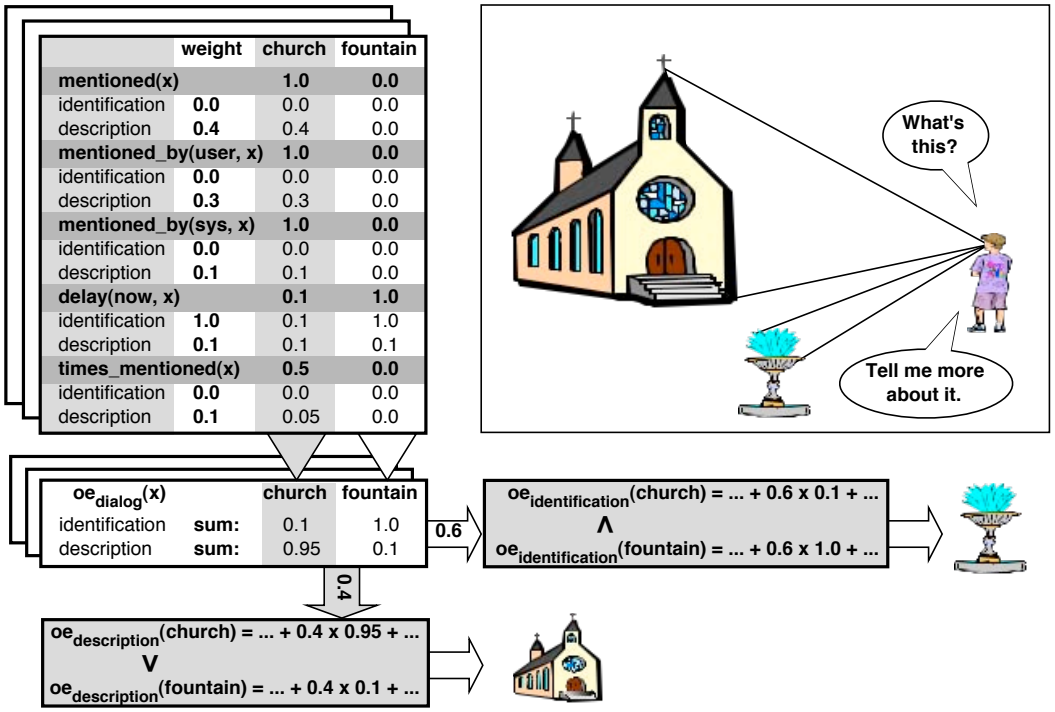


Figure 4.1: Object evaluation: An example

before continuing on the main topic. Hence, a stack-based discourse model is more appropriate to capture natural dialogs [Wahlster, 2000] but would considerably increase the complexity of equation 4.16. While it is possible to integrate such an approach, the corresponding linguistic considerations lie outside the scope of this thesis as well as the prototypical implementation, where we realized the simplified version shown in equation 4.16.

The factors, for which we introduced evaluation functions in the previous paragraphs, are certainly not the only ones that are relevant in the context of object evaluation. However, they are not only fairly general in scope as they apply in many different contexts but they are also highly relevant in various tasks. For example, using this approach it is a straightforward task to distinguish between *landmarks* and *routemarks* [Krieg-Brückner and Röfer, 1998], e. g. by adjusting the weights of proximity and visibility from the route accordingly. Whereas landmarks generally refer to objects that are highly recognizable to a large group of people [Lynch, 1960], the relevance of routemarks is deduced from a specific route. In a way, routemarks are a special case of *salient objects*, which we can define as objects that are recognizable or relevant in a specific task to a single individual user. Using the approach proposed above, we can model the distinction between landmarks, routemarks, and salient objects explicitly and at a high level of detail.

In order to illustrate the flexibility of this approach, figure 4.1 depicts an example scenario, which is taken from the application domain of the prototypical implementation presented in chapter 6: A tourist on a sightseeing tour through a city ask her mobile assistant two different questions

in the same situation. "What's this?" calls for the identification of an unknown object, and "Tell me more about it." calls for more information about an object that most likely is already known by the user. The table on the upper left hand side list the values for the attributes of the dialog history evaluation function $oe_{dialog}(x)$ as well as the corresponding weights for the tasks *identification* and *description*. The corresponding sums – respectively the result of $oe_{dialog}(x)$ is shown in the box below. Due to the different weights for both task, the evaluation for the description task is much higher for the church than for the fountain, while in case of the identification task, it is the other way around. In addition, the weight of $oe_{dialog}(x)$ within the overall object evaluation function $oe(x)$ is also different for the two task. In case of the description task, it is 0.4, which will result in the church being selected, and in case of the identification task, it is 0.6 in order to account for the fact that one usually requests the name for objects that have not been mentioned before. There are of course further dimensions (such as $oe_{visibility}(x)$ or $oe_{proximity}(x)$), which also have to be evaluated. For clarity reasons, they are just shown as additional boxes behind the tables but not listed explicitly in the figure, and we assume that they do not tilt the evaluation in another direction. However, the weights in the table are just examples. Empirical studies and/or machine learning mechanisms are required in order to identify appropriate values. Therefore, the weights as well as the factors and their modeling can only serve as a starting point for object evaluation. The modeling approach we proposed – MAUT – facilitates the inclusion of further (domain-dependent) factors, and allows for easy adjustment of weights and the application of learning algorithms.

4.1.2 Frames of reference

As we have seen in the previous chapter, a central component in most spatial reasoning processes is the consideration of spatial relationships between different objects. In order to unambiguously specify the location or direction of objects, a frame of reference (FOR) is required which structures the embedding space in a way that allows for relating to this structure. Following [Frank, 1998] a reference system or a frame of reference is specified by three characteristics: the origin of the coordinate system (which is independent of the kind of coordinate system used), its orientation and its handedness (which establishes the relation between the axes). There have been several proposals on how to classify frames of reference such as according to the way in which the origin is defined [Herskovits, 1986], or depending on the current scope [Montello, 1993].

Immediate frames of reference

From one point of view, three basic types of frames of reference are frequently distinguished: deictic, intrinsic, and extrinsic frame of reference. *Deictic* frames of reference designate those frames that inherit their origin, orientation and handedness from the speaker of an utterance. *Instrinsic* frames of reference are established based on an anchor object: it determines the origin of the coordinate system as well as its orientation. Depending on the type of object, the direction is derived from the topology, size, or shape of the object. For example, if the anchor object is a building, the orientation is often defined by a prominent front and/or by the location of the main entrance. *Extrinsic* frames of reference also inherit their origin from an anchor object. However, their orientation and handedness is not determined by intrinsic properties but rather by external factors such as the direction of motion.

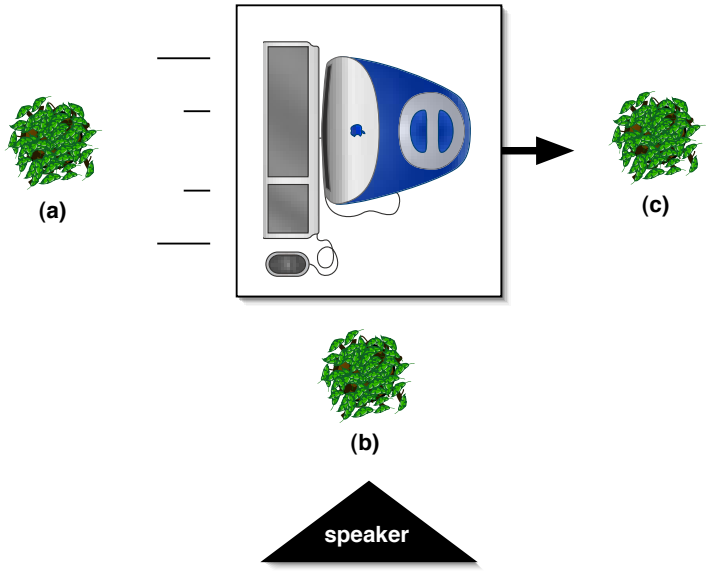


Figure 4.2: Intrinsic (a), deictic (b), and extrinsic (c) frames of reference

Figure 4.2 depicts examples for these three types of frames of reference for the localization "The plant is in front of the computer.". In case (a), an intrinsic frame of reference is used, which is established using the location of the computer and its intrinsic front (the screen). Case (b) shows a deictic frame of reference, which is determined by the location and orientation of the speaker. In case (c), an extrinsic frame of reference is established: It inherits its origin from the location of the computer but its orientation is determined by the motion of the computer respectively the mobile table underneath (the direction of motion is indicated by the arrow).

A further frequent distinction is made between allocentric and egocentric frames of reference. *Allocentric* frames of reference rely on a fixed coordinate system: Its direction and origin are imposed by external factors such as the compass points, and they are independent of the observer's or addressee's current position. Consequently, in an allocentric frame of reference, one can refer to objects in the environment from a survey perspective, e. g. "Go north across the lawn." (see, for example, [Tversky and Lee, 1998, Tversky and Lee, 1999, Werner et al., 1997]). In an *egocentric* frame of reference, the origin of the coordinate system is determined by the location of a human observer or addressee, and its orientation is established with respect to the intrinsic body axis. Therefore, egocentric frames of reference can be considered to be a special case of the intrinsic type [Klatzky, 1998]. However, due to their relevance in practical applications such as navigational assistance, it makes sense to define a distinct category. Verbal route directions, for example, often rely on an egocentric frame of reference. They encode the path to follow using landmarks (or routemarks) as well as spatial relations between objects in the current environment. The addressee is assumed to undertake a mental journey [Schmauks, 1998, Maaß and Schmauks, 1998]: objects in the environment are then localized in relation to her current position or to each other from an egocentric point of view. This view is also known as route perspective (cf. [Tversky, 1993]) or

field perspective (cf. [Schweizer et al., 1998]). In addition to these categories, a further distinction between static and dynamic frames of references can be made. The main criterion in this case is the fact whether or not the anchor object that was used to establish the current frame of reference is in motion relative to the world. We would like to introduce an additional differentiation, which takes into account how a frame of reference is established in the following section.

Induced frames of reference

So far, we have only considered 'fixed' frames of reference that are established directly. Instead of relying on the immediate establishment of a frame of reference, an alternative approach consists of using *meta-communicative acts* such as turn instructions to *induce* a frame of reference. The resulting induced frames of reference can then be defined as follows:

> An *induced frame of reference* is a frame of reference that is not established directly but rather requires the listener to first perform one or more mental or physical actions before the frame of reference is established. These actions include rotation and relocation, which may be applied to the origin and/or the orientation of an original frame of reference.

From this definition, several conclusions can be drawn: First of all, the actions inducing the frame of reference can be either absolute or relative. In the first case, no information on orientation or origin from an original frame of reference is required to perform the corresponding operation (see, for example, sentence (2)). In the later case, the action is relative to an original frame of reference (see sentence (1)), which implies that the corresponding induced frame of reference can only be established if the original one is known. A second (related) observation is that it is also possible to establish an induced frame of reference 'out of the void', e. g., when the inducing actions include absolute reorientation and relocation (see sentence (3)). It is therefore possible to compensate any kind of lack of information about the 'original' frame of reference (at least in theory), which can enable a system such as the one described in chapter 6 to address the real world problem of missing or inaccurate positional information (see 5.3.4).

> *If you turn a little bit to the right,* the castle is exactly behind the church. (1)

> *If you stood in front of the church,* the fountain would be to your right. (2)

> *Standing on the market facing the church,* the library is to your left. (3)

Finally, inducing a frame of reference may help to generate 'better' relational expressions such as localizations. Usually, the set of available frames of reference in a given situation consists of the ones defined by the listener and the speaker as well as those established by the target object and all potential anchor objects. Applying the orientation of either speaker or listener to any of those objects can yield further frames of reference, but this really only is a special case of inducing a frame of reference. Even if we include the latter ones, it is still possible that there is no combination of a frame of reference, a spatial relation, and an anchor object that yields a satisfying

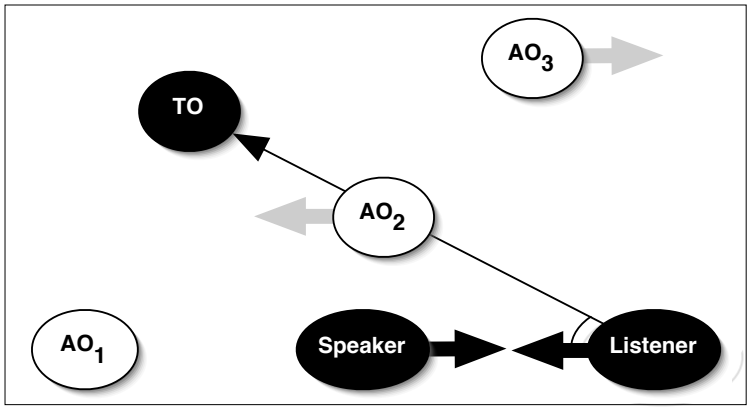


Figure 4.3: Using an induced frame of reference: an example

localization. In this case, the induction of a frame of reference can help to improve the resulting relational expression since the frame of reference used can be 'adapted' more precisely to the corresponding spatial relation.

To illustrate this, consider the following example (see figure 4.3): A speaker wants to describe the location of the target object TO to a listener who is facing her. Of all objects in their environment, only the anchor objects AO_1 through AO_3 are suitable candidates for use in an relational expression, e. g. because all other objects are hard to distinguish or unknown to either the listener or the speaker. However, neither frame of reference established by the potential anchor objects,[1] the speaker, or the listener yields a single spatial relation (such as `left-of`) that applies well to the given situation. While it is possible to introduce additional relations (such as `left-of-in-front`), not all languages provide means to easily verbalize those. However, in this situation the speaker can easily induce a frame of reference by giving a turning instructions such as "Turn towards AO_2.", which will result in very good applicability of the relation `behind`(AO_2, TO).[2]

The advantages of induced frames of references – namely improvement of relational expressions and compensation of lacking information on the 'current' frame of reference – come at a cost: On the one hand, the listener has to perform one or more mental or physical spatial operations before being able to decode the information based upon the induced frame of reference. Since, for example, mental rotations are among the most demanding operations in terms of cognitive resources, induced frames of reference can increase the 'cognitive load' of the user compared to direct establishment, and are therefore not suited when the user's cognitive resources are strained (e. g. while she is performing a secondary task).

On the other hand, the inclusion of induced frames of references in the reasoning process also entails a much higher computational load: When considering only direct establishment, the set of

[1] The arrows attached to the objects indicate the orientation of the corresponding frame of reference.

[2] In this example, we assume that the speaker just wants to communicate the location of the target object. Otherwise, the occlusion of TO by AO_2 may be a problem.

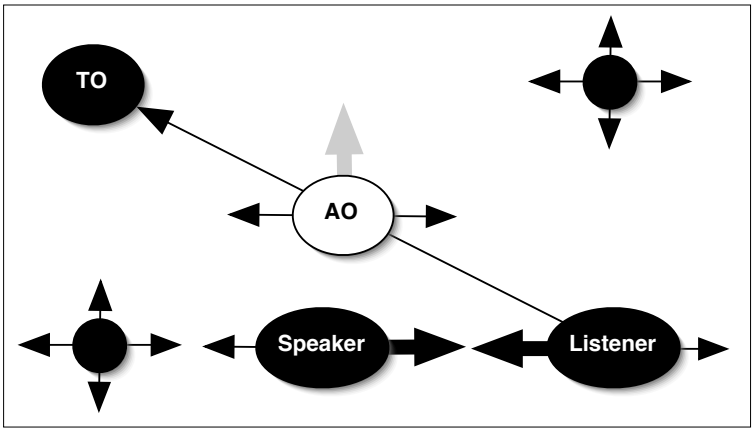


Figure 4.4: Improving the quality of relational expressions using an induced frame of reference

potential frames of reference is restricted to those defined by the listener, the speaker, and by all suitable anchor objects. Depending on the situation (e. g. localization in an urban environment), the resulting set may already consists of hundreds of candidates. If we take into account that some anchor objects do not have an instrinsic front, this number grows further since we can then apply the orientation of either the listener or the speaker to the corresponding objects.

Even if we do not count the resulting frames of reference as being induced – which we could as the orientation is imposed on a previously directionless frame of reference – the impact of induced frames of reference is still large: For every suitable anchor object (including the listener and the speaker), we have to consider several different orientations instead of a single one. In theory, we could generate an infinite number of candidates by applying every possible orientation. In practice, the number of potentially meaningful orientations is limited, e. g. by the listener's and the speaker's own frame of reference as well as by the target object. The spatial constellation of the target object, the listener and the speaker as well as the potential anchor object also induces some potentially meaningful orientations.

Figure 4.4 shows and example situation to illustrate these considerations: The intrinsic orientation of listener, speaker, and the anchor object (AO) are depicted using thick arrows, all of which are potentially useful orientations for an induced frame of reference. The two unlabeled circles indicate additional potentially useful origins for an induced frame of reference, which are implied by the constellation of the objects in the scene: these origins allow for certain angular relations (such as `left-of` or `in-front-of`) to apply perfectly. For example, if there is a good way to induce the origin depicted by the unlabeled circle in the upper right corner, then the speaker could describe the location of TO very precisely using the `in-front-of` relation. A corresponding linguistic realization is given in sentence (4).

If you stood at <`circle`>, TO is exactly in front of you. (4)

4.1.3 Spatial relations (two-point relations)

Spatial two-point relations are another very fundamental concept of spatial reasoning and interaction on spatial topics. Consequently, many approaches have been propose to computationally extract spatial relations from geometrical scenes, to use them in various reasoning calculi, or to interact with human users about spatial topics (see section 2.1.3). We have selected Gapp's model [Gapp, 1994] as a basis for our approach for two main reasons. On the one hand, it is less expensive in terms of computational resources than set-based approaches and it allows for various object representations (e. g. point, bounding box, outline) to be used. On the other hand, it relies on the same two basic parameters as the process for modeling n-point relations (path relations) does, which we will present in the next section. Furthermore, there is some empirical evidence [Gapp, 1995] that the model captures – at least partially – some human concepts of space.

Gapp models a large body of spatial relations using this approach, including projective (or angular) relations, distal relations (topological and distance-dependent relations), and even addresses the n-point relation `between`, which is a rather difficult case [Hanßmann, 1980, Habel, 1989]. He also shows how to use this mechanism to model combined or complex relations such as `in-front-right` or `far-left`, which in some languages (for example, German) can easily be mapped onto prepositions or spatial expressions. In some cases, these combined relations allow for a more precise description of a spatial constellation than 'cardinal' relations. The basic parameters used in the computations are the *distance* between the anchor and the target object, and the *angle* which the line connecting them sets with respect to the underlying frame of reference. In Gapp's model, this frame of reference is scaled by the axial extension of the anchor object in order to capture the effect of size and to determine a universal relative distance, and rotated to align the intrinsic front of the anchor object with the axes of the frame of reference. This process modifies the original angle and distance before a spline function is applied to the resulting values in order to compute a *degree of applicability* (DA) for the relation in question [Schirra, 1994]. Since the spline function maps the raw values to the closed interval from 0.0 to 1.0, the DA is normalized.

Figure 4.5 shows two example spline functions: The dark line corresponds to the relation `near` and is shaped to take into account that this relation applies very well when the distance between anchor and target object is small or zero, and does not apply at all when the distance is large. The light line models the relation `right-of` peaking at $\frac{1}{2}\Pi$, which corresponds to a 90 degree angle from the main axis. The advantage of this indirect approach lies in the flexibility of adaptation: spline functions can be fine-tuned to take almost any form, and therefore allow for the unified modeling of all different relations. Obviously, this flexibility also facilitates the incorporation of new empirical results. Gapp has run a small series of experiments on the understanding of spatial relations and was able to present a set of spline functions that can reproduce the results of the human participants [Gapp, 1995].

However, the model introduced by Gapp does not take into account any non-geometrical factors, and reduces distance effects to purely geometrical phenomena only accounted for by the scaling of the frame of reference. Furthermore, it requires an intrinsic orientation to perform the alignment of the anchor object with the world coordinate system. In addition, the computation of angular relations neglects the distance between anchor and target object.[3] However, common

[3] While the axial scaling according to the extension of the anchor object produces a new distance unit to measure the distance between anchor and target object, only the angle is take into account when determining the applicability of an

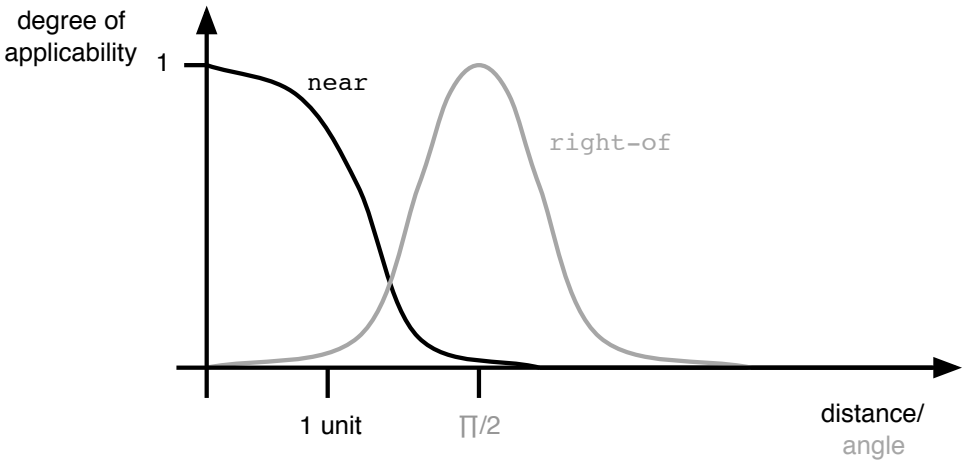


Figure 4.5: Using splines to model spatial relations (adapted from [Gapp, 1994])

sense and empirical evidence (e. g. from our study on path relations, which we will present in the following section) suggest that it makes a difference for the applicability of angular relations whether or not anchor and target object are close to one another. We have therefore selected a modified version[4] of Gapp's calculus to serve as a basis for our approach that mainly relies on the (untransformed) angle and distance between anchor and target object and the application of a spline function. Equation 4.17 shows a formal definition.

$$
\begin{aligned}
& da(rel, frame, anchor, target) \qquad (4.17)\\
&= spline(rel, angle(frame, anchor, target), dist(anchor, target))\\
&= \begin{cases} rel \in \Re_{dist}: & spline_{rel}(dist(anchor, target) \cdot scale^{\star}(\mathfrak{C}, \mathfrak{U})) \\ rel \in \Re_{angle}: & spline_{rel}(angle(frame, anchor, target)) \cdot \\ & spline_{near}(dist(anchor, target) \cdot scale^{\star}(\mathfrak{C}, \mathfrak{U})) \end{cases}
\end{aligned}
$$

The function *da* takes as input parameters a two-point relation *rel*, a frame of reference *frame*, an anchor and a target object (*anchor*, *target*) and returns a degree of applicability. It relies on the *spline* function that selects the spline $spline_{rel}$, which models the relation *rel*. If *rel* is a distance-dependent relation (i. e. $rel \in \Re_{dist}$) the euclidean distance between the anchor and the target object $dist(anchor, target)$ is scaled according to information on the current context $\mathfrak{C}$ and user $\mathfrak{U}$ before the application of the spline $spline_{rel}$ corresponding to *rel*. This scaling function is defined in detail in 4.1.5. If the relation is angular (i. e. $rel \in \Re_{angle}$) $spline_{rel}$ is applied to the angle $angle(frame, anchor, target)$ between anchor and target object within the current frame of reference. In order to account for the distance effect, the resulting value is multiplied by degree of applicability of the distal relation `near`.

angular relation.

[4] We left out the transformation of the coordinate system according to the size of the anchor object since we model scaling effects through the *scale* function presented in 4.1.5.

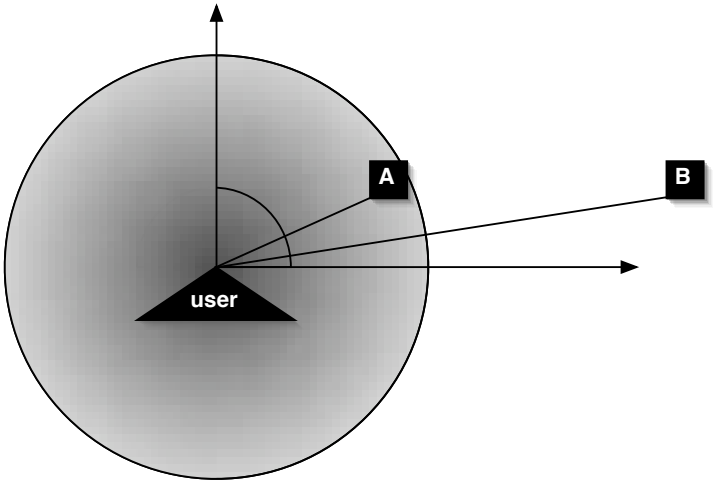


Figure 4.6: Distance effects in angular relations

Figure 4.6 provides an example for this approach. If we compute the applicability of the relation `right-of` in the depicted situation, the resulting value for object 'B' will be less than the one computed for 'A'. The current frame of reference has its origin in the location of the user, and its orientation is illustrated by the axes: the horizontal one is pointing straight to the right. Since $spline_{right-of}(0) = 1.0$ for objects on this axis, the closer an object lies to that axis (i. e. the smaller the angle formed at the origin of the frame of reference) the higher the value returned by $spline_{right-of}$. While $spline_{right-of}(frame, user, A)$ consequently is less than $spline_{right-of}(frame, user, B)$, $spline_{near}(dist(user, A))$ is greater than $spline_{near}(dist(user, B))$. (In the figure, the core area, where `near` is most applicable is depicted by a circle: the darker the color, the higher the applicability of the relation.) Hence, the applicability of *right-of* is higher for 'A' than for 'B' – assuming that the situation does not change, i. e. $scale^*$ remains constant.

4.1.4 Path relations (n-point relations)

Although path prepositions such as *along, across, past* are often part of path descriptions or navigational instructions, only limited effort has been put into investigating their properties and into modeling them (cf. [André et al., 1986, Krüger and Maaß, 1997, Raubal and Worboys, 1999, Kray and Blocher, 1999]). This is unfortunate as path prepositions offer some unique means to express, for example, route descriptions, and a single path relation can convey a lot of information [Tversky and Lee, 1998]. Consider, for example, a path following the shape of a river. We can generate a very precise and concise route instruction for such a trajectory using "along", while we would have to generate a sequence of instructions otherwise. Furthermore, path prepositions/relations relate to the *shape* of an object, whereas their distance-dependent, or angular counterparts do not. Neither do topological relations, as they are invariant to elastic deformations. Therefore, path prepositions can enrich route instructions by introducing shape, and they can also contribute to reducing the complexity of route instructions.

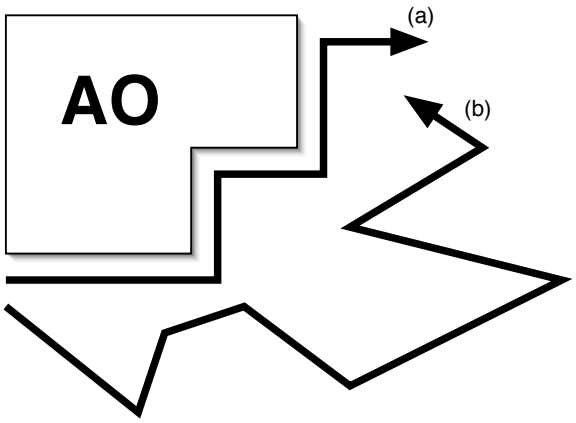


Figure 4.7: Path relations require more than two points (from [Kray and Blocher, 1999])

In [Kray and Blocher, 1999], we proposed a computational model for path relations, where we introduced the notion of *basic path relations* along the lines of Gapp's work on spatial relations [Gapp, 1994]. We defined six basic path relations, each modeling a change (or lack of change) of either distance or angle. As we were primarily concerned with identifying the basic meanings of path relations, we focussed on the analysis of simple straight lines. Subsequently, we extended the model to the more general case of arbitrarily shaped poly-lines. We also tried to identify fundamental concepts underlying some common (German) path prepositions, and map them onto basic path relations. Nevertheless, we pointed out that there is no one-to-one correspondence between path relations and prepositions, and that contextual factors need to be taken into account.

Following [Herskovits, 1986] we distinguish between the *basic meaning* of a spatial relation and its instantiation in a concrete situation. Since the computation of distance-dependent and angular relations relies two essential parameters – the angular disparity and the distance between anchor and target object – to evaluate the applicability two-point relations, our goal was to find a compatible approach for n-point relations. However, path relations differ from their topological and projective counterparts in two ways. Firstly, the target object is expected to be path-like: either it's shape has to be path-like, or it can be abstracted to a path-like shape. In some cases, this also holds for the target object. (In the computation of the applicability of two-point relations, the shape of the target object is of lesser importance.) Secondly, the computation of path relations cannot be reduced to a simple two point problem. Figure 4.7 illustrates this fact: Trajectory (a) is certainly a better match for a relation describing a path that follows the form of the target object (TO) than is trajectory (b). However, there is no single point on either trajectory that can be used to determine the applicability of this relation. Instead, we have to compare them according to their shape, i. e. several critical points.

Based on an analysis of several expressions commonly used to describe paths or similarly shaped objects (see [Kray and Blocher, 1999]), we can identify several *basic path relations* that form the basic building blocks for the more complex meaning of path prepositions. The following five simple path can be combined with each other, and/or with two-point relations in order to form

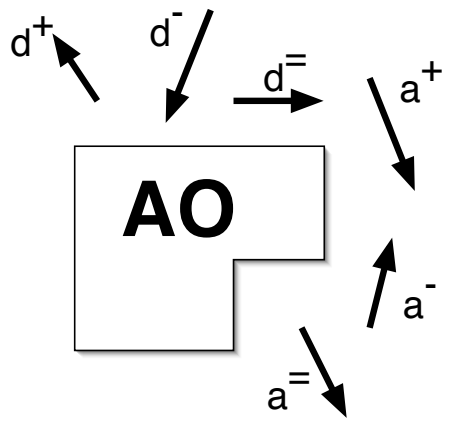


Figure 4.8: Basic path relations (from [Kray and Blocher, 1999])

higher order path relations and to capture the meaning of various expressions describing paths.

- `approach`(t,AO) the end point of the trajectory is located closer to the anchor object than is the start point.
- `depart`(t,AO) the start point of the trajectory is located closer to the anchor object than is the end point.
- `follow`(t,AO) the distance of every point on the trajectory from the anchor object is the same.
- `turn`(t,AO) the start and the end point form an angle with the anchor object. We can further distinguish `turn-cw` and `turn-ccw` if we want to encode the direction of the turn (clockwise, counter-clockwise).
- `no-turn`(t,AO) the start and the end point show no angular disparity in relation to the anchor object.

Since the direction of the angular disparity cannot easily be expressed using path prepositions (at least as far as German, French, and English are concerned), it makes sense to have just one relation expressing undirected change. This is not true in the case of distal change, where we consequently differentiate between an approach and an increase of distance. Figure 4.8 shows some prototypical examples for these basic path relations. The trajectories are labeled with either 'a' or 'd' for **a**ngular and **d**istal relations and with either plus, minus, or an equal sign for increase, decrease and no change. The trajectories a^+ and a^- are examples for `turn`.

Formally, we can compute the applicability of a basic path relations based on the angular or distal change between the start and end point of a two-point line. Equations 4.18 through 4.22 show the corresponding formulas, which are adapted from [Kray and Blocher, 1999]. The distal path relations `approach`, `depart` and `follow` mainly depend on the difference of distance that the

start and end point of the trajectory t have from the anchor object AO – scaled by the length of the trajectory. Analogously, the angular relations `turn` and `no-turn` are determined by computing the angular change from start to the end point of the trajectory. Equation 4.21 and 4.21 show the formulas for `turn-cw` (turn clockwise) and `turn-ccw` (turn counterclockwise), which can be used to differentiate the direction of angular change. The corresponding absolute value encodes the `turn` relation.

This computational approach to model n-point relations for two-point trajectories can easily be extended to n-point trajectories by determining the weighted sum over all subsegments consisting of two points. Equation 4.23 shows the corresponding function for `approach`: An n-point trajectory t^* is defined by n points p_i. Its degree of applicability for `approach` is obtained by computing the sum of the degrees of applicability of `approach` of all subsegments $\overline{p_i, p_{i+1}}$, which are weighted by their relative length $\frac{length(\overline{p_i,p_{i+1}})}{length(t)}$. Analogously, we can compute the other basic path relations.

$$\texttt{approach}(t, AO) = \frac{\Delta(dist(end(t), AO), dist(start(t), AO))}{length(t)} \tag{4.18}$$

$$\texttt{depart}(t, AO) = -\frac{\Delta(dist(end(t), AO), dist(start(t), AO))}{length(t)} \tag{4.19}$$

$$\texttt{follow}(t, AO) = 1 - \left|\frac{\Delta(dist(end(t), AO), dist(start(t), AO))}{length(t)}\right| \tag{4.20}$$

$$\texttt{turn}(t, AO) = \begin{cases} \texttt{turn}-\texttt{cw}(t, AO) = \frac{\Delta angle(end(t), AO, start(t))}{2\pi} \\ \texttt{turn}-\texttt{ccw}(t, AO) = -\frac{\Delta angle(end(t), AO, start(t))}{2\pi} \end{cases} \tag{4.21}$$

$$\texttt{no}-\texttt{turn}(t, AO) = 1 - \left|\frac{\Delta angle(end(t), AO, start(t))}{2\pi}\right| \tag{4.22}$$

$$\texttt{approach}(t^*, AO) = \sum_{i=0}^{n-1} \frac{length(\overline{p_i, p_{i+1}})}{length(t^*)} \cdot \frac{\Delta(dist(p_{i+1}, AO), dist(p_i), AO))}{length(t^*)} \tag{4.23}$$

$$\texttt{approach}'(t^*, AO) = \texttt{near}(t^*, AO) \cdot \texttt{approach}(t^*, AO) \tag{4.24}$$

While we were able to use these basic path relations to capture the meaning of several (German) phrases (prepositions, adverbs) expressing path-related concepts [Kray and Blocher, 1999], we also wanted to obtain some empirical data. Therefore, we conducted an empirical study (which we will present in the following section) on the human concept of path relations. One key result from this study was the impact of distance: the farther away a trajectory was from the anchor object, the smaller the applicability of a path relation became – even if it was, for example, perfectly parallel. In order to account for this observation, we included the degree of applicability for the `near` relation as a factor in the equations computing the applicability of path relations. Equation 4.24 shows the corresponding formula for `approach'`; the equations of the other n-point relations are modified analogously.

In the appendix A, we describe an empirical study that informed the design of this approach. The following section addresses a complimentary problem, namely the segmentation of complex paths into smaller parts.

4.1.5 Path segmentation

Path segmentation is another key process in the context of tasks related to trajectories, routes, and directions. Partitioning a complex trajectory into smaller chunks is also very relevant when generating path relations, since often the more complex the target trajectory is the harder it is to find a suitable anchor object and path relation. Consider, for example, a round trip through a city that includes several sights: in this case it is very unlikely to find a single anchor object and a corresponding path relation that apply well to the *entire* trajectory. However, if we subdivide the trajectory into smaller segments, we can usually find a combination of a street and `along` that has a high degree of applicability in an urban environment. Path segmentation is also a process, where the influence of situational factors such as the current means of transportation, weather and (possibly) traffic conditions, or the user's familiarity with her current environment is very obvious.

The representation and reasoning about distances are two concepts that are closely related to the subdivision of trajectories. Based on various observations both in cognitive psychology and in artificial intelligence, Berendt [Berendt, 1999] has proposed a computational model of inferred route distances, which was tested in a series of empirical studies. In her PhD thesis, Jansen-Osmann [Jansen-Osmann, 1998] has evaluated two prominent theories in distance cognition, the feature accumulation hypothesis [Allen, 1981] and the route segmentation hypothesis [Sadalla and Magel, 1980]. She has found both to apply in some cases, but in others either one of them applied or neither one. While it is certainly important to learn more about how humans perceive and represent distances, this only accounts for a part of all interactions on spatial topics between a human and a computer. Aside from the situational factors mentioned above and in previous sections, it is also very important how well the a path segment can be communicated. In this context, the subjective length of the segment can serve as a starting point but we need to consider further factors.

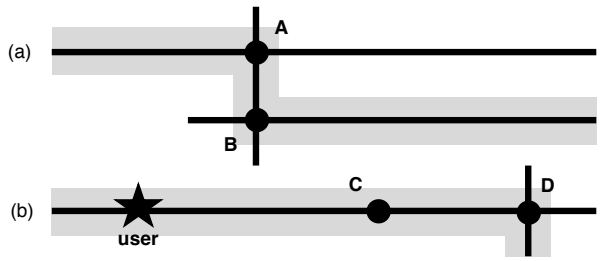


Figure 4.9: Topology, distances, and path segmentation

Among other factors, the topology of a route has a major impact on segmentation. For example, if we want to generate incremental route directions, we need to take into account decision points such as crossings. Figure 4.9(a) illustrates this factor: although the maximum base length for a segment might extend from the beginning at (a) over A to B, it is needs to change direction at that point. Furthermore, we have to assure that the user is instructed not only at decision points but also after a given time span – for instance, in order to assure her that she is still on the right way. Figure 4.9(b) shows an example for this case: Assuming that we want to provide the user with

instructions with a minimum frequency or with a maximum distance between two instructions, we may have to give instructions at point C. This is necessary if the user has traveled for a longer time than the time threshold or for a longer distance than the distance threshold, even though point C is located on a straight line well before the next decision point D.

In order to capture spatial, temporal, and situational factors in the segmentation process, we propose a two-level approach: We first compute a segmentation based on static assumptions about the user and the situation she is in. Dynamic changes are then accounted for by installing time and region triggers that can cause a re-segmentation. These triggers are described in detail in section 6.5. The segmentation function itself requires (in addition to the geometry of the route and the beginning of the actual segment) information about the maximum length and the maximum angular change of a segment. For example, if the threshold for the cumulated angular change (i. e. its curvature) of a trajectory is 90 degree, segmentation occurs at every perpendicular turn. In order to account for the situation, we compute these thresholds based on user- and context-related data. Equation 4.25 to 4.28 formalize this process.

$$max\_length(\mathfrak{C},\mathfrak{U}) = \begin{cases} means\_trp \in \mathfrak{C}: & max\_length'(means\_trp,\mathfrak{C},\mathfrak{U}) \\ means\_trp \notin \mathfrak{C}: & max\_length'(ped,\mathfrak{C},\mathfrak{U}) \end{cases} \tag{4.25}$$

$$max\_length'(m,\mathfrak{C},\mathfrak{U}) = \begin{cases} m \in \{ped, bike, ...\}: & length(m,\mathfrak{U}) \cdot scale(\mathfrak{C},\mathfrak{U}) \\ m \notin \{ped, bike, ...\}: & length(m) \cdot scale(\mathfrak{C},\mathfrak{U}) \end{cases} \tag{4.26}$$

$$length(m,\mathfrak{U}) = length(m) \cdot scale'(phys\_const \in \mathfrak{U}) \cdot scale'(age \in \mathfrak{U}) \tag{4.27}$$

$$scale(\mathfrak{C},\mathfrak{U}) = \prod_{i=0}^{n} scale'(c_i \in \mathfrak{C}) \cdot \prod_{j=0}^{m} scale'(u_j \in \mathfrak{U} \setminus \{phys\_const, age\}) \tag{4.28}$$

$$scale^{\star}(\mathfrak{C},\mathfrak{U}) = \prod_{i=0}^{n} scale'(c_i \in \mathfrak{C}) \cdot \prod_{j=0}^{m} scale'(u_j \in \mathfrak{U}) \tag{4.29}$$

A function *max_length* is defined to determine the maximum length of a segment, which it computes using contextual information contained in $\mathfrak{C}$ and user-related information contained in $\mathfrak{U}$. It first evaluates whether information about the current means of transportation (*means_trp*) is available. If the corresponding data is missing, it assumes a pedestrian user (*ped*). The maximum length is then derived by multiplying a base value *length*(*m*) for the means of transportation *m* with a scaling $scale(\mathfrak{C},\mathfrak{U})$ induced by the current context $\mathfrak{C}$ and the user-model $\mathfrak{U}$. If the current means of transportation requires a physical effort by the user – such as riding a bike (*bike*) or walking (*ped*) – the base *length*(*m*) is first scaled according to the physical constitution (*phys_const*) and the age (*age*) of the user to account for the varying speed and acceleration different users may attain. The general scaling function $scale^{\star}$ is computed by multiplying the scale factors $scale'(x)$ associated with each contextual information c_i in the current context $\mathfrak{C}$ with those associated with each user-related information u_j in the current user model $\mathfrak{U}$. The scaling function *scale* adapted for path segmentation takes into account that two user-related factors are already evaluated during the computation of *length*. The maximum angle can be computed analogously.

Before we can define the segmentation function itself, we have to give a more specific definition of what constitutes a path. Equation 4.30 shows a formal specification of a path *R* as a union of tuples (r_i, r_{i+1}). These tuples represent straight lines between nodes r_i and r_{i+1}.[5] Furthermore,

[5]The data used in real-world applications often contains small errors – e. g. from digitization – that result in very

we assume that the set $\Re$ is ordered in a way, that (r_0, r_1) represents the first segment of the path and (r_{n-1}, r_n) defines the last segment ending in the end point r_n of the path. The intermediate points and segments are sorted in the same way. The corresponding path $\Re$ hence consists of $n-1$ straight lines that are connected.

$$\Re = \bigcup_{i=0}^{n-1} \{(r_i, r_{i+1})\} \tag{4.30}$$

Based on the definition shown in equation 4.30, we can then specify the segmentation function as shown in equation 4.31. It takes as parameters a path $\Re$, the start point *loc* of the current segment, the maximum length *max_length*, and the maximum angular change *max_angle*. The result is a set representing the next segment, which consists of the tuples (r'_i, r'_{i+1}). While this set is not a true subset of $\Re$, it describes a line string that is a true substring of the entire path. The side conditions given for the nodes r'_i capture the recursive process that leads to the result set:

Beginning with the start point – which corresponds to r'_0 – we add a line to the following node in $\Re$ – and test whether the entire resulting segment still satisfies the requirement for maximum length and maximum angular change. If that is the case, we repeat the process with the previous end node of the segment. If one condition becomes false, we remove the last line added (r'_j, r'_{j+1}), and determine the point r''_j closest to r'_{j+1} that still satisfies the condition. If this point is not equal to r'_j, we add a line (r'_j, r''_j). Otherwise, we do not add a line. In both cases, we set the starting point of the next segment to r''_j, and we are done.

This last step is the reason why the segmentation process does not return true subsets of $\Re$: at the end of segment, the line between the last nodes of $\Re$ that we added to the segment may have to be split, and a new node is created. The following equation and the corresponding side conditions formalize this procedure.

$$segmentation(\Re, loc, max\_length, max\_angle) = \bigcup_{j=0}^{m-1} \{(r'_i, r'_{i+1})\} \tag{4.31}$$

where

$$\begin{aligned}
r'_0 &= loc \in \bigcup_{i=0}^{n} \{r_i\} \cup \bigcup_{i=0}^{n-1} \{line(r_i, r_{i+1}\} \\
\forall r'_{0<j<m}: \quad & r'_j \in \bigcup_{i=0}^{n-1} \{r_i\} \wedge \left((r'_j = r_k) \Rightarrow (\forall j' < j : r_{j'} \notin \bigcup_{i=k}^{n} \{r_i\}) \right) \\
r'_m &\in \{r_l | r'_{m-1} = r_{l-1}\} \cup line(r_{l-1}, r_l) \\
& \sum_{j=0}^{m-1} \left| (r'_i, r'_{i+1}) \right| \leq max_length \\
& \sum_{j=1}^{m-1} angle(r'_{i-1}, r'_i, r'_{i+1}) \leq max_angle
\end{aligned}$$

short segments that may form arbitrary angles with the neighboring segments. In order to apply the algorithm presented in the following, such data would either have to be cleaned, or the algorithm would have to be modified to ignore such distortions.

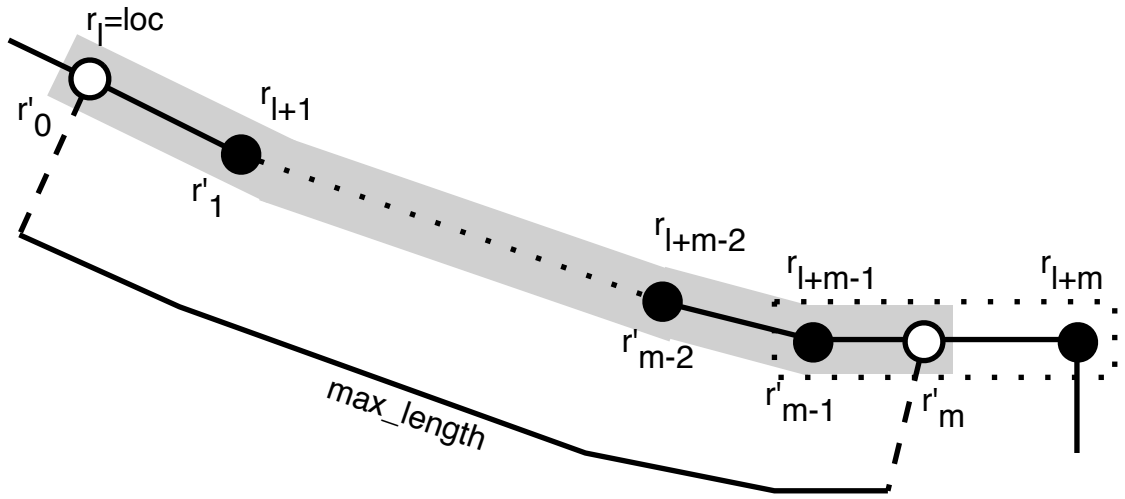


Figure 4.10: Graphical illustration of the segmentation algorithm

The side conditions of equation 4.31 specify the constraints for all nodes r'_j in the result set: r'_0 equals *loc* (the start of the current segment), which is either a node of $\Re$ or lies on a line defined by the tuples $(r_i, r_{i+1}) \in \Re$. (The function $line(r_i, r_{i+1})$ returns the set of all points located on the straight line connecting r_i and r_{i+1}.) For all other nodes r'_j in the result set (except the last one), the condition holds that they are equal to a node r_k of the tuples in $\Re$, and that previous nodes $r'_{j'}$ correspond to previous nodes of r_k. The last node r'_m is either the node $r_l \in \Re$, which immediately follows r_{l-1} that corresponds to r'_{n-1}, or lies on the line (r_{l-1}, r_l). Generally, the length of the segment may not grow beyond *max_length*, and the angular change has to stay below *max_angle*.

Figure 4.10 illustrates the segmentation process graphically. In the example shown, the start point r'_0 corresponds to a node $r_l \in \Re$, while the final node r'_m of the computed segment (thick gray line) does not. This is due to the fact, that adding the line (r_{l+m-1}, r_{l+m}) (which is highlighted by a dotted lined box) would result in a segment that is longer than *maximum_length*. The process can be extended to include further factors by introducing further side conditions. For example, if want to guarantee that segments do not include decision points (such as crossings), we can account for that by labeling nodes, where more than two edges meet, and introduce a side condition, that disallows edges that start with a decision point (except for the first edge).[6]

[6]We did not include a special treatment for decision points as this can also be done on a higher level: Since we used the segments as a basis for route instructions – which do not change while the user follows the segment – it is a straightforward task to repeat them at all decision points on the segment.

4.2 Language independent interaction

Although there are cases such as autonomous robot navigation, where a system that performs spatial reasoning tasks does not have to interact with a human user, there are far more, where such an interaction is required, e. g. navigational assistance, tourist information, or virtual reality systems. In order to separate language dependent processes (recognition, parsing, generation) from reasoning as much as possible, we require a language independent representation of the user's input to the system and of its output. In the following sections, we first analyze the requirements that such a representation has to address (see section 4.2.1). We then present the representation format that we developed for interaction on spatial topics (see section 4.2.2).

4.2.1 Requirements

In Section 4.3, we will present several complex tasks that typically arise in the context of spatial reasoning. Obviously, a suitable representation format must provide means to encode the user's requests that trigger these tasks as well as the corresponding replies. A single representation for both input and output not only simplifies the design of the system, but also greatly facilitates reasoning about the dialog as well as the realization of an interaction history. Furthermore, the format should enable a component that transforms the result of spatial reasoning processes to a specific target language (i. e. a generator) to vary its output. This is important for two reasons: On the one hand, a system that always generates exactly the same, monotonic replies, e. g. using the same syntactic structure, will be perceived by its users as being artificial and (possibly) boring. On the other hand, the current situation may impose constraints such as restricted display space or limited time for audio output that can be addressed better if the corresponding generator has some flexibility. For example, the production of an anaphora can help to shorten the output, but it requires a reference to the corresponding object instead of a simple lexical item. Furthermore, the representation should allow for various target languages without sacrificing more than necessary of the expressional power of each language. We may also want to include graphical languages, since it does not only make sense in the context of mobile systems such as PDAs to support the generation of graphical output (e. g. to account for the small screen size). In the WIP system [André and Rist, 1995], for example, a user can select whether she prefers graphical, verbal, or multimodal output, which enhances the user's satisfaction with the generated presentation. In order to achieve this flexibility, the WIP system relies on a subset of the speech act theory [Austin, 1962, Searle, 1969], which we will shortly introduce in the following section.

4.2.2 Preverbal messages

Prior to Austin's influential work [Austin, 1962] it was a wide spread approach to assume that people's utterances correspond to propositions that were either true or false. Austin broke with this traditional view by suggesting that a human speaker does not only utter a phrase but does perform a *speech act* that is very similar to a physical act. For example, by saying "I promise that I will finish my thesis before Eastern." a human performs the act of making a promise as opposed to simply stating that he will finish his thesis before Eastern (which may be either true of false). Searle [Searle, 1969] modified and further elaborated the theory by establishing a classification of

four distinctive categories:

- **utterance act**
  This most basic form encompasses the uttering of sentences (morphemes, words). It is important to note that an utterance act lacks a definite meaning. Consequently, performing such an act (without an propositional act) would result in uttering words without transmitting any coherent meaning.

- **propositional act**
  The basic meaning of a speech act is captured in the propositional act and consists of a *reference act*, which establishes a relationship to extra-lingual entities such as real-world objects, and the *act of predication.*, which encodes predicates such as properties and relations.

- **illocutionary act**
  While the utterance and the propositional act convey the form and the content of a speech act, its function in an ongoing communication is encoded by the the *illocutionary act.* For example, a complete speech act can function as a request, a threat, a promise, or a statement – to list only a few possibilities.

- **perlocutionary act**
  This category encompasses the consequences and effects of an illocutionary act on the listener. This concept includes, for example, that the listener may be frightened (by a threat), or convinced (by a statement).

Although the theory of speech acts has been designed with a focus on written or spoken (natural) language, it has been successfully applied to pictorial communication as well.[7] Additionally, recent studies (see [Tversky and Lee, 1999] for an example on route instructions) have shown a strong correspondence between pictorial and verbal 'speech' acts. This observation has been used to apply speech act theory in interactive systems such as WIP [André and Rist, 1995], which can generate presentations in various languages and media from a single representation. Therefore, a subset of the speech act theory – along with Grice's maxim of cooperation [Grice, 1975] – is a valid base to built upon when addressing the requirements listed in 4.2.1.

Another important work in the field of computational generation and comprehension of natural language was Levelt's book on speaking [Levelt, 1989]. In his book, he introduced a 'blueprint' of a speaker, consisting of several components: The conceptualizer conceives of an intention to speak, selects relevant information, and keeps track of the interaction. The formulator transforms the output of the conceptualizer into a phonetic plan by means of grammatical and phonological encoding. This plan enables the articulator to produce overt speech, which in turn can serve as input to the audition component that translates it into a phonetic string. The speech-comprehension system transforms this string into parsed speech, which serves as input to the conceptualizer. Levelt calls the output of the conceptualizer a *preverbal message (PVM)*, which he defines as a "semantic representation" [Levelt, 1989, p.73] of what is to be said. This concept of a language independent description has been widely used and specifically, among other fields, in the context of topics related to space (see, for example, [Klabunde et al., 1999, Guhe et al., 2000,

[7] see [André, 1995] for a short review

Porzel et al., 2002]). While Levelt refers to preverbal messages as being a composition of propositions – in the sense of Searle's propositional act – Guhe et al. [Guhe et al., 2000] have loosened this notion by allowing "sequences of well-formed propositional structures on a sub-propositional level", which include valid parts of propositions, e. g. predicate symbols or terms.

In order to meet the requirements introduced in 4.2.1, we have to enhance the basic preverbal message in two ways: Firstly, we want to encode all interactions that arise in the complex tasks described in section 4.3. This does not only include spatial concepts (or propositions) but also different illocutionary speech acts, e. g. requests and statements. Since we do not require very many illocutionary acts in a human-computer-interaction such as a navigational assistant,[8], we can restrict the corresponding set to a very limited number.

Secondly, a PVM that solely consists of propositions (or partial propositions) does not provide a generation component with much flexibility. While it is possible to select different words or syntactical constructs, or to generate anaphoric expressions, leaving out entire propositions, or choosing among different alternative propositions is not feasible. These restrictions make sense if one thinks of a PVM as encoding only the exact content that has to be verbalized.

However, if one takes into account that some (spatial) concepts vary in terms of how easily they can be realized in a given target language, the inclusion of alternatives into a PVM becomes desirable. Consider, for example, spatial deixis in English: There are only two categories ("here" and "there") whereas in Japanese, there are three categories ("koko", "soko", and "asoko"). While in this case, it is certainly possible to find a suitable realization for either categorization in either language, it does demonstrate the difficulty to find suitable semantic concepts [Fillmore, 1982, Yoshida, 2001]. One way to address this problem lies in including redundant information. In our example this could mean to include a metrical distance as well. This approach does also facilitate the generation of pictorial output.

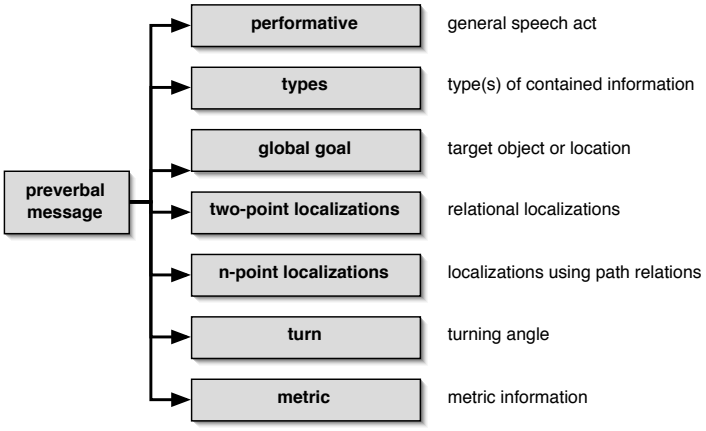


Figure 4.11: Structure of a preverbal message.

[8]We assume a cooperative and sincere scenario in the sense of Grice's maxim of cooperation as our focus is on assistive systems. These should not intentionally lie to the user or threaten her, which – if they did – would require further illocutionary acts.

In order to further support the generating component in the selection among alternative propositional descriptions, we propose to introduce graded propositions. Since we are mainly concerned with spatial concepts or relations, the degree of applicability presented in 2.1.3 is well suited for this task. Including this rating in a PVM enables a generator to evaluate alternative propositions in terms of their spatial and situational fitness and thus, to weight potentially better realizations in a given target language (or medium) against the possible loss of precision. This approach is especially advantageous in the context of route instructions as we will demonstrate later on in this section (see also 4.2.3). The following equations give a more precise definition of what is contained in a PVM as well as a suitable notation (see figure 4.11 for an overview). This abstract definition on a a conceptual level complements the corresponding definition on the implementation level, which we present in detail in section 6.4.2.

$$pvm = (perf, \mathfrak{T}_{pvm}, goal, \mathfrak{R}_{pvm}, \mathfrak{C}_{pvm}, angle, metric) \tag{4.32}$$

where

$$\begin{aligned} perf &\in \{\text{inform, request}\} \\ \mathfrak{T}_{pvm} &\subset \{\text{identification, description, localization, directions, choice,} \\ &\quad \text{agreement, disagreement, indetermination}\} \end{aligned} \tag{4.33}$$

$$\begin{aligned} goal &= (\mathfrak{O}_{goal}, name, id) \in \mathfrak{W} \\ \mathfrak{R}_{pvm} &= \{rel_1, ..., rel_n\} \\ &\quad \text{where } rel_i = \{\text{start, end, path, goal}\} \times \mathfrak{R} \times \mathfrak{W} \times ([0;1] \cup \{\text{undefined}\}) \\ \mathfrak{C}_{pvm} &\subset \mathfrak{W} \\ angle &\in [0;360] \cup \{\text{undefined}\} \\ metric &\in [0;+\infty[\cup \{\text{undefined}\} \end{aligned}$$

A preverbal message *pvm* is a seven tuple consisting of a performative $perf$, a set of preverbal message types $\mathfrak{T}_{pvm}$, a target object or location *goal*, a set of relations $\mathfrak{R}_{pvm}$, a set of choices $\mathfrak{C}_{pvm}$, an angle *angle*, and a metric distance *metric*. Of the seven elements of the tupel, only $perf$ and $\mathfrak{T}_{pvm}$ have to be present. All other entries are optional. Since our focus is on situated interaction on spatial topics between a human user and an artificial system, it is sufficient to distinguish two basic performatives (or illocutionary acts): inform, which marks statements, and request, which signals a question. The main types of the PVM are given in a set $\mathfrak{T}_{pvm}$ that contains one or more of the types listed in 4.33, which correspond to the complex tasks presented in the following section. Additionally, there is a type 'choice' to encode requests for choosing among a number of alternatives.

The *goal* entry represents the overall target object or location of the speech act encoded in a preverbal message. It is a triple that is an element of the set of world objects $\mathfrak{W}$, and consists of a set $\mathfrak{O}_{goal}$ of object types, a name, and a unique identifier *id*.[9] The set $\mathfrak{R}_{pvm}$ of two-point and n-point relations rel_i encodes the main propositional content, where each relation rel_i is defined by a quadruple. The first entry defines the target object of the relation, which is either *goal*, or a trajectory. In the latter case, either the beginning (*start*), the end (*end*), or the entire trajectory

[9] We assume that $\mathfrak{W}$ does not only contain a complete description for each world object but also partial entries as well as a triple $(\emptyset, \text{undefined}, \text{undefined})$: $\mathfrak{W} = \mathfrak{O}^n \times (\mathfrak{N} \cup \{\text{undefined}\}) \times (\mathbb{N} \cup \{\text{undefined}\})$.

(*path*) can be selected. The second entry is an element of the set of relation names $\mathfrak{R}$, which contains the names of all two-point and n-point relations. The corresponding anchor object is the third entry, an element of the set of world objects $\mathfrak{W}$. In addition to this, the seven tuple of a PVM also contains a set of choices $\mathfrak{C}_{pvm}$, a subset of $\mathfrak{W}$, as well as an angle *angle* and a metric distance *metric*. In the following we will point out how we encode all speech acts arising in the context of situated interaction on spatial topics using this simple structure.

$$\begin{aligned} pvm^{Q}_{ide} &= (\text{request}, \{\text{identification}\}, target', \emptyset, \emptyset, \text{undefined}, \text{undefined}) \\ pvm_{ide} &= (\text{inform}, \{\text{identification}\}, target, \emptyset, \emptyset, \text{undefined}, \text{undefined}) \end{aligned} \tag{4.34}$$

Equation 4.34 lists the entries corresponding to an request for object identification (see 4.3.1) and the resulting reply. The query pvm^{Q}_{ide} is a 'request' of the type 'identification', and any information of the object to be identified is encoded in $target'$. Consequently, $target' \in \mathfrak{W}$ is either $(\emptyset, \text{undefined}, \text{undefined})$ or partially defined, e. g. only an object type or a name is included. The preverbal message encoding the reply pvm_{ide} has an 'inform' performative and is again of the type 'identification', but now the goal object *target* is filled with all information, which can then be translated into an appropriate presentation.

$$\begin{aligned} pvm^{Q}_{loc} &= (\text{request}, \{\text{localization}\}, target', \emptyset, \emptyset, \text{undefined}, \text{undefined}) \\ pvm_{loc} &= (\text{inform}, \{\text{localization}\}, target, \mathfrak{R}_{loc}, \emptyset, angle_{loc}, distance_{loc}) \end{aligned} \tag{4.35}$$

In equation 4.35, the preverbal message encoding a request for localization (see 4.3.2) as well as the corresponding reply are shown. The request pvm^{Q}_{loc} is very similar to the one used in the object identification case, only the preverbal message type is set to 'localization'. The reply pvm_{loc}, however, does not only contain the target object *target* but also a spatial relation describing the location of *target* (in $\mathfrak{R}_{loc}$). In addition, it may also contain an angle $angle_{loc}$ that describes a necessary turn instruction in case an induced frame of reference (see 4.1.2) is used. Furthermore, there is a metrical distance $distance_{loc}$, which encodes the distance from the listener to the target object.

$$pvm^{Q}_{dir} = (\text{request}, \{\text{directions}\}, target', \emptyset, \emptyset, \text{undefined}, \text{undefined}) \tag{4.36}$$

$$pvm_{dir} = (\text{inform}, \{\text{directions}\}, target, \mathfrak{R}_{dir}, \emptyset, angle_{dir}, metric_{dir}) \tag{4.37}$$

where

$$\begin{aligned} \mathfrak{R}_{dir} = \{&(\text{start}, r_{start}, anchor_{start}, da_{start}), \\ &(\text{end}, r_{end}, anchor_{end}, da_{end}), \\ &(\text{path}, r^{two-point}_{path}, anchor^{two-point}_{path}, da^{two-point}_{path}), \\ &(\text{path}, r^{n-point}_{path}, anchor^{n-point}_{path}, da^{n-point}_{path})\} \end{aligned} \tag{4.38}$$

The preverbal messages shown in equation 4.36 encode the request for guidance (see 4.3.3). The query pvm^{Q}_{dir} again corresponds to the ones defined for object identification and localization except for the type, which is 'directions' in this case. The reply pvm_{dir}, however, does contain much more than in both previous cases. Aside from the performative 'inform', the type 'directions', and the overall target of the guidance *target*, there is a set of spatial relations $\mathfrak{R}_{dir}$, which

describes the trajectory of the current segment of the route. In addition, a turn angle $angle_{dir}$ is included that describes a turn the listener has to perform at the beginning of the segment. Finally, the length of the segment is encoded in $metric_{dir}$. The set of spatial relations contains at most four elements: three two-point relations (describing the start and the end of the route segment as well as of the entire trajectory) and a single n-point relation (capturing the shape of the trajectory). Most frequently, a route is too long and too complex to adequately describe it using a single instruction. Therefore, we assume that the reply to a request for guidance is usually answered by a sequence of directions $\{pvm^0_{dir}, \ldots pvm^n_{dir}\}$ that individually describe subsequent segments of the route.

Obviously, not all the information contained in pvm_{dir} is required in order to produce a presentation that guides the user to her desired location. However, there is a key benefit in including more than the purely necessary information in a preverbal message: The component that generates the presentation can select, which parts of the PVM it wants to realize according to the current situation. For example, if the user is very unfamiliar with the environment, all information can be used. Or, if the presentation can only use audio (e. g. when the user cannot look at a display at the moment), the system can verbalize solely the most relevant information such as the path relation and the turn instruction. Generally, the selection process can be guided by situational factors and also by the degrees of applicability, which enable the generating component to evaluate the quality of the corresponding relations compared to the other ones. This flexibility can also help to address resource restrictions, which we will discuss in section 5.2.

In addition to giving route direction, we will present further complex tasks such as the geo-encoding of spatial relations (4.3.4), data collection (4.3.5), and map interaction (4.3.6). Although they differ substantially from other tasks, we can encode the underlying interaction using the preverbal messages for identification (see equation 4.34) and localization (see equation 4.35). In order to account for different complex tasks, we can include a further preverbal message type (in addition to 'identification' respectively 'localization') that indicates the task. In section 6.4.2 the corresponding types are listed, and several concrete examples from the prototypical implementation are given.

$$pvm^Q_{pos} = (\text{request}, \{\text{localization}\}, user, \emptyset, \emptyset, \text{undefined}, \text{undefined}) \tag{4.39}$$

$$pvm_{pos} = (\text{inform}, \{\text{localization}\}, user, \Re_{pos}, \emptyset, angle_{pos}, metric_{dir})$$

$$\begin{aligned} pvm^Q_{street} = {} & (\text{request}, \{\text{localization}\}, user, \\ & \{(\text{goal}, \text{on}, (\{\text{street}\}, \text{undefined}, \text{undefined}), \text{undefined}\}, \\ & \emptyset, \text{undefined}, \text{undefined}) \end{aligned} \tag{4.40}$$

$$\begin{aligned} pvm_{street} = {} & (\text{inform}, \{\text{localization}\}, user, \\ & \{(\text{goal}, \text{on}, (\{\text{street}\}, streetname, id), 1.0)\}, \emptyset, 0.0m) \end{aligned}$$

The interactions in the context of the complex task described in section 4.3 are not the only ones that we can encode using preverbal messages. In section 5.3 we will introduce several strategies to determine the user's current position – some of these include interactions, which we can also express using preverbal messages. Equation 4.39 lists the two basic PVMs that encode the request (pvm^Q_{pos}) and the final reply (pvm^Q_{pos}). Note that these are equivalent to the ones shown in equation 4.35 except for the goal, which – in this case – consists of the user. Since the process of position determination may require further interactions, these have to be encoded them as well.

Equation 4.40 lists the resulting preverbal messages for the steps required to determine the name of the street the user is currently in. They are both basically the same as in the generic localization case shown in equation 4.35: the query pvm^{Q}_{street} explicitly asks for a target object of type 'street' without a name, which is then provided by the reply pvm_{street}.

$$\begin{aligned} pvm^{Q}_{visibility} &= (\text{request}, \{\text{identification, choice}\}, \text{undefined}, \\ &\quad \emptyset, \mathit{sights}, \text{undefined}, \text{undefined}) \end{aligned} \tag{4.41}$$

$$\begin{aligned} pvm_{visibility} &= (\text{inform}, \{\text{identification, choice}\}, \text{undefined}, \\ &\quad \emptyset, \mathit{sights}', \text{undefined}, \text{undefined}) \end{aligned}$$

$$pvm_{yes} = (\text{inform}, \{\text{agreement}\}, \text{undefined}, \emptyset, \emptyset, \text{undefined}, \text{undefined}) \tag{4.42}$$

$$pvm_{no} = (\text{inform}, \{\text{disagreement}\}, \text{undefined}, \emptyset, \emptyset, \text{undefined}, \text{undefined})$$

$$pvm_{indeterminate} = (\text{inform}, \{\text{indetermination}\}, \text{undefined}, \emptyset, \emptyset, \text{undefined}, \text{undefined})$$

Further interactions in the context of determining the user's position concern the visibility of objects in her environment: the system asks the use a series of question such as "Can you see X, Y, or Z?", and the user answers with a corresponding reply (see section 5.3). The visibility request is encoded in a preverbal message $pvm^{Q}_{visibility}$, which is of type 'identification' and 'choice', the target objects are included in the set of choices (*choices*). The straightforward reply to such a query names those sights that are visible, e. g. "I can see Y and Z.", and would be encoded in a PVM such as $pvm_{visibility}$, which is very similar to $pvm^{Q}_{visibility}$ except for the performative and the set of choices: Since only some of the sights in *choice* may be visible, *choice′* is a subset of *choice*. If all sights are visible, the user can reply by a simple "Yes.", or if no sights are visible, by a simple "No.". These replies are represented by pvm_{yes} and pvm_{no}. Finally, it is possible that the user is unable to determine whether any of the sights are visible. In this case, she may say so, for example, by uttering "I don't know.", which is encoded in $pvm_{indeterminate}$.

$$\begin{aligned} pvm_{example} &= (\text{inform}, \{\text{localization}\}, (\{\text{gate, building}\}, \text{“Karlstor”}, 7), \\ &\quad \{(\text{goal}, \text{next-to}, (\{\text{station, building}\}, \text{“Karlstor station”}, 8), 0.7)\}, \\ &\quad \emptyset, \text{undefined}, 47.0m) \end{aligned} \tag{4.43}$$

In order to illustrate the flexibility of preverbal messages in terms of language independence, consider the following short example: Equation 4.43 shows a preverbal message encoding a localization for the "Karlstor", a historic gate at the edge of the old town of Heidelberg. It is localized using the two-point relation "next-to" with the anchor object "Karlstor station". In addition, a metric distance is included.

The **Karlstor** is `close to` the *Karlstor station*. (5)

It is `near` *Karlstor station*. (6)

The **red ark** is `near` *Karlstor station*. (7)

The **Karlstor** is about 50 m from the *Karlstor station*. (8)

From this simple preverbal message, a great variety of verbalizations can be generated such as the ones listed sentence (5) to (8). In order to highlight the correspondence between different parts of the preverbal message and the actual realization in a sentence, the target object (the Karlstor) is displayed in bold typeface, the anchor object (the Karlstor station) in italics, the relation (next-to) in typewriter font, and the metric distance (47.0 m) is underlined. Note that Karlstor is assigned the object types 'gate' and 'building', while the Karlstorstation is a 'station' and a 'building'. Information such as the color used in sentence (7) has to be retrieved from a database, e. g. using the object identifier, which is '7' in case of the Karlstor. But verbal realizations are only one possible means to present the information encoded in a PVM to the user. The following section will explore further options.

While the preverbal message proved to be highly versatile and useful in the context of modeling complex tasks related to space as well as in our prototypical implementation, its purpose is limited to situated interaction on spatial topics and a strict division of functions. For example, we did not specifically allow for underspecification and its on-demand resolution, i. e. backtracking [Wahlster, 2000]. Thus, the preverbal message mainly serves as a vehicle to encode relevant information related to space between various processes, and it is definitely not a general and full-fledged semantical or pragmatic representation format.

4.2.3 Presentations

Textual and spoken presentations – such as presented in the previous section – are only one way to generate an output from a preverbal message. Within Deep Map (see chapter 6.3.1) we implemented several components that can create various uni- and multi-modal/-medial presentations. For example, an alternate way to present route instructions is is familiar to users of commercial car navigation systems: it consists of a (mostly qualitative) 2D route sketch such as the one shown in figure 4.12. In its most abstract form, only an arrow pointing in the intended direction of motion is shown. The PVM can then be used to annotate this with additional information. Since key points of a segment are localized using qualitative spatial relations, it is a straightforward task to add the corresponding annotations to the basic arrow. Using the path relation included in the PVM, a label for the arrow may be generated. As for most other presentation means, the selection of which components of a PVM should be realized can be guided by the degree of applicability of the corresponding relation as well as by situational factors. The same type of presentation can, in principle, also be used to output localizations: In that case, the two-point relation determined the annotation of the arrow, which points from the anchor object to the target object. If a turn angle is part of the PVM, the direction of the arrow can be adjusted accordingly. More sophisticated route sketches can include information about the (qualitative) shape of the route (see, for example, [Tversky and Lee, 1999]).

Another very common way to present route instructions to a user consists of a two-dimensional geographic map. In this case, the objects contained in the PVM can inform the process of selecting which area to depict, and at the same time, these objects provide information about what to highlight on a map. Therefore, the PVM helps to address two key problems in automatic map generation: Since a map should contain enough information to enable its user to perform the task

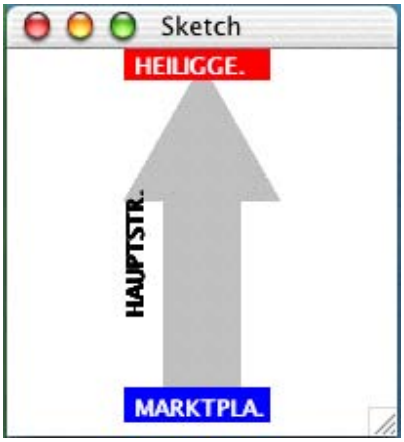


Figure 4.12: A route sketch generated from a PVM (screenshot).

at hand but should also avoid including unnecessary content, the generation process can rely on the objects contained in the PVM (as these are precisely the ones that the system wants to communicate to the user). A second common problem in map generation consists of selecting a proper zoom factor and what region to depict – both of these are especially important in the context of mobile devices, where the screen size is severely limited. The objects in the PVM can help to address this issue since a bounding box that encompasses them can serve as a starting point for the selection process. Figure 4.13 shows an example map that was generated by the system we will present in chapter 6. If the system knows the current orientation of the user precisely and if the PVM includes a turn angle, such a map can also be rotated to be aligned to the real world. In that case the user does not have to perform any mental (or physical) rotation, which is a very demanding cognitive process.[10] If the user's current position is included in such a map, it can be classified as a personalized you-are-here map [Richter, 2001].

There is an additional type of presentation, which is also well suited to transmit route instructions, and which shares some properties with two-dimensional maps as well as route sketches. *Schematic maps* [Casakin et al., 2000] (or *aspect maps*) [Berendt et al., 1998] are more abstract than purely geographical maps as they focus on those aspects that are most relevant for a specific purpose, e. g. robot navigation [Freksa et al., 2000]. Stronger distortions, for example, in terms of the angles and distances between objects, are accepted in order to highlight qualitative properties, for example, branching nodes and topology. Although we did not implement a corresponding generator, the information encoded in a PVM facilitates its presentation in the form of a schematic map. Not only does a PVM contain a number of relevant objects, which can serve as a starting point for the computation of a schematic map, but also several qualitative relations between those, which can inform the process of deciding what connections to depict in the resulting map. Due to their higher degree of abstraction, schematic maps are well suited for small displays.

[10] Although such a rotation can often be beneficial, there may be situations, where users will prefer the alignment to the cardinal directions – such as when directional information is not updated quickly and the map is 'lagging' behind the user's motion.

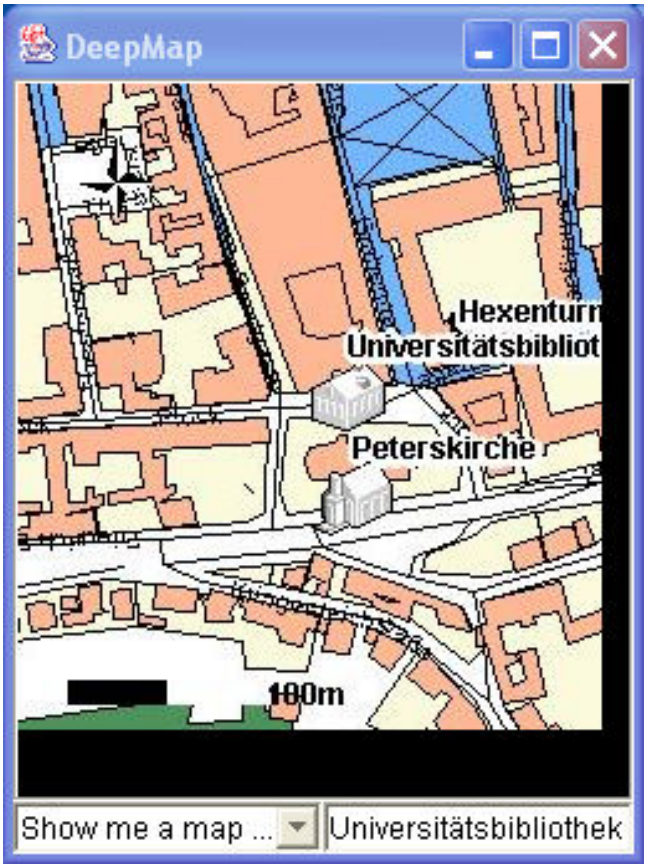


Figure 4.13: A two-dimensional map generated using a PVM.

A further means to present the content encoded in a preverbal message to the user is a natural extension to two-dimensional maps: Three-dimensional maps, i. e. pseudo-realistic renderings, have become a feasible alternative [Baus et al., 2002] – even on mobile devices [Kray et al., 2003]. In addition, there is some evidence that this kind of presentation has some distinct advantages, e. g. for landmark recognition [Laakso, 2002]. Figure 4.14 shows an example sequence for an animated three-dimensional route instruction: The animation shows the way the user should follow by means of a flight-through from its start point to its end point. Landmarks are visualized in detail with textured models to attract the users attention. Less relevant buildings are rendered in gray and in a semi-transparent way, thereby helping the user to focus on objects that will help her to find her way.[11] A preverbal message contains all the information that is needed to generate static or animated three-dimensional presentations. For example, the PVM for a route description contains the anchor objects for all key points on the segment as well as for a path relations. These enable the corresponding generator to select those objects that should be displayed in full detail in order to maximize the usefulness of the resulting presentation.

[11] This approach is also used to guide the user's focus in static three-dimensional presentations (see [Krüger, 2000]).

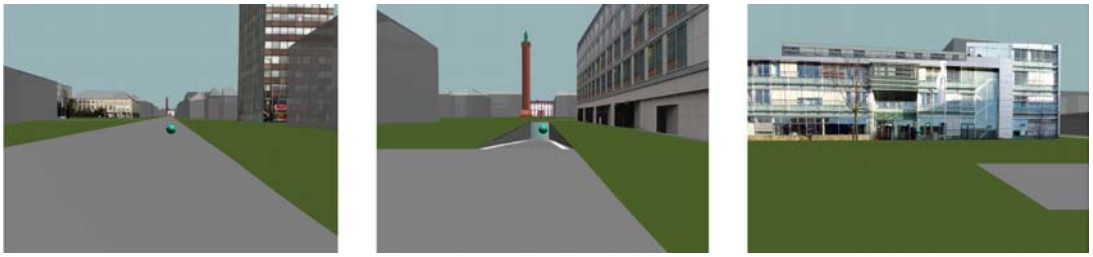


Figure 4.14: An animated three-dimensional route instruction (from [Kray et al., 2003]).

The different types of presentations reviewed in the previous paragraphs can also be combined. However, not all combinations are equally beneficial. Furthermore, it is a fundamental problem of presentation planning to compose multimodal presentations that are coherent across media and modalities, and that can be decoded easily by the user [Maybury and Wahlster, 1998]. This is especially true in the case of presentations, in which different modalities include coreferences to the same world objects (e.g. buildings in route instructions) [André and Rist, 1995]. A key concern in this context is to assure that the user can match references to the same object in different modalities, which usually requires interaction between different generators and/or iterative planning [André, 1995].

Despite these considerations, there are some key advantages in combining different media and modalities. Firstly, we can introduce redundancy by realizing the same part of a PVM in different media and modalities, which may help a user to better understand the main content of the presentation. Secondly, we can compensate weaknesses of certain media and modalities by combining it with another one. For example, while spoken route instructions do not require the visual attention of the user, they are not persistent. If the user wants to check later on, whether she remembers it correctly, an accompanying map and/or textual instruction compensates for the lack of persistence. Thirdly, a "rich" presentation may be aesthetically more pleasing, and therefore, more enjoyable to the user [Kray et al., 2003]. Finally, a combination of different media and modalities may help to overcome resource limitations such as small screen size by using additional channels – such as audio output – to transmit information that would otherwise require screen estate.

In the prototypical system (see chapter 6), which realizes our model, we employed several combinations in order to address various issues. For example, most spoken output is complemented by a textual presentation to address the problem of non-persistence of audio output and the limited quality of speech synthesis on mobile devices. The presentation of a complete route description is a further example for the benefits of combining several media and modalities. In a mobile setting, a key issue of this task consists of the small screen size and the difficulty to display a complex route appropriately. In the prototypical implementation, we addressed this problem by generating a slideshow, which is repeated automatically and consists of instructions for subsequent segments. The latter ones are composed of a two-dimensional map as well as a textual instruction for the corresponding segment. In this case, only the segment and the anchor objects contained in the PVM are displayed on the map, while the textual instruction most frequently contains a turn instruction, metrical information and a path preposition.

4.3 Complex tasks related to space

In the previous two sections, we presented modeling approaches for various basic processes related to spatial reasoning, and we introduced a generic encoding scheme for interactions on spatial topics. When we identified these basic processes in section 2.5.2 we did so by analyzing typical tasks related to space that arise, for example, in the context of a mobile tourist guide. Since we now dispose of a means to model theses basic processes in a way that takes into account situational information, the next logical step is to apply the approach presented in 4.1 to handle more complex tasks. The preverbal message is a key factor to achieve this goal as many tasks include interactions between a human user and the reasoning system. The PVM enables us to encode these interactions in a language- and media-independent way.

The power and flexibility of the modeling approach in this chapter is illustrated by the prototypical implementation, which we describe in chapter 6. Using our model, we were able to directly map the structure of the task decompositions (presented in the following sections) to corresponding agent teams within the prototype. The same is true for the interactions between tasks, which are equivalent to those between the corresponding agents. Furthermore, we introduce two complex tasks in addition to the ones analyzed in 2.5.1: geo-encoding of spatial expressions (4.3.4) and in-field data collection (4.3.5) in order to highlight the extensibility and flexibility of our approach. While the list of tasks presented in this section is certainly not exhaustive, the modular approach that we propose provides an easy means to realize further tasks related to space.

4.3.1 Identification

As we have seen in section 2.5.1, the identification of objects in the (immediate) environment is a typical task related to space that arises in many settings, e. g. a mobile tourist guide or a navigational assistant. An example for a typical scenario, where a human user would request this service, is the following: A user arrives at a location in an unfamiliar city, and sees a prominent building that is unknown to her. She then asks her 'mobile assistant' a question such as "What's this?". A typical reply to this request would be "This is the church of saint Peter.". We have already presented a first analysis of this service in section 2.5.1, and – based on this analysis – we can now use the modeling approach proposed in 4.1 to realize this service.

Figure 4.15 shows a detailed diagram of the interaction needed to perform the identification task *identify*. In case a user triggers the invocation of the identification task, her utterance[12] first needs to be translated from the natural language used to input the request into the generic representation format we proposed, a preverbal message containing a language-independent translation of the original request. Upon arrival of the PVM, we have to determine the current frame of reference, i. e. the user's current position, in order to restrict the set of potential target objects. Consequently, we request this information from the basic process *frame-ref*, which determines the current frame of reference and provides the identification task with this information.

The next step consists of generating a set of potential target objects, which can be done, for example, by querying an external GIS for all objects within the region defined by the origin and orientation of the frame of reference. If the original query contained further specifications – e. g.

[12]The user may not only use text or speech to input her request but could also rely on gestures or a combination of these means.

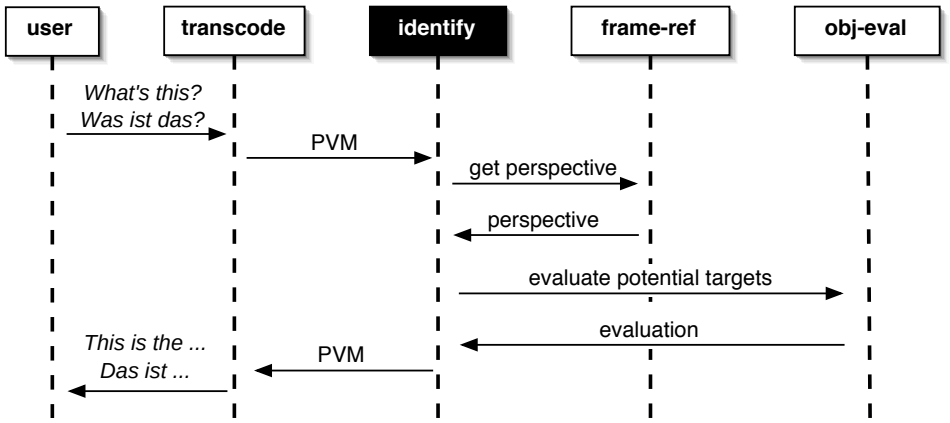


Figure 4.15: Object identification: Interaction of basic processes

an object type such as in “What’s this church?” – we can use this information to further reduce the set. Once the set of potential objects is available, we need to select the most likely target object(s) from it, i. e. the object that the user intended when formulating her request.

Hence, the set of candidates is passed to the object evaluation process *obj-eval* (see 4.1.1) along with a task-specific weight function. This weight function is very important as the impact of spatial factors as well as user- and context-related factors depends on the task which is currently performed. For example, when identifying an object in the context of a “What’s this?” query, the probability of a candidate increases considerably if it is visible within the current frame of reference. However, if the query was ”Tell me more about it.”, the visibility is of lesser importance.

The object evaluation process gathers information on the elements of the candidate set, evaluates them according to the task-specific weight function, and returns the evaluated set to the identification task. The most likely target – the object with the best evaluation – is then returned by *identify*. This result is again encoded as a PVM. Optionally, a small set of those objects that were rated best during evaluation can be returned, e. g. to allow the user to select the target object from a short list. If the user did trigger the object identification task, the returned PVM has to be translated into a suitable presentation such as a spoken reply “This is the ...” or any other suitable means (which we discussed in the previous section). It should be noted that the user is not the only instance that can trigger object identification: the next section will present an example where this complex process is used within another one.

An important issue in the context of object identification is the so-called *pars-pro-toto deixis*. In this case the user points at an embedded part or object while actually intending to refer to the object as a whole [Wahlster, 1991]. In order to properly account for this kind of referencing, further factors would have to be included in the object evaluation process (such as the internal structure and composition of objects). Within the XTRA project [Kobsa et al., 1986], several means have been proposed to cope with pars-pro-toto deixis, which also could inform the adaptation of the object evaluation process.

4.3.2 Localization

A second common task related to space is the localization of an arbitrary object or of the user herself. In section 2.5.1 we analyzed several example cases where this task needs to be performed such as when generating a reply to the question "Where is X?", or when providing navigational assistance (see 4.3.3). Based on the modeling approach presented in the previous section, we can now realize the localization task in a decompositional way as shown in figure 4.16.

Assuming that the localization is triggered by a user request such as the one depicted in the figure, it is first necessary to transcode the raw input into the generic representation format. In analogy to the identification task, the original input can be in any natural language, consist of gestures, or any other medium/modality or combination thereof. As long as there is a translation mechanism that transforms the raw input into a preverbal message, the localization task *localize* can be used without modifications. Consequently, other tasks can also relay localization requests using a PVM (see, for example, 4.3.3).

The first step in generating a localization consists in identifying the target object of the localization. Obviously, this corresponds to the complex task *identify* that was discussed in the previous section. Upon arrival of the response to the identification request – which are both encoded as a PVM – the *localize* task needs to determine the current frame of reference. This central information is provided by the basic process *frame-ref*.

In order to generate a relational expression, we also require an anchor object. Hence, the next step involves determining a set of potential anchor objects. Similar to the analogous step for the object identification process, *localize* calls upon an external GIS to gather objects within certain regions, e. g. near the origin of the frame of reference and the target object. These objects are then passed to the basic object evaluation process *eval-obj*, which evaluates them according to the specific weight function for the localization task.

The set of evaluated objects forms one of the two basic criteria for the generation of an appropriate localization. The other one consists of the set of those two-point relations that apply best for each potential anchor object in the current frame of reference. In order to obtain this set, *eval-obj* passes the set of candidates along with the frame of reference to the basic process *relations*, which computes for each element the two-point relation with the highest degree of applicability (according to the model presented in 4.1.3). The result of this process is a set of that has the same size as the one of potential anchor objects, and that contains the most applicable relation for each object.

The final step in the generation of a localization consists of combining these two sets by building a weighted sum for each object's evaluation and the corresponding degree of applicability of the selected two-point relation. This process yields a new ranking and the combination of anchor object and relation with the best overall rating is encoded in a preverbal message and returned to the querying agent. In case of a verbal response, the PVM is then translated into an appropriate target language resulting in an utterance such as "The X is to the left of Y.".

The localization of the user (self-localization) is a special case of the more general one described above. While we can treat it in the same way as the general case (depicted in figure 4.16), it is often requested in a situation, where the user is completely lost and/or when the system does not dispose of information about the current position of the user. On the one hand, a purely relational description may not help the user to get an idea about her current location. On the other, we may need to determine her position prior to being able to respond to her request (see section 5.3).

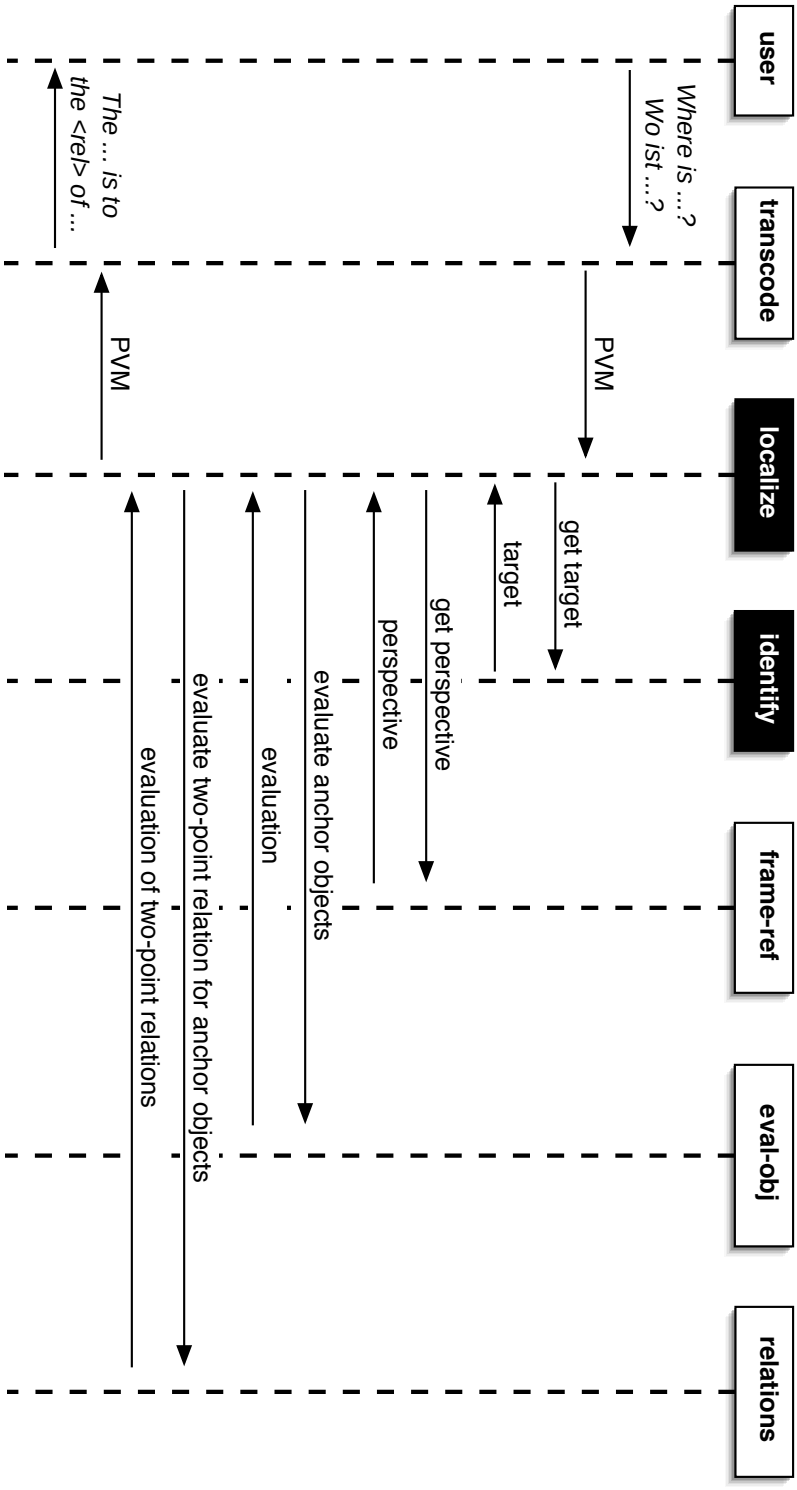


Figure 4.16: Localization: Interaction of basic processes and complex tasks

It is worth mentioning that the process proposed here to model the localization task does not have to be performed sequentially. In figure 4.16, a gray box indicates those steps that can be interwoven and/or be run in parallel (similar to the BOLA system [Blocher, 1999]): Determining a frame of reference, getting and evaluating anchor objects as well as the computation of two-point relations can all be performed simultaneously and/or incrementally. This does not only allow for resource adaptation but also for the implementation of anytime behavior [Zilberstein, 1996]. While we did not realize this feature in our prototypical implementation (see chapter 6), we designed it in a way that facilitates the inclusion of anytime behavior. For example, our implementation is based on a multi-agent system (see 6.2), which supports the incorporation of transactional concepts, and there is also a dedicated scheduler agent, that can be enhanced to implement anytime behavior.

4.3.3 Directions

The generation of incremental directions and complete way descriptions is arguably one of the most involved tasks as it relies on all basic processes described in section 4.1 as well as on several other complex tasks. It is also a very good example for the benefits of the modular approach that we propose: We were able to incrementally model this process using the available basic and complex tasks in a way that openly displays the underlying interactions and relationships. Figure 4.17 shows the resulting task model for incremental and complete directions.

Unlike object identification and localization the task of giving directions is most frequently triggered by a human user asking a question such as “How do I get to X?” or “Please guide me to X.”. As in the previous cases the input of the user is then translated into a preverbal message, which is passed to the directions task *route.* Again, the first step consists in determining the most likely target object[13] by means of the object identification task *identify.* In the case of no source location being specified, we require the current frame of reference or the user’s current position in order to determine the beginning of the route to be described. In order to obtain this information, we can rely on the basic process *frame-ref*.

Once the key locations/objects of the route are known, we can compute a suitable route. This is not only a computationally intensive process – it is closely related to the traveling salesman problem, which is in NP [Cormen et al., 1989] – but also very challenging if the user’s preferences as well as contextual factors are taken into account. We do therefore assume an external process that returns a tour according to the locations/objects passed along with the request. This process could also benefit from being modeled using the basic processes and complex tasks described in this chapter; for more details on this concept, refer to 7.2.

The next step after determining a suitable route consists in finding a segmentation that allows for the generation of appropriate directions. Obviously, this sub-task corresponds to the basic segmentation process *segment*, which provides the directions task *route* with this information. The resulting segment enables us to generate a matching n-point relation using the basic process *path-rel.* In order to further enrich the description of the current segment, we can then generate two-point localizations using the complex task localization (*localize*) for several key points of the

[13] It is also possible that we have to identify the most likely source object if the original query was of the type “How do I get from X to Y?” or even several objects if the query asked for a tour such as in “I would like to take a tour visiting X, Y, and Z.”

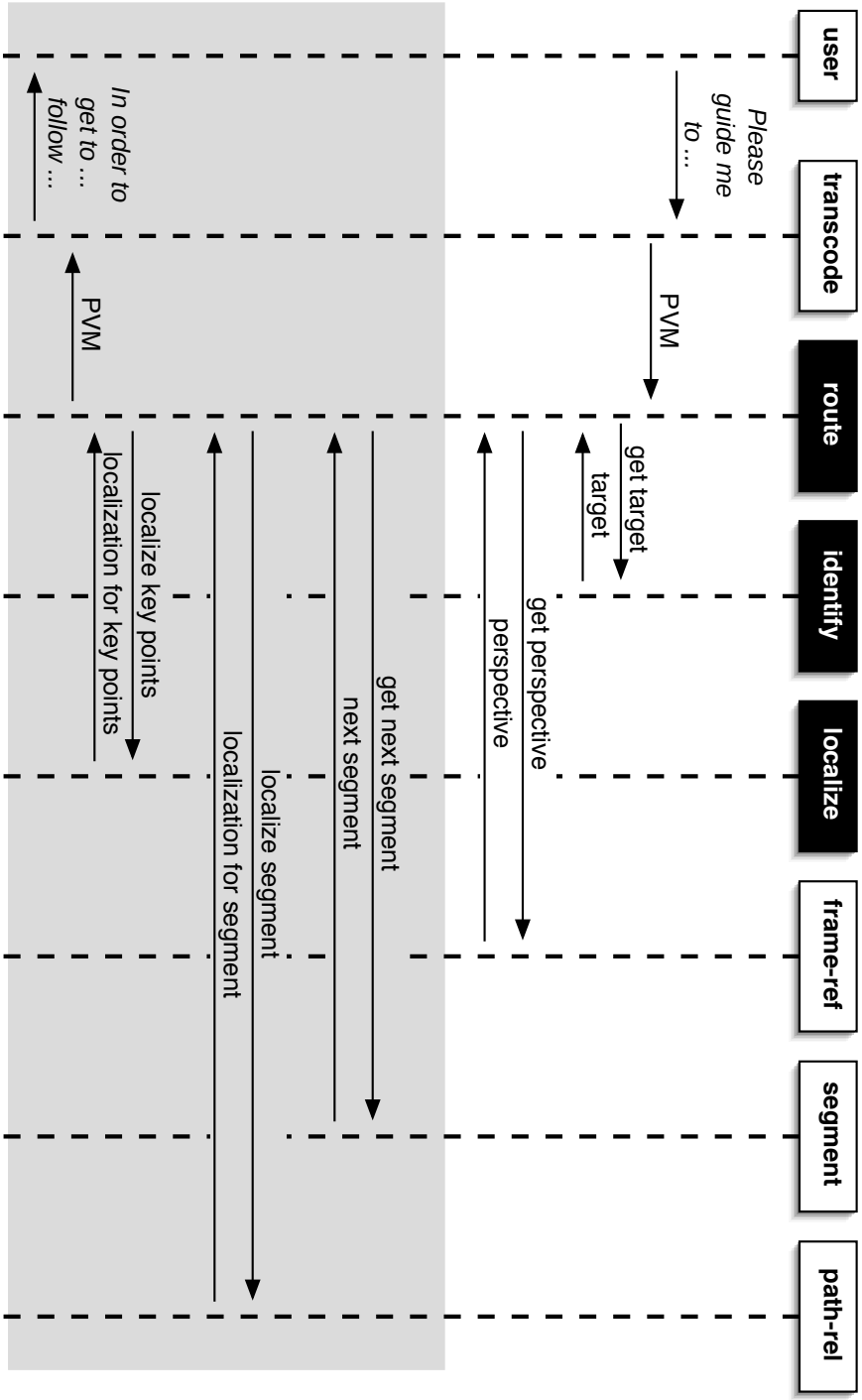


Figure 4.17: Directions: Interaction of basic processes and complex tasks

segment. This does not only include the start and the end point of the segment but also the segment as whole, which can be localized using a two-point relation as well. (Since the identification and the localization task incorporate the object evaluation process, the guidance task does indeed rely on *all* basic processes.)

This process results in a rich description of the current segment, which can then be encoded in a preverbal message. The information contained therein can then be presented to the user, e. g. by means of a spoken instruction such as “In order to get to X, follow street Y until Z is to your left.”. In analogy to the localization process, several sub-tasks within the directions task can be run in parallel; in figure 4.17 these processes are highlighted by a gray box. It is also possible to skip some subtasks, for example, in case of resource restrictions (see 2.4) – resulting in faster processing times but also in less rich descriptions. The steps highlighted in the figure also mark the difference between incremental and complete directions: In the incremental case, these sub-tasks are performed once every time the user reaches the end of the current segment. In the complete case, a description for the entire route is generated and then communicated to the user.

It should be noted that an important issue has to be addressed before the directions task can be applied in a real-world system: the handling of positional information. This is due to the fact that we need to track the user’s current position in order to determine whether she is still on the proposed route, and whether she has reached the end of the current segment. There are several ways to achieve this (e. g., pushing/pulling positional information, subscription based access), and in section 6.5 we present an innovative architecture for handling this issue.

4.3.4 Geo-encoding of spatial expressions

While the localization of arbitrary objects or the user was described in 4.3.2, the inverse case is also a very common complex task: the geo-encoding of spatial expressions. A typical example for this task is a request from the user that includes a spatial expression such as “I want to find a hotel *close to the central station.*”. While this input basically calls for a hotel reservation system, it is not directly apparent what “close” means in this context. Furthermore, if the user refers to the anchor object using an anaphora or if she does not use a term that allows for direct retrieval in a GIS/database, the hotel reservation system is at a loss as well. What is needed is a service that translates expressions referring to space into a geometric representation that can be used, for example, to directly formulate queries to the GIS. In order to provide this service, we can rely on the basic processes described in 4.1 and other complex tasks, combining them in a way that is depicted in figure 4.18.

In the figure we again assume that the complex task of geo-encoding (*geo-enc*) is triggered by a direct input of the user, e. g. a request containing a spatial expression. This request has to be parsed and translated into an internal representation; the relational statement can be transformed into a preverbal message. This PVM forms the input to *geo-enc*. It is worth noting that it could also originate from another task, [14] and that it does not have to come directly from a human user. However, the first step on the geo-encoding process is independent of where the original query came from; it consists of determining the anchor object relayed in the PVM. The complex task

[14] for example, from a task parsing a natural language description of a scene in order to generate a map or a graphical representation of the scene

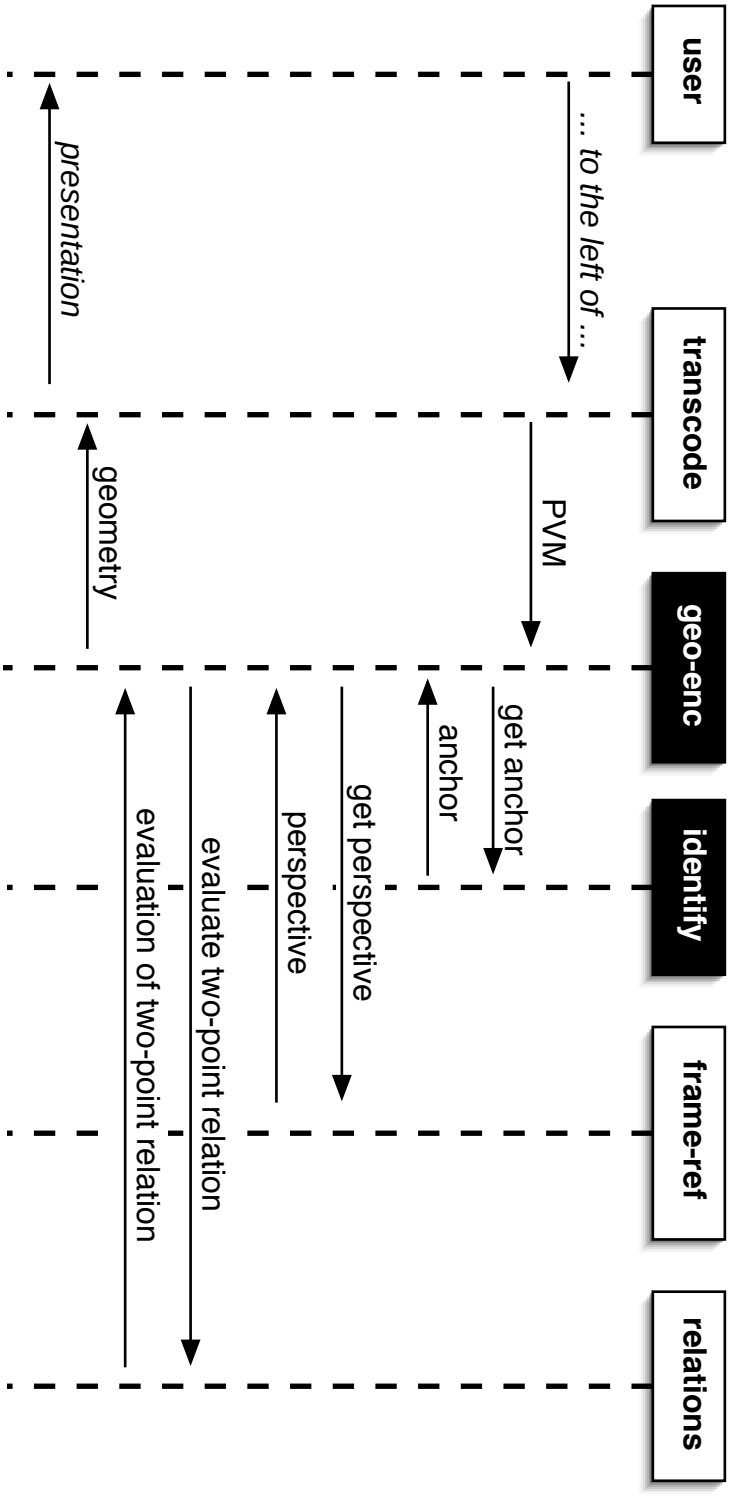


Figure 4.18: Geo-encoding: Interaction of basic processes and complex tasks

identify takes care of this and returns the most likely anchor object to *geo-enc.* The second component required for the resolution of a relational expression is the underlying frame of reference, which is provided by the basic process *frame-ref.*

Once these factors are known, we are able to compute the region corresponding to the relation included in the original PVM, which is obviously handled by the basic process *relations.* However, the process described in 4.1.3 was only concerned with computing the applicability of a relation for a single given target object. Since we are now dealing with concretion [Schirra, 1994] – determining target locations or area, where a given relation applies well – further analysis is needed. One solution that does not require any modification or extension of the basic process for evaluating spatial relations relies on a discretization of space: We select a discrete set of points on growing circles around the origin of the frame of reference that are equally spaced out. For these points, we can then compute the degree of applicability using the unmodified *relations* process. If a degree of applicability is above a certain threshold, the corresponding point lies within the region we want to determine, otherwise it lies outside the region. This region is equivalent to the convex hull of all points that lie inside of it. Thus, this procedure yields an approximation, and its quality is determined by the density of test points.

Since we are assuming that spatial relations are graded concepts with no crisp borders, an approximation is a suitable solution in most real world cases. However, an optimal solution can be obtained if we define an 'inverse' function $spline^{-1}$ for the $spline$ function introduced in 4.1.3. $spline^{-1}$ maps a degree of applicability *da* for a given frame of reference and anchor object to a (possibly) empty set of locations, where the $spline$ function evaluates to *da*. Whether the region is obtained using this approach or by approximation – once the geometry of the region has been determined, it is returned to the original entity invoking the geo-encoding task.[15]

4.3.5 Data collection

A very common problem for (mobile) systems that are put to real world use is that the world is continuously changing. For example, a building may be torn down while it is still present in the database, or a store may close and a restaurant may open in the same spot. In order to keep track of all these changes, a lot of effort has to be put into constantly monitoring the area modeled inside the system for changes. This does not only include the actual discovery of a change, but also the entering of the corresponding data into the world model. We call this complex task *data collection,* and we subsume the initial collection of information about new objects as well since they can both be captured by the same process (shown in figure 4.19).

The task model depicted in figure 4.19 again assumes that the task was initiated by a human user. This is also the most likely case as a 'non-human' initiation would require the artificial agent invoking the data collection task to be able to extract information from its environment, e. g. by means of image understanding. Since there are currently no systems that can extract highly complex information (such as 'the first floor of the building contains a restaurant') from an unstructured environment, the assumption of a human triggering the task is especially valid in this case. In analogy to the tasks described previously, the raw input first has to be translated into an internal representation (a PVM) before it is sent to the data collection task *data-coll.*

[15] In this case, encoding the reply in a PVM is not suitable as the goal of the geo-encoding task is to produce a *geometric* representation.

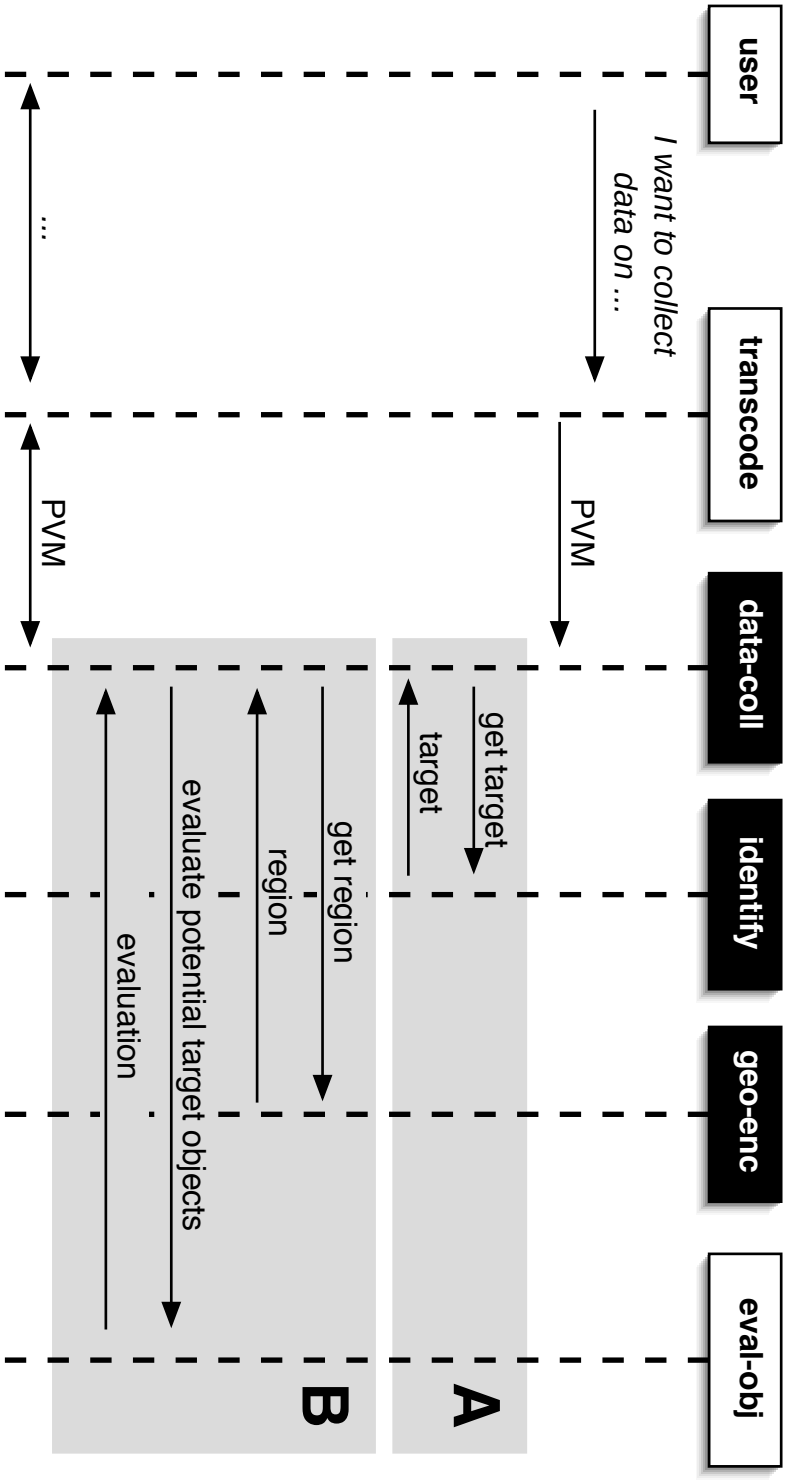
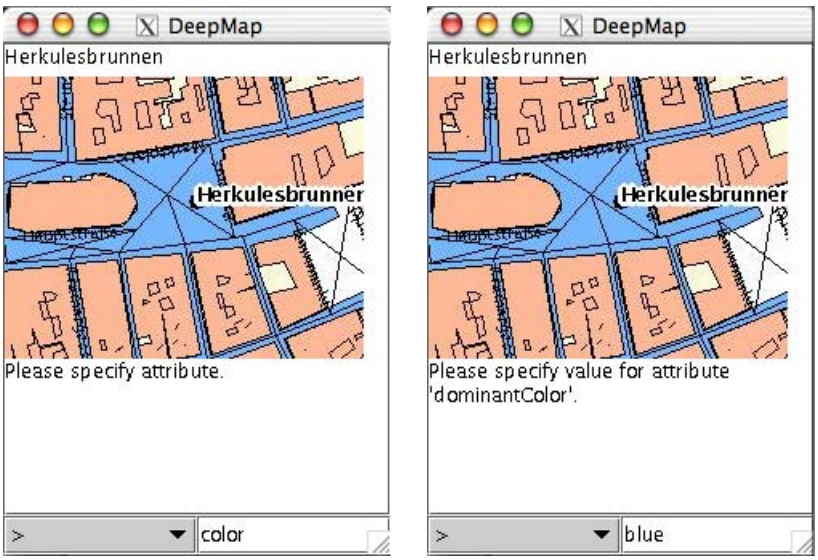


Figure 4.19: Data collection: Interaction of basic processes and complex tasks



(a) (b)

Figure 4.20: Data collection: An example interaction of a user specifying an attribute (color) and the corresponding value (blue).

There are two alternative ways how the target object (the one that the user wants to collect data on) can be specified: either directly (using its name or an anaphora) or indirectly (using a spatial relation such as "the building in front of me"). The former case implies that the first step in the data collection task corresponds to an invocation of the identification task *identify*) – in figure 4.19 this is depicted in box 'A'. The latter case is slightly more complicated: we first need to determine the region corresponding to the relational reference using the geo-encoding task *geo-enc* (see box 'B'). Once this region is known, we can retrieve potential target objects from a GIS, which we can then evaluate using the basic object evaluation process *eval-obj* in order to determine the most likely target object.[16]

After the target object has been identified, the remainder of the task consists of a dialog, where the actual data about the object is gathered, e. g. features, functions, or properties of the object. In the current implementation (see chapter 6), most of the attributes that are defined in the underlying ontology can be collected – including the color, main use, and object type. The corresponding interaction between the user and system consists of series of queries and answers, where the user first specifies what attribute she wants to collect data on, and then provides the data, e. g. the specific color or object type.

Figure 4.20 shows an example dialog. After the user has specified that she wants to collect data on the object in front of her (which happens to be the "Herkulesbrunnen"), she can then

[16]In neither case, the result has to be a single object: the implementation described in chapter 6 initiates a disambiguation dialog in case of several objects being likely target objects.

input data. She does so by first specifying the attribute “color”, which is than mapped an internal representation using the system-wide ontology. The latter one is also used to parse the values that the user than inputs for this attribute (in this example: “blue”), and to save the data once the collection is complete.

4.3.6 Map interaction

Maps play key role in transmitting various kinds of information such as the location of the user or the route she is currently following – for example, in the context of a mobile tourist guide or a car navigation system. Consequently, the interaction with maps – their generation and manipulation – is another complex task that we have to model. Using our modular approach, this is rather simple as figure 4.22 illustrates. The map interaction task *map-int* mainly relies on the object identification task *identify* to determine the target object of a map interaction. If no target object is given, e. g. when the user just asked “Show me a map.”, or if the target object cannot be identified, the map interaction process request the user’s current position/perspective from the basic process *frame-ref* in order to generate a map of her present environment. (Hence, the interactions shown in box ‘A’ and ‘B’ in figure 4.22 either take place both or just one of them.)

Figure 4.21: Map interaction: An example output

Once the target region is known, the next step is to select objects that should appear on the map. This calls for the object evaluation process *eval-obj*, which generates a ranking of all nearby objects. From this list, *map-int* choose those objects that are rated highest, and returns them along with the region to depict in the map. Figure 4.21 shows an example output from the system described in chapter 6, which is generated in reply to the query “Show me a map of the Torturm.”.

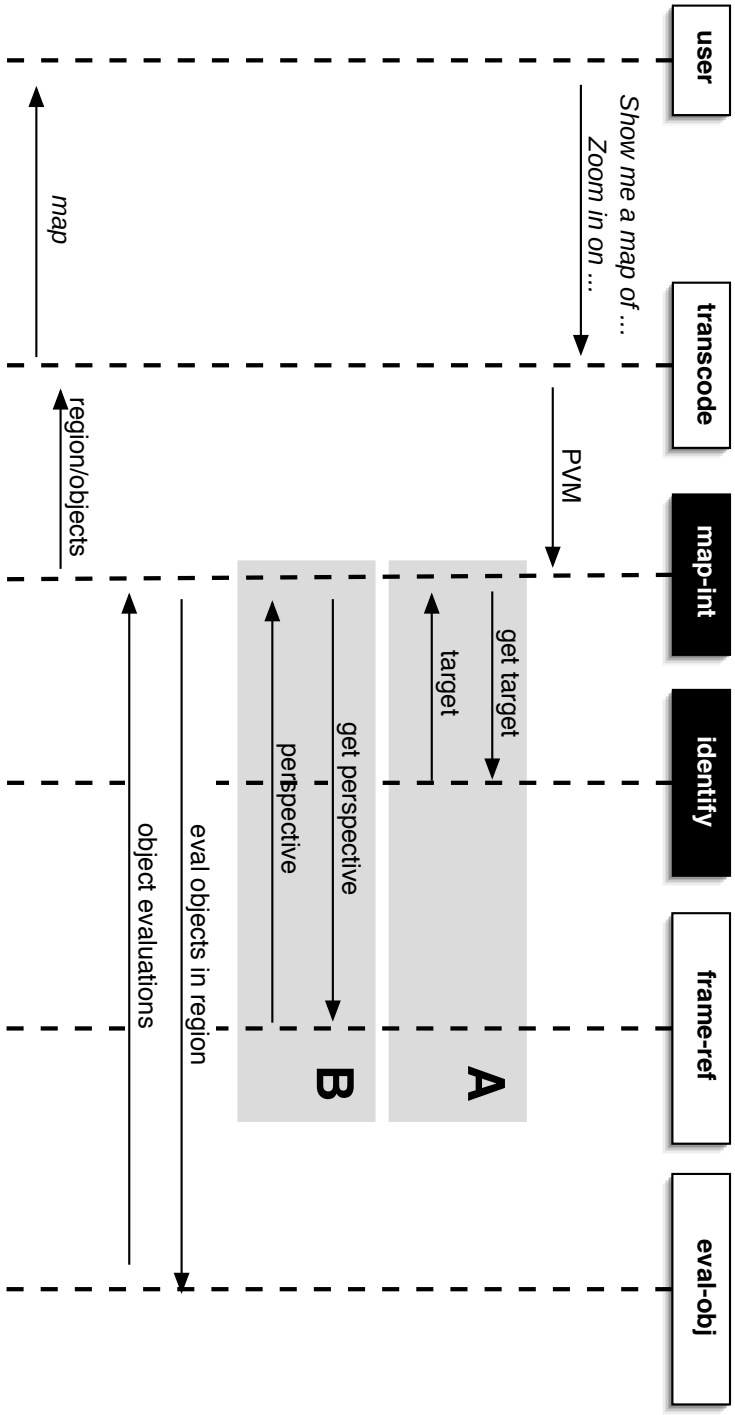


Figure 4.22: Map interaction: Interaction of basic processes and complex tasks

4.4 Summary

In this chapter, we introduced our approach for modeling processes and tasks related to space in a way that takes into account situational factors. We presented computational methods to realize the basic processes identified in section 2.5.2 that form the foundation of many more complex spatial tasks. Key points in this context were the determination of relevant factors for object evaluation and the application of the multi attribute theory (MAUT) to this process. In addition, we presented an analysis of induced frames of reference, which will also help to address the problem of missing or imprecise positional information in the next chapter. Furthermore, we introduced an approach for the computation of path relations that is based on empirical evidence, and we presented a path segmentation algorithm that relies on a distance concept based on situational factors.

In order to realize more complex tasks that involve direct interaction between a human user and the system, we then addressed the problem of language independent representation. The main requirements in this context were its independence of a specific target language, support for different media and modalities as well as the possibility to encode both the human's utterances and those of the system (e. g. in order to maintain a unified dialog history). In order to address these needs, we proposed the preverbal message (PVM) as a means to address the requirements stated earlier. The PVM not only enabled us to represent all the interactions arising in the context of the complex tasks (see below) but also to encode them in a very concise format. Furthermore, we presented several examples illustrating that the preverbal message does allow for the flexible generation of natural language as well as for pictorial or graphical output.

The last section of this chapter was concerned with the modeling of complex tasks related to space. In addition to the ones introduced in 2.5.1 we presented three further tasks, namely the geo-encoding of spatial expressions, the in-field collection of data on real world objects as well as the interaction with maps. All these tasks can easily be modeled using the basic processes described earlier, and the modular approach we propose also makes the interaction between those processes explicit. For each of these tasks, we provided an interaction diagram, which shows what steps are required to perform it, and what basic processes or complex tasks are involved in addressing the corresponding task. In this context, all interactions with the user were encoded using preverbal messages.

The next chapter will build upon this model, and show how it can be used to address several common problems that arise in the real world application in a mobile system.

5 Adaptation strategies

When a model such as the one described in chapter 4 is developed, its designers often start from an ideal world in order to concentrate on the central concepts and processes. However, once such a model is put to use in a real world application, a new set of problems arises. Since the world that we live in is not a perfect one, it is quite possible that information needed in the reasoning process is simply not available, or at a lower precision than what the model requires. Consequently, building a truly helpful system for real world use means to take these issues into account. Therefore, we designed several adaptation strategies for common problems in spatial interaction on spatial topics. These strategies have enabled us to apply the modeling approach presented in the previous chapter in a real world scenario. The system that we implemented to illustrate this is a central part of a mobile tourist guide (see chapter 6), but the strategies described in this chapter are sufficiently general to apply in other scenarios as well.

In this chapter, we first present an analysis of different ways to adapt to the lack of information such as situational factors in 5.1. We then review strategies to cope with the limited availability of cognitive and technical resources in 5.2. Since positional information plays such a central role in mobile computing and in determining the current situation, we propose an comprehensive approach to dealing with it in 5.3.

5.1 Lack of situational information

A frequent problem arising in the context of human-computer interaction is the unreliability of information sources: often, sensors will not return sufficiently precise information or no information at all, network connections will fail disabling access to remote databases, and some information may be immeasurable and can only be derived over time (e. g. the user's interests). Consequently, if a system is to be used in a real world scenario, it is important for it to function properly even if some information is missing. In this section, we will discuss several general approaches to address this issue, and present concrete ways how to apply those to the basic processes described in 4.1. On an abstract level, we can address the lack of information in several general ways:

- **ignoring missing information**
  A straightforward approach of handling missing information lies in ignoring it, e. g. adapting weighting factors and removing the terms including this information from the corresponding formulae.

- **accessing alternative sources**
  Another obvious way to overcome the lack of information is to retrieve it from alternative sources should the primary source become unavailable. An example for this approach would be a scaled-down version of a database that runs locally and that is used in case the large remote database cannot be accessed due to network failure (see, for example, the GUIDE system presented in 3.3.

- **using default values**
  A third way to deal with unavailable information is to rely on default values where information is missing. For example, we may assume that we are dealing with an adult user if we dispose of no age information.

- **inferring missing information**
  Instead of using default values, it is also possible to infer some missing information, albeit at the cost of increased unreliability. An example for this approach is the dead reckoning algorithm presented in 5.3.1.

- **adapting computation**
  A further means to handle the lack of information is to explicitly evaluate the availability of information during computation. For example, we can assign a degree of precision or confidence (cf. [Kray, 1998]) to the result of a computation that reflects the amount of information that was available to compute it. Furthermore, the availability of information can inform the process of selecting among a number of alternative algorithms. For example, a system may resort to a simplified method that is solely based on geometry if no situational information is available.

- **requesting information from the user**
  In case of an interactive system, a further means to address the lack of information lies in directly asking the user to provide this information. In 5.3, we present a mechanism based on this approach.

Obviously, not all these approaches are suitable for any kind of information. For example, not all information can be inferred: We cannot derive the user's interest, if we do not have a record of previous interactions with the system, or if the user did not provide us with this information. Therefore, an analysis of which methods are applicable to which basic spatial reasoning process is required.

The evaluation of objects was the first basic process we described in section 4.1. The approach that we proposed for modeling it is based on the Multi-Attribute Utility Theory, which facilitates the handling of missing information. The basic formulae used to evaluate objects equations 4.3 and 4.4 are shown below:

$$\begin{aligned} oe(x) &= \sum_{i=1}^{n} w_i(t) oe_i(x) \\ oe_i(x) &= \sum_{a \in A_i} w_{a,i}(t) oe_{ai}(l(a)) \end{aligned}$$

If we assume that all object evaluation functions and weights are normalized on all dimensions (i. e. $oe(x), oe_i(x), oe_{ai}(l(a)), w_i(t), w_{a,i}(t) \in [0.0; 1.0]$), we can easily ignore missing information by reassigning the weights accordingly: We normalize over the sum of weights for the available information, and recompute the corresponding weights in relation to the new overall sum – thereby maintaining the relative relevance compared to the other weights.

$$W' = \sum_{j=1}^{n} w_j(t) \cdot available(oe_i(x)) \tag{5.1}$$

$$\forall j \in [0; n] : w'_j(t) = \begin{cases} \frac{\sum_{i=0}^{n} w_i(t)}{W'} w_j(t) & \text{if } available(oe_i(x)) \\ 0 & \text{otherwise} \end{cases}$$

Equation 5.1 shows the modification required to implement this procedure: the new weights $w'_j(t)$ are normalized over the sum of the weights associated with available information. The function $available(oe_i(x))$ returns $oe_i(x)$ if the information on dimension i is available, otherwise it returns 0. We can apply the same method recursively to $w_{a,i}(t)$ and $oe_{ai}(l(a))$.

In addition to ignoring missing information, all the other general strategies presented above can be applied to the specific case of object evaluation – which is mainly due to the flexibility provided by MAUT. While it is quite clear that accessing alternative sources, using default values, inferring missing information, and asking the user are somewhat orthogonal to this approach and – if implemented – easy to include in this scheme, the adaptation of computation to account for the lack of information needs further investigation. One way to realize this strategy lies in introducing a confidence dimension and then recursively applying MAUT to it. Equation 5.2 lists the corresponding formulae:

$$\begin{aligned} oe^{+}(x) &= \sum_{i=0}^{n} w_i(t) oe_i^{+}(x) \qquad (5.2) \\ oe_i^{+}(x) &= \frac{1}{k} \sum_{a \in A_i} w_{a,i}(t) oe_{ai}(l(a)) + (1 - \frac{1}{k}) \sum_{i=0}^{n} w_{a,i}(t) confidence(oe_{ai}) \end{aligned}$$

The regular object evaluation function $oe(x)$ is replaced by a new function $oe^+(x)$ that factors in confidence by means of $confidence(oe_{ai})$. This function evaluates the confidence the system has in the the object evaluation function for attribute a on dimension i, and results in a confidence value between 0 and 1. The most simple approach to model $confidence$ is a function that returns 1 if information is available, and 0 if it is not. More sophisticated methods could employ sensor models and explicit reasoning about the quality of the corresponding information source. The confidence value is again weighted with $w_{a,i}(t)$, as is the overall sum of all confidence values. This sum is added to original $oe_i(x)$ after being weighted by a factor $k \in [0;1]$ that determines the importance of high confidence in relation to the actual attribute values: the closer k is to 1, the more relevant are the attribute values. The closer it is to 0, the more important is the confidence value.

In addition to object evaluation, we identified four other basic processes. Since frames of reference will be discussed in section 5.3, we will focus on spatial two- and n-point relations as well as path segmentation here. For these basic processes, the distance function *scale* respectively *max_length* (see 4.1.5) are the central mechanisms to take into account situational factors. Consequently, we have to concentrate on how to compensate for the lack of information in these functions (equation 4.29 and 4.27 – also shown below):

$$\begin{aligned} length(m, \mathfrak{U}) &= length(m) \cdot scale'(phys\_const \in \mathfrak{U}) \cdot scale'(age \in \mathfrak{U}) \\ scale^\star(\mathfrak{C}, \mathfrak{U}) &= \prod_{i=0}^{n} scale'(c_i \in \mathfrak{C}) \cdot \prod_{j=0}^{m} scale'(u_j \in \mathfrak{U}) \end{aligned}$$

A straightforward method of ignoring missing information consists in simply redefining $scale'$ so that it returns 1 in case some information is not available. Except for the adaptation of these formulae, all other general strategies to deal with the lack of information can easily be applied as well, since they are orthogonal to the computation. In analogy to the approach proposed for object evaluation, the functions shown above can be adapted using a confidence measure as well.

$$scale^+(x) = \begin{cases} 1 - (1 - scale'(x)) \cdot confidence(x) & \text{iff } 0 \leq scale'(x) < 1 \\ 1 + (scale'(x) - 1) \cdot confidence(x) & \text{iff } scale'(x) \geq 1 \end{cases} \tag{5.3}$$

Equation 5.3 shows a possible adaptation of the scaling function: Based on a confidence value $confidence(x) \in [0;1]$, $scale^+(x)$ returns a value between 1 and $scale'(x)$. When the confidence is low, the resulting value approaches 1 (eventually equalling 1 when the confidence value is 0), thereby effectively reducing the impact of x on the overall scaling (eventually eliminating it altogether when $confidence(x)$ equals 0). When the confidence value is high, $scale^+(x)$ approaches the original value $scale'(x)$. In case of perfect confidence, $scale^+(x)$ equals $scale'(x)$.

5.2 Resource restrictions

The interaction with an artificial system strains the cognitive resources of its user at a varying degree (see section 2.4). In order to provide the user with information that is tailored to her current situation, we have to take this into account. Otherwise, the system might generate an elaborate presentation that requires a lot of concentration to decode in a situation, where the user has to perform a secondary task (such as driving a car) and cannot afford to concentrate solely on the output of the system. In addition to resource restrictions on the cognitive level, we have seen that technical resources may also be restricted. Especially in a mobile scenario, this seriously impacts system performance since mobile devices inherently are more severely limited in terms of computational power and output capabilities than stationary gear. In the following we will discuss several means to address resource restrictions on different levels.

5.2.1 Adaptation of presentation

One way to address the problem of limited resources lies in generating different types of presentations that vary in terms of their complexity. Hence, we designed the abstract format for specifying what the system should say – the preverbal message (see 4.2) – with this kind of adaptation in mind. It allows not only for verbal output in various languages but also supports a number of graphical realizations, which we presented in section 4.2. In general, the more sophisticated a presentation is, the more technical resources are required to generate it. However, the same does not necessarily apply in the case of cognitive resources. For example, on the one hand, a highly detailed three-dimensional rendering of an object may be computationally expensive to produce but it may be relatively easy for the human user to match it with the actual object in the real world. On the other hand, a very precise verbal route instruction, which is computationally less expensive, may be very hard to decode by the recipient and thus allocate a large share of her cognitive resources.

The most simple presentation type consists of a verbal output such as the ones presented in section 4.2, which can either be realized in textual form, as spoken text, or as a combination of both. Spoken text is well suited in cases, where a secondary task has to be performed, as it frees up the user's eyes (and possibly, her hands, too, if she is using a headset). Depending on the underlying text generator (e. g. simple pattern-based approaches versus sophisticated planning [Wahlster, 2000]), it is possible to produce textual output without requiring the allocation of considerable technical resources. The same is true in the case of speech generation, which relies on text generation. Depending on the degree of sophistication, the amount of required resources can be kept low.

2D sketches share several properties with verbal presentations: they are fairly abstract, which facilitates their generation from the abstract representation format (PVM). It is also possible, to generate them in a way (e. g. by concatenating previously rendered symbols or by only drawing simple geometrical shapes) that requires little technical resources. Generating 2D maps is a process that, in general, consumes significantly more resources than simple verbal output or 2D sketches. Depending on the implementation, a map has to be generated from a large dataset (or has to be clipped from a larger image), which may include the selection of an appropriate zoom factor, of a level of detail, and of which objects to depict. Placing labels on the corresponding map is also

a computationally demanding task. And if the map is aligned to the user current view direction, it has to be rotated continuously. However, a map does provide much more context than verbal instructions as it naturally includes nearby objects. This may be beneficial in terms of reducing the cognitive load, for example, when the user has to orientate herself in an unfamiliar environment.

3D visualizations are usually even more demanding in terms of technical resources than 2D maps. Depending on the algorithms used to produce such graphics (e. g. ray-tracing, or volume renderer), these presentations cannot be generated in real-time – even on current high-end desktop workstations. However, there may be cases, where they can contribute to the recognition of landmarks which may make it easier for a human user to find her way around.[1]

In addition to selecting a specific type of presentation, the preverbal message also allows to adapt the presentation content to some degree. A PVM often (and purposefully) contains more information than is strictly required to answer the corresponding query of the user. This provides the component generating the presentation with some flexibility in terms of selecting what to include. Evidently, this flexibility also allows for the adaptation of the presentation content in response to technical and/or cognitive resource restrictions.

A PVM for directions, for example, does not only contain a spatial relation describing the location of the start and the end point of the corresponding segment of the route, but also a path relation and another two-point relation describing the shape and location of the trajectory. It further includes metric information about the length of the segment as well as angular information encoding the turn that may be required at the beginning of the segment. Since not all of this information is required to generate a route instruction, the system can adapt to resource restrictions by gradually considering fewer entries. The resulting verbal presentations can range from a short number of sentences realizing all entries to a simple turn instruction. See equation 5.4 and sentences (9) to (13) for an example PVM and the corresponding realizations. Note that the fields of the PVM, which are not used in the presentation, do not have to be computed and hence reduce the amount of technical resources needed to generate the PVM.

$$pvm_{example} = (\text{inform}, \{\text{directions}\}, \mathit{Jesuitenkirche}, \Re_{example}, \emptyset, 305, 124.0m) \quad (5.4)$$

where

$$\begin{aligned} \Re_{example} = \{&(\text{start}, \texttt{next}-\texttt{to}, \mathit{Marktbrunnen}, 1.0), \\ &(\text{end}, \texttt{next}-\texttt{to}, \mathit{Fischmarkt}, 0.67), \\ &(\text{path}, \texttt{left}-\texttt{of}, \mathit{Heiliggeistkirche}, 0.73), \\ &(\text{path}, \texttt{follow}, \mathit{Hauptstraße}, 0.92)\} \end{aligned} \quad (5.5)$$

> In order to get to the Jesuitenkirche, locate the Marktbrunnen, which is next to you. Turn a little bit left, and follow the Hauptstraße for about 120 meters until you are next to the Fischmarkt. You will see the Heiliggeistkirche on your left as you walk. (9)

[1] At least, the field test we reported in [Kray et al., 2003] provided some evidence that this may be a benefit of using 3D graphics in route instructions.

Turn a bit left, and follow the Haupstraße for about 120 meters until you are next to the Fischmarkt. You will see the Heiliggeistkirche on your left as you walk. (10)

Turn a bit left, and follow the Haupstraße for about 120 meters until you are next to the Fischmarkt. (11)

Turn a bit left, and follow the Haupstraße until you are next to the Fischmarkt. (12)

Turn a bit left. (13)

Generally speaking, the amount of technical resources needed to generate a presentation decreases as the content is reduced since fewer items have to be computed and/or considered in the process. However, it is much harder to estimate the amount of cognitive resources that is required to decode a specific presentation. For example, a verbal instruction, which may be easy to understand for a human user in one situation, may be much more difficult to decode when she is performing a secondary task that interferes with the decoding. Consequently, while the adaptation of the presentation type and content may help to address limitations of cognitive resources, there is no simple relationship between certain presentations and the amount of cognitive resource a human user would have to invest in order to understand it.[2]

5.2.2 Adaptation of computation

An alternative approach to adapt to varying resources was used, for example, in the system BOLA (cf. [Blocher, 1999]), where depending on the available (technical) resources different computational methods were applied. While this is also possible within the model presented in this thesis,[3] there are several other means that exploit specific properties of the model and its prototypical implementation presented in chapter 6.

Due to the very modular nature of the model, it facilitates an implementation as a multi-agent system (see chapter 6.4) which entails a number of possibilities in terms of adapting to varying resource restrictions. One of the frequently cited advantages of multi-agent systems [Jennings, 1999] consists in the ability to autonomously decide when, how, and where to perform its computations. This enables an agent implementing our model to evaluate the current availability of the resources it requires (such as bandwidth or computational power), and to move to a different agent platform if its current location does not provide the resources it needs.[4] More specifically, this approach helps to address situations were bandwidth, memory, or computational power are limited: If an agent needs to interact with a remote information source (such as a large-scale database) and the bandwidth of the connection is limited, moving to the remote location and then performing the

[2]The investigation of this relationship is one of the goals of the Special Research Center 378 (resource-adaptive cognitive processes), which includes, for example, the REAL project (see 3.7).

[3]One way to achieve this, for example, would be to use different computational methods for each basic process and then switching between them depending on the current resource availability. This idea is shortly discussed in 7.2.2.

[4]Of course, the agent platform has to support both agent mobility and resource awareness such as the one used in the prototype implementation [Ding et al., 2001].

interaction helps to reduce the required bandwidth. The same is true if the local memory or computational power is insufficient for the agent to perform: moving to a remote site reduces the local load and enables the agent to succeed.

For 'external' sources of information such as a GIS or a database we can apply a similar strategy that exploits the fact that multi-agent systems usually include look-up or directory agents (see, for example, [The Foundation for Intelligent Physical Agents, 2002]). These agents maintain a list of all available agents that provide certain services, and it is possible to include several agents that provide the same service. Consequently, it is straightforward to have a number of agents that realize a given service at various levels of quality, but that also consume a different amount of resources. For example, if we assume that the system includes a large remote GIS, there can be a local agent providing the same services but that consumes fewer resources because it relies on a simplified model compared to the remote server. Such a scenario allows for the adaptation to various resource restrictions: If the connection is lost, unreliable, or slow, we can rely on the local GIS. If the time to compute a reply is limited, we can use both services, either by using them in parallel (and taking the results from whichever agent is faster) or by alternating between them and/or overlapping queries. If local memory or computational resources are tight, we can rely on the remote server.

5.3 Positional information

In a mobile setting, information about the user's current location, her orientation, speed, etc. is a key element in determining what situation the user is in. In an ideal world, all positional information would be precise and available at all time. However, real sensors fail and are prone to errors (see section 2.6). Therefore, a mobile system has to be able to cope with sensor failures, imprecise information, and errors. The following sections present several strategies to handle positional information of varying quality. There are two basic directions from which we attack the problem: On the one hand, the system can try to improve the available positional information by means of additional knowledge sources and/or interaction with the user. On the other hand, is it possible to adapt the task that requires information about the user's current position to the quality of the available data. The strategies presented in the following sections hence include non-interactive techniques that solely rely on the information available in the system (section 5.3.1) as well as approaches that include interaction with the user (section 5.3.2 and 5.3.3).

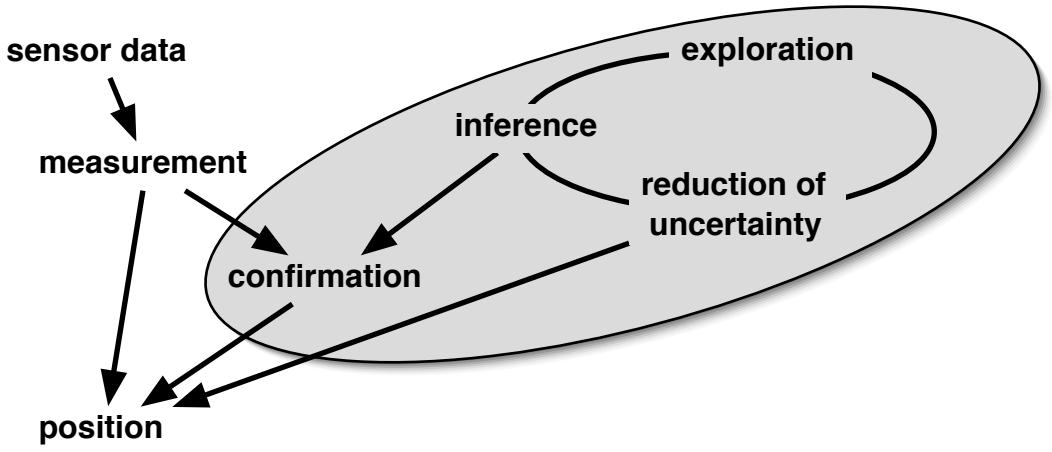


Figure 5.1: Determination of the user's current position.

Figure 5.1 depicts the relationship between these strategies: In order to determine the current position, the system starts out with sensor readings. If these meet the current requirements as far as precision and recency are concerned, no further processing is needed. In case the confidence value of the current measurement falls below a given threshold, the system can ask the user to confirm the position. If the available sensor data does not allow for a single hypotheses, the system can try to infer it and/or to disambiguate between a small number of potential positions. In case no measurement is available or there are too many hypotheses, the system needs to explore, where the user is located. The following sections provide a detailed analysis of these strategies.

5.3.1 Inference: Knowledge-based dead reckoning

When positional information is not available at all or with less than the desired precision, it is often possible to infer it from other data sources than direct measuring: If the location, view direction, etc. are logged over time and stored in a 'position history', it is a straightforward task to extrapolate the current position from the one that was most recently stored and from the motion vector at that point of time. This process is also called *dead reckoning* (see, for example, [Lee et al., 2000]). (The basic idea of dead reckoning has been used for long time, e. g. by sailors navigating the sea in ancient times.) The initial 'guess' can be refined using several heuristics and knowledge sources:

The model of the world, e. g. vector data stored in a geographic information system (GIS), provides constraints that can help to eliminate wrong projections. If the extrapolated position lies within 'impassable' or 'unreachable' terrain, or such a terrain would have to be traversed in order to reach that position, then it has to be adjusted taking into account these constraints. 'Impassable' in this context includes not only real obstacles such as a river but also terrain impassable to the current user such as private houses and highways. The same is true for the term 'unreachable', which is meant to apply to strictly unreachable locations (e. g. a mountain top) and to currently unreachable ones. Consider, for example, a position within a fenced park, where the sole entry is farther away than the user's last measured speed would allow her to travel within the time since the last measurement. This example also illustrates that this reasoning is based on some default assumptions, e. g. an 'average user' does not climb over fences in an 'average situation'. In order to account for non-default situations, we would have to explicitly model the passability of an area depending on the current situation.

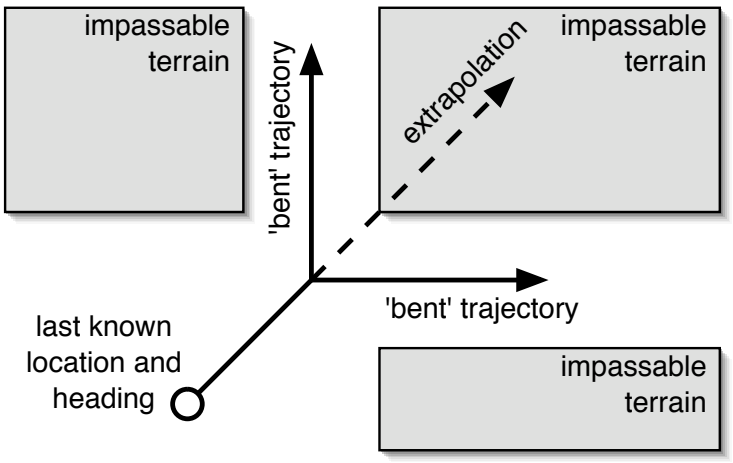


Figure 5.2: Bending straight extrapolations according to world model

A first approach to adjusting the extrapolated position according to these constraints is to map it onto the route/path network and/or to passable terrain by 'bending' the projected trajectory around impassable/unreachable terrain (see figure 5.2). This process can be refined by taking into

```
retrieve position history
    infer heading from last two positions
    infer speed from last measurements or situational factors
map position to street network
    find streets nearest to last measurement
    project position on streets
determine potential positions
    determine maximum distance
    follow street
      if node is reached
        follow outgoing edges
      else if maximum distance reached
        store position
```

Figure 5.3: Dead reckoning algorithm: an overview

account additional knowledge such as the user's current goal, which can help to resolve ambiguities. Consider, for example, a large obstacle on the path to the extrapolated position: if the user's target lies more to the left of the obstacle, she is more likely to circumvent it on the left. This is a simplified view: more precisely, the projected path to the target must be considered – which might lead around either side of an obstacle – so that the probability of positions close to the path increases.

Figure 5.3 gives a schematic overview of the dead reckoning algorithm. In order to apply it, the position of the user has to be tracked continuously and to be stored in a position history. Initially, the algorithm retrieves the most recent positions from it, and uses them to infer heading and speed at that time. In absence of measurements for speed, several situational factors allow for an educated guess: If information is available on the user's current means of transportation, an average speed can be calculated. In case the user is walking, riding a bicycle, or using any other means of transportation that relies on physical exercise, additional factors such as her physical constitution or age can inform the estimation process.

Then, the last known position is mapped onto the street network. In order to achieve this, all neighboring streets are collected and the position is projected onto them. This results in a set of projections from which the algorithm selects the one that deviates least from the last known position in terms of distance and heading. The mapped position serves as the starting point for determining which points the user could have reached since the last measurement under the assumption of unchanged travel speed.

Beginning with the starting point on the street network, the algorithm follows all outgoing edges that do not point in the opposite direction of the previously computed heading. If a node is reached, again all outgoing edges are followed except those that lead back to nodes that were already visited beforehand.[5] This process is stopped once the distance travelled from the starting point equals the previously computed maximum distance. The corresponding point on the street

[5] assuming that the user is not going in circles

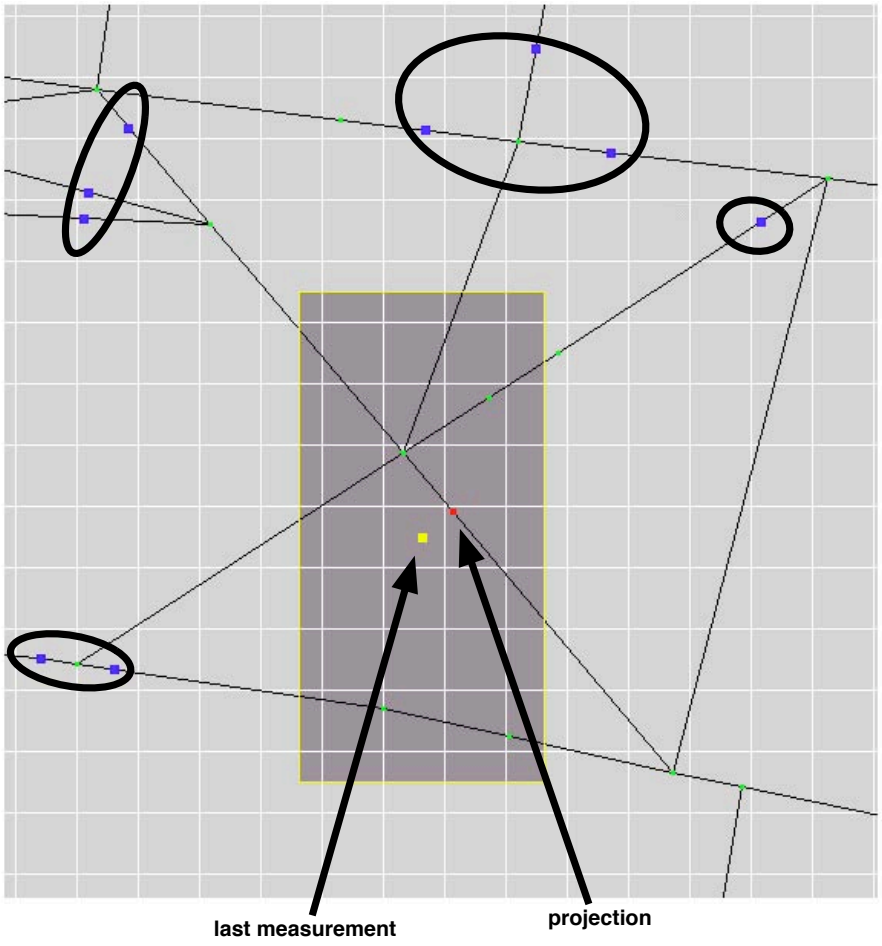


Figure 5.4: Dead reckoning: screenshot with last measurement (yellow dot), projection (red dot) on street network (black lines), and resulting position hypotheses (encircled blue dots)

network is stored in the set of potential positions. The algorithm terminates since the maximum distance is finite, and it does so once all branches have been explored up to the maximum distance. Figure 5.4 shows an example visualization generated by the system presented in chapter 6, which implements this algorithm. The dark rectangle illustrates the error region from the GPS signal.

The result of the dead reckoning algorithm is a set of potential positions under the assumption that the user did not change his speed and direction. Ideally, this set has only one element, which can be considered the most likely position. This is the case, for example, when the last confirmed measurement is very recent and precise, or when the route network is rather sparse. More often, the algorithm returns several potential positions as their number increases sharply the longer the last measurement dates back and the more nodes are reachable within the street network. Finally, if no sensor readings are available, the set of potential position is of infinite size.

It is possible to further process a small set of potential positions that are close to each other in

order to obtain a single position by collapsing them while merging their respective error regions. The two potential positions in the lower left corner of figure 5.4 can serve as an example for such a situation. The resulting position is less precise than the original ones but if it suffices for the current purpose no further reasoning or interaction is necessary. However, as the number of potential positions grows and the distance between them increases, this approach becomes unfeasible. Nevertheless, it is still possible to come to a single, fairly precise hypothesis if we have access to a world model and if we can interact with the user.

5.3.2 Reduction of uncertainty

The process of selecting a single hypothesis from a set of potential positions can be seen as a *reduction of uncertainty* about the user's current position. Only in some cases is it possible to successfully reduce the set solely from internal knowledge sources. For example, if the user currently follows a route that was suggested to her by the system, the likelihood of nearby positions increases. This may help to pick a most likely candidate from a small set of alternatives.

However, more often this approach is not feasible since the available situational knowledge does not provide sufficient evidence to select a single position. At this point, interacting with the user may be beneficial, e. g. by asking her whether she can see a certain objects. A human user has several abilities that computers are still striving to attain such as instantaneous object recognition, extensive world knowledge, and senses that are continuously active. Consequently, humans can easily check whether an object they know is visible, while a computer would require a camera, computationally expensive image recognition algorithms, and a large knowledge base in order to achieve this. Even then, simple changes to an object (such as a house being painted in a different color, or a scaffolding put up at a building) will still be a great challenge for an artificial system. The same is true for every day items such as street signs: while humans may pick up the name of the street subconsciously, computers would have to actively look for street signs, run OCR[6] algorithms on the camera image, and then try to find the (potentially misspelled) street name in their database.

This situation clearly calls for a cooperation between a human user and her mobile computer. Instead of failing when faced with more than a single position, the system can instead use its world model to determine for all potential positions, which streets they are located on. If the number of hypotheses is small (two to five) it is often the case that each position lies in a different street. Then the system can ask the user, which street she is in. Even if she does not know right away, in an urban area there is often a high probability that a street sign is nearby. Once the user tells the system the name of the street she is in, the corresponding position becomes the most likely hypothesis. Should the user be unable to provide that information, the system has to fall back to exploration (see section 5.3.3), or to rely on the following approach.

Asking for street names can be inappropriate under certain conditions: The number of potential positions may be higher than just two to five, or several positions may be located on the same street. Or, the user may not know the name of the street she is on. In that case, an alternative approach provides a means to still determine the user's current position. This requires that the system has access to a rich database that contains textual and graphical information about the objects of the

[6] optical character recognition

$$V(S,P) = \begin{pmatrix} vis(s_1,p_1) & vis(s_1,p_2) & \dots & vis(s_1,p_n) \\ vis(s_2,p_1) & vis(s_2,p_2) & \dots & vis(s_2,p_n) \\ \vdots & \vdots & \ddots & \vdots \\ vis(s_m,p_1) & vis(s_m,p_2) & \dots & vis(s_m,p_n) \end{pmatrix}$$

where $S = \{s_i | 0 < i < m+1\}$ (set of salient objects)
$P = \{p_j | 0 < j < n+1\}$ (set of positions)

and $vis(s_i,p_j) = \begin{cases} 1 & \text{iff } s_i \text{ is visible from } p_j \\ 0 & \text{otherwise} \end{cases}$

Figure 5.5: Visibility matrix and its constituents

world, and that the underlying GIS can determine the visibility of objects from arbitrary positions.

In a first step, the system retrieves all world objects that are close to the potential positions. In order to reduce the number of objects, the methods used for object evaluation (see section 4.1.1) can be applied as well. The resulting set consist of *salient objects*, which are not necessarily landmarks or routemarks[7] but rather objects that are salient to a specific user, e. g. because they are familiar to her, correspond to her interests, or are standing out visually. For all objects of the resulting set S, we then determine whether or not it is visible, i. e. for all potential positions we check the visibility of each object. We then dispose of a *visibility matrix* $V(S,P)$ as shown in Figure 5.5 which is used to determine what objects to ask for next.

Since the user's reply to a question – whether or not she can see an object – should allow us to eliminate as many hypotheses as possible, we have to select those salient objects that best partition the set of the potential positions. An ideal example for such an item would be a salient object that is visible from exactly half of the potential positions. Normally, there is no such salient object, and we have instead to select the ones that partition the set of potential positions in two sets of roughly the same size, i. e. we are looking for the salient object s_k for which the following statement holds:

$$\forall i \in \{1,\dots,k-1,k+1,\dots,m\} : \left| \frac{n}{2} - \sum_{j=1}^{n} vis(s_i,p_j) \right| \leq \left| \frac{n}{2} - \sum_{j=1}^{n} vis(s_k,p_j) \right|$$

If more than one salient object meets this criterion we can either randomly select one, or recursively determine which salient object entails the lowest number of questions once its visibility is known. The later alternative yields a more informed choice (at the expense of higher computational costs), since we analyze in advance what questions will follow when the user either confirms visual contact with the current salient object or not. This approach can also be iterated after each reply by reevaluating the set of the remaining candidates in the same way (again at the

[7] although there it is very likely that landmarks and routemarks are salient to a specific user as well

expense of higher computational costs). However, neither approach can guarantee that the number of questions will be minimal as the reply to the current question is not known in advance.

It is important to keep the number of questions as small as possible when implementing this algorithm. In a mobile setting, many services rely on positional information (see section 4.3), and could potentially initiate an interaction to determine the current position. This could possibly result in the user being overwhelmed with a large number of questions that are not directly relatable to her current task, thus irritating her and reducing the overall acceptance of the system. In order to avoid this situation, inferred positional information should be stored alongside measured information in the position history so that immediately following queries for the user's current position do not trigger a repetition of the same interaction that just took place.

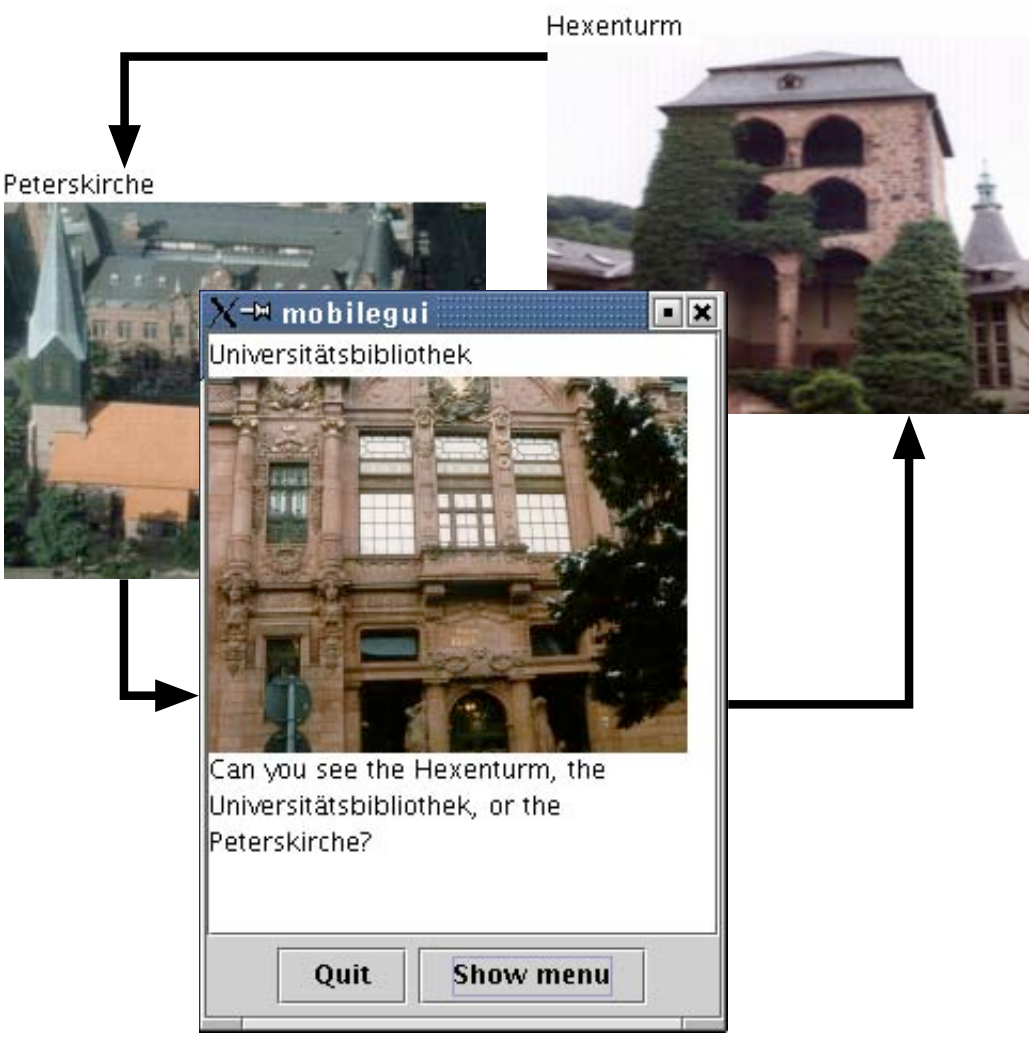


Figure 5.6: Combining several questions in a slide show

$$elim_s(k, V) = \begin{pmatrix} vis(s_1, p_1) & vis(s_1, p_2) & \dots & vis(s_1, p_n) \\ \vdots & \vdots & \ddots & \vdots \\ vis(s_{k-1}, p_1) & vis(s_{k-1}, p_2) & \dots & vis(s_{k-1}, p_n) \\ vis(s_{k+1}, p_1) & vis(s_{k+1}, p_2) & \dots & vis(s_{k+1}, p_n) \\ \vdots & \vdots & \ddots & \vdots \\ vis(s_m, p_1) & vis(s_m, p_2) & \dots & vis(s_m, p_n) \end{pmatrix}$$

$$elim_p(j, V) = \begin{pmatrix} vis(s_1, p_1) & \dots & vis(s_1, p_{j-1}) & vis(s_1, p_{j+1}) & \dots & vis(s_1, p_n) \\ vis(s_2, p_1) & \dots & vis(s_2, p_{j-1}) & vis(s_2, p_{j+1}) & \dots & vis(s_2, p_n) \\ \vdots & \vdots & \vdots & \ddots & \vdots & \\ vis(s_m, p_1) & \dots & vis(s_m, p_{j-1}) & vis(s_m, p_{j+1}) & \dots & vis(s_m, p_n) \end{pmatrix}$$

$$V' = elim_s(x, V_k) \text{ where } k = \sum_{j=0}^{n} vis(s_x, p_j)$$

$$V_l = \begin{cases} l = 0: & V \\ l \neq 0: & elim_p(min(j | j < columns(V_{l-1}) \wedge vis(s_x, p_j) = 0), V_{l-1}) \end{cases}$$

Figure 5.7: Elimination of false hypotheses in the visibility matrix

Combining several questions is another approach to reduce the number of interactions that are necessary before the user's position is known. Figure 5.6 shows a presentation generated for such a combined question. Pictures of the salient objects are shown in a slide show (accommodating the small screen size) together with the question. The user can now either reply by confirming visual contact with one or more salient objects, or she can inform the system that she does not see any of them. Either way, we can reduce the number of potential positions tremendously as the reply provides us with visibility information for all salient objects included in the question. The combination of three salient objects has proven to be very effective in practical use: the salient object that divides the set of potential positions in two subsets of roughly the same size (see above), and the two salient objects that divide either subset best.

Once the user provides the system with visibility information (either for one salient object or for several), we can adjust the visibility matrix by eliminating all positions that contradict the user's reply. The row(s) corresponding to the salient object(s) included in the query can be removed as well since it is of no further use. Figure 5.7 shows the formal procedure of elimination for a single salient object question. (Multiple salient objects questions can be treated as a sequence of single salient object questions.) In order to determine the updated visibility matrix V' we need to eliminate all potential positions from the original matrix V from which salient object s_x is not visible. This is an iterative process that removes position p_j if $vis(s_x, p_j) = 0$ resulting in intermediate matrices V_l. Once all invisible positions have been eliminated, the current salient object s_x can be removed as well ($elim_s(x, V_k)$). If the user reports that s_x is invisible, the only differences are that we have to eliminate positions p_j if $vis(s_x, p_j) = 1$, and that $k = n - \sum_{j=0}^{n} vis(s_x, p_j)$. If the user provides information about multiple salient objects simultaneously we can apply the

same procedure to one salient object after the other. The process of interaction and elimination continues until one of the following conditions holds:

- **All hypotheses have been eliminated.**
  This implies that either the original set of potential positions was wrong, or that the user was unable to recognize a salient object, or has overlooked one or more salient objects.

- **There is only one hypothesis left in the visibility matrix.**
  We have successfully determined the user's current position, and the system can proceed with the task that requested positional information.

- **The remaining hypotheses can be merged into a single position.**
  This happens when the remaining salient objects do not allow for a reduction of uncertainty (e. g. they are visible from all positions), and if the remaining hypotheses are located close to each other.

- **The remaining hypotheses cannot be merged.**
  In this case, the remaining salient objects cannot be used to reduce the uncertainty, which of the remaining positions is most likely the true position of the user. In addition, the remaining positions are also located too far apart from each other to be merged into a single position.

The second and third case allow for the termination of the position determination, and enable the system to continue to work on the task that originally requested information about the user's current position. In the first and fourth case, however, the reduction of uncertainty has failed, and other means have to be employed to still provide the service the user has asked for. The following two sections describe two approaches that can be used to address this issue.

Figure 5.8 shows an overview over the entire algorithm that reduces the uncertainty, which position of a set of several candidates is most likely the actual position of the user. We first select the best divider, i. e. the salient object that partitions the matrix in a way that allows us to quickly reduce its size of the matrix once we know whether the object is visible. In order to limit the number of interactions, we select the best dividers for the resulting submatrices as well. Then, we generate the query for the user, which consists of a repeating slide show of labeled images of the selected salient objects. The reply of the user is used to reduce the matrix as described above. The algorithm terminates either when the user's position has been determined, or when we cannot identify it. In the latter case, we can apply the strategy described in the following section.

5.3.3 Exploration

Whenever the set of potential positions is too large to determine the user's real position with a short series of questions, we can rely on the fact that – at least in most urban environments – street signs are virtually omnipresent. The same is true, if we do not have any information at all about the user's current position, i. e. the set of potential positions is empty. When we ask the user for the name of the street, not only is it likely that she will know the answer but the range of possible replies is sufficiently restricted by the set of all the street names to make speech recognition a

```
select best dividers
    find salient object that best divides matrix
    find salient objects that best divide resulting submatrices
generate query for user
    retrieve images for salient objects
    show repeating slide show and question
evaluate the user's reply
    reduce matrix
    if matrix is empty
      exploration
    else if matrix has only one element
      return as user's position
    else if elements of matrix can be merged
      return as user's position
    else if matrix does not allow for further reduction
      exploration
    else
      repeat
```

Figure 5.8: The reduction algorithm: an overview

possibility.[8] Even if the user cannot reply right away, there is a substantial probability that a street sign is nearby. In that case she can either provide the answer after simply looking around, asking a passerby, or by moving a short distance to locate the next street sign. While the latter case requires the person using the system to relocate, it has the advantage that the user's position can be immediately determined afterwards: Since street signs are frequently installed at crossings, we can directly compute the position if we learn that the user is at the crossing of street A and B.

Even if the user is not at a crossing and can only provide us with the name of the street she is in, this information enables the system to fetch all streets with that name from the GIS. Hence, the approach presented here can cope with several streets of the same name – unlike the simple interactive mechanism realized in the LoL@ system (see 3.6): LoL@ relies on house numbers to identify the segment of a street, where the user is located, and could therefore not distinguish between several streets of the same name. From a mathematical perspective, a street still contains an infinite number of potential positions if the underlying model consists of geometrical shapes such as lines: all points on the line would have to be considered. However, depending on the available resources (see section 2.4) and on the precision of the position that is required by the current task, we can compute a discrete subset. Based on the street geometry, we can generate a set of potential positions by spreading points along the corresponding geometry (e. g. a line). These positions p_j should be located at a constant distance from their neighbors in case of a continuous shape, and include all end points or extreme points such as the corners of a rectangle.

[8] However, while the range of admissible replies is limited, spelling problems as well as the difficulty to recognize names may either call for clarification dialogs, or alternative input means such as multi-modal interfaces.

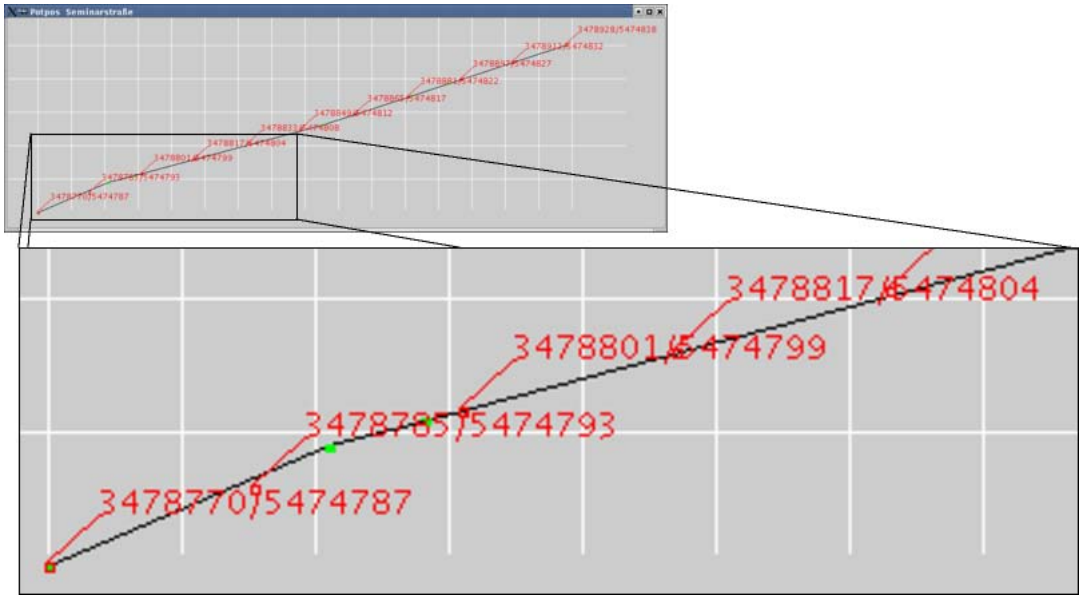


Figure 5.9: Generating potential positions (labeled with their world coordinates) along a street (screenshot)

Figure 5.9 shows a exemplary screenshot from the system presented in chapter 6. The black line is the geometric representation of the street “Seminarstraße”. Since the original request for positional information specified a precision of ten meters, potential positions are generated along the street at ten meter distance. In the figure, these are labeled with their corresponding world coordinates. Since streets are most often stored as network of nodes (crossings, topological change, etc.) and edges, we are usually faced with computing a number of evenly spaced out points that are located on a single (poly-)line. Once these have been determined, we are in a position to apply the techniques described in the previous sections on the resulting set of potential positions.

Figure 5.10 gives an overview over the algorithm for exploration. We first ask the user what street she is in and evaluate her reply. If she does not know we ask her to locate a street sign and start from the beginning. If the street name given by the user is unknown, we can either fail or repeat (e. g. to overcome speech recognition errors). Otherwise, we generate the visibility matrix by first selecting a set of salient objects from all objects near the street, and then generating discrete positions along the street. Finally, we fill the matrix by computing the visibility of every object from every position, and apply the reduction algorithm (shown in figure 5.8).

5.3.4 Adaptation

However, there may still be cases, where it is impossible or infeasible to determine the user’s position at the desired precision, or at all. This may happen, for example, if the user is unwilling or unable to provide the name of the street she is in, or if an interaction would take too much time and would cause the task requiring positional information to fail. While there are tasks that

```
ask for street name
evaluate the user's reply
    if user does not know
      ask user to locate street sign
      repeat
    else if street name unknown
      fail/repeat
    else
      continue
generate visibility matrix
    find all objects near street
    select a set of salient objects
    generate discrete positions
    compute visibility for every object and position
apply reduction algorithm
```

Figure 5.10: The exploration algorithm: an overview

will fail if no positional information is available or if it is less precise, often it is possible to still provide the corresponding service albeit at a reduced quality. The following section reviews the complex tasks that were introduced in section 4.3, and analyzes how they can be adapted to positional information of lesser quality. We then review a means for coping with imprecision and uncertainty at the language level, namely linguistic hedges. Finally, we introduce the concept of induced frames of reference that can help to overcome even the total lack of positional information.

Task relaxation

From a user's perspective, a mobile assistance system is not very useful, if it fails when faced with less than expected information quality. In section 2.6, we have seen that all measuring devices for positional information are prone to errors, and especially GPS readings may vary widely with weather conditions, the height of surrounding buildings, and alley width. Therefore, a system should at least provide its services at a reduced quality instead of failing if it is unable to deliver the high quality version (*graceful degradation*).

In case of *self localization* task, the user's goal is to learn about her current location in such a way that she is able to position herself within her current model of the world. Frequently, this is achieved by means of a you-are-here map [Richter, 2001], on which not only the user's current position is marked (by an arrow and/or a cross) but also familiar landmarks. When the precision of the positional information decreases, one way to compensate is to use other graphical means to mark the user's position such as circles that grow with the imprecision instead of crosses (see, for example, LoL@ and REAL in 3.6 and 3.7). An alternative approach that can also be combined with this consist of resorting to a higher level of granularity, e. g. by referring to or depicting city quarters instead of individual streets.

This approach can be applied to the general *localization* task as well, which does also allow for several further strategies: We can generate replies that rely on an allocentric frame of reference instead of an egocentric one, thereby becoming more independent of positional information. In addition, we can compensate the lack of directional information by resorting to distal or topological relations (such as `near` or `in`), which do not require this information at all.

The *identification* task is highly dependent on precise positional information since the user's viewing direction and location are key factors in the process of determining the target object. Therefore, if we dispose of no information at all, we cannot provide this service. However, if we have at least some information about the user's current location, we can introduce an additional interaction where we present the user with a list of all potentially interesting objects in her vicinity and let her select the one that she meant. A corresponding presentation could, for example, consist of a slideshow containing pictures of all objects, or a map, where these are highlighted. If we also dispose of some information of the user's current viewing direction, we can apply the means described in section 5.3.2 and ask the user to disambiguate between a small set of potential targets.

Unless the starting or the end point is the user's current location, complete *route instructions* do not require positional information. Their incremental counterpart, however, rely heavily on it as each increment has to be timed to be communicated at the right location. Additionally, the viewing direction is crucial for the generation of turn instructions. While the way in which route instructions are presented (see section 4.2.3) as well as induced frames of references (see section 5.3.4) can help to compensate imprecision to some degree, we have to state that incremental route instructions are infeasible when positional information is entirely unavailable. Even the techniques described in the previous sections are limited in their application, since going through a (short) conversation every few minutes is highly annoying to a user who only wants to be guided from one place to another. What we can do, however, is to substitute *complete* route instructions for their incremental counterparts. In doing so, we gain a high degree of independence from the availability of positional information while sacrificing the intuitiveness of instructions that are tailored to the user's current location and precisely timed to her progress along the route.

Geo-encoding of relational statements such as `left-of castle` rely on the underlying frame of reference: the less of its components (origin, orientation, handedness) depend on positional information, the more independent the geo-encoding process is of it. While the total lack of information of a certain type will keep this task from completing (e. g. no location implies failure, no directional information excludes angular relations), we can account for imprecision in several ways: If the current heading or viewing direction is only roughly known, we can resort to a coarser granularity for computing regions corresponding to angular relations. Instead of using directed cones, we can apply a half-plane model (cf. [Blocher, 1999]. In case of distal relations and imprecise knowledge about the user's current location, we can enlarge the region that results from geo-encoding a relational statement by the size of the error.

The same strategies applies for the *acquisition* task, which relies on geo-encoding. Since it also incorporates object identification, the corresponding methods are applicable as well. In addition to the approaches which we described in this section, it is possible for every task to communicate uncertainty and imprecision verbally using linguistic hedges, which are reviewed shortly in the following section. Letting the user know about the systems uncertainty and about the imprecision of the currently available data enables her to better understand the meaning of the system's output.

Linguistic hedges

An alternative approach to coping with imprecision is to bring it to the user's attention, and thereby enable her to take it into account when planning her actions. Linguistic hedges are one way to achieve this on the verbal level. This group of linguistic modifiers is very diverse, and has been used widely – especially in fuzzy logic and fuzzy reasoning (cf. [Zadeh, 1965, Zadeh, 1972, Lakoff, 1973]. Since one concern in this field lies in expressing the vagueness and imprecision, which very often is addressed by the use of linguistic hedges, the same should be true in case of communicating imprecision in terms of positional information. But before we can analyze the corresponding possibilities, we need to define more precisely what exactly is a linguistic hedges.

> Als *linguistische Hecke* (kurz: Hecke, engl.: *linguistic hedge*) bezeichnen wir sprachliche Einheiten, die Prädikationen nach Grad oder Hinsicht ihres Zutreffens modifizieren und als Operatoren interpretiert werden können, welche die Vagheit des sprachlichen Konzeptes, auf das sie angewendet werden, verstärken oder abschwächen.
>
> By the term *linguistic hedge* we refer to linguistic entities that modify predications in terms of the degree or aspect of their application and that can be interpreted as operators, which either increase or decrease the vagueness of the linguistic concept that they are applied to.
>
> [Wahlster, 1977]

In [Kray, 1998], we thoroughly analyzed linguistic hedges and we proposed a model based on the precision of the predicates, which are modified by linguistic hedges. We presented a hierarchical classification of German linguistic hedges that certainly does not translate directly into English or any other language. However, there are several subclasses that have a correspondence in English, namely Gradierungspartikeln (grading particles) and Steigerungspartikeln (comparison particles). The first group includes terms such as "probably", "possibly", or "it is rather unlikely that", and can serve the purpose of expressing uncertainty, e.g. when the system has only little evidence that its positional hypothesis really is the current position. The second one mainly serves the purpose of quantification or scaling, e. g. "very" or "somewhat".

In the context of situated interaction on spatial topics, we can apply hedges in several ways. Firstly, we can communicate the confidence the system has in the current positional information in the above mentioned way. Secondly, the precision of the information can be verbalized using hedges such as "roughly" or "exactly", which enables the user to better judge the output of the system. Thirdly, linguistic hedges such as the last two examples can also help to convert quantitative to qualitative information (which can be beneficial in case the quantitative information is imprecise) and to grade qualitative relations, e. g. "A is *a little bit* to the left B".

Induced frames of reference

Induced frames of reference enable the system to compensate for the lack of any kind of positional information: When the current viewing direction is unknown, the system can select a direction, which is best suited at the moment, and then precede the actual output with a turn instruction such

as “If you turn towards the fountain, ...”. Even if the viewing direction is known precisely, there might be another one, which is preferable in the current situation, e. g. in case of a localization and the use of directional relations. Here, an induced frame of reference can help to improve the output quality by establishing the superior frame of reference using a precedent statement such as “If you turn slightly left, ...”.

If the current location of the user is unknown, the system can analogously select an origin, which best suits the actual purpose, and instruct the user to (either mentally or physically) relocate, e. g. by generating instructions such as “If you stood on the corn market, ...”. The same can be done in case of imprecise positional information by inducing precision through statements such as “If you stand exactly in front of the church, ...”. Note that this does not require the user to perform a (physical or mental) reorientation, as only the origin of the frame of reference is affected. However, the total lack of positional information may require the combination of both relocation and reorientation (e. g. “If you stood on the corn market facing the church, ...”). Theoretically, this combination can help to address all possible situations from very imprecise information on one or more constituents to the total absence of any information. In practice, it is not always feasible. For example, incremental route instructions are of little help if not tailored to the exact position of the user. Therefore, inducing location and orientation prior to giving the corresponding instruction is infeasible.

5.3.5 Modeling the positioning task

The process of determining the user’s current position in a way we described in the previous sections really is a further complex tasks that we can model in terms of the basic processes and complex tasks introduced in 4.1 and 4.3. Figure 5.11 shows the corresponding interaction diagram, which does not only illustrate the interaction with other processes but also those with further positioning processes.

In the figure, the positioning task ‘positioning’ is triggered by the user asking “Where am I?”. This request is then translated into a preverbal message (PVM). However, this is only one possible way to initiate the positioning task – other processes or tasks can request the user’s current position as well, e. g. in the context of a localization task. Once a query starts the positioning task, the positioning task first requests sensor readings from the component supervising the sensor measuring information related to position. If these readings are sufficient to answer the position request (in terms of availability and precision), a PVM localizing the user is requested from the complex task ‘localize’, which is then used to generate an appropriate reply – for example, a personalized you-are-here map along with a verbal description of the user’s current location. The corresponding procedure is depicted in box ‘A’ in figure 5.11.

In case the sensor data does not provide sufficient precision (or is unavailable), the ‘position’ task request all recent entries in the position history. The corresponding component ‘pos-hist’ keeps track of positional information over time, and can thus provide the most recent information. If this information meets the criteria specified in the original query, ‘position’ can generate an answer in the same way as described above. Box ‘B’ in figure 5.11 illustrates this portion of the positioning task.

However, it is possible that the entries in the position history do not allow for the answering of the original request. In that case, the interactive processes described in this section have to be

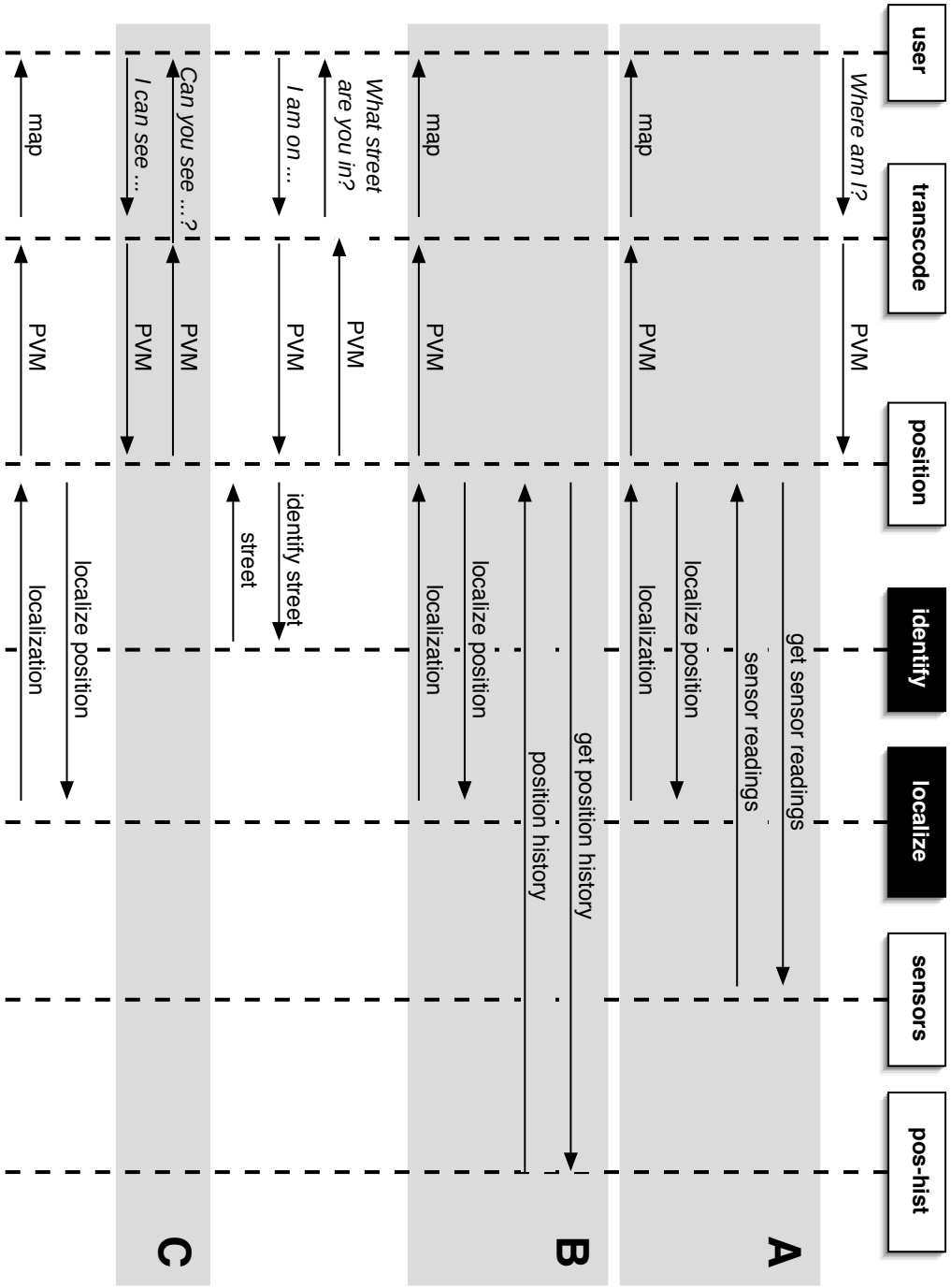


Figure 5.11: Positioning: Interaction of basic processes

applied. This can involve, for example, asking the user what street she is in. The user's reply is then translated into a PVM, from which 'position' extracts the information specifying the street. This information is included in a request for the complex task 'identify', which in turn identifies the most likely street and returns it to the positioning task. In the following, several interaction steps take place, where 'position' requests information from the user which is then used to determine her current position. These steps – shown in Box 'C' – are repeated until 'position' has inferred the current position. In analogy to the two previous cases, this information is then used to generate the appropriate reply. In section 6.5 the procedure described here is used to design a generic architecture for handling positional information.

5.3.6 Application to complex tasks

In the previous sections, we presented several strategies to address the problem of varying quality of positional information. Most of these have been implemented and tested in a prototypical system (see chapter 6), where they proved very helpful in coping with the total or partial lack of positional information as well as with changing precision thereof. Although the entire system (SISTO on Deep Map) has not yet been empirically evaluated, in a first test, we were able, for example, to determine the user's current position at a precision of ten meters in a street of approximately 170 meters length without any sensor information in only three interactions. Table 5.1 lists all the strategies that we introduced in this chapter, and their applicability in the context of complex tasks presented previously. In the table, entries in brackets apply only in certain cases (see below), $\oplus$ and $\ominus$ signal the general applicability of the corresponding combination of a complex task and an adaptation strategy, and $\bigcirc$ identifies partial applicability.

Since some points depicted in table 5.1 have already been discussed (such as task-specific relaxation strategies in 5.3.4), we will focus on the most interesting cases, respectively on those combinations that are not feasible or less useful. In the context of the identification task, for example, we heavily rely on very precise information about the user's position, especially her viewing direction. Therefore, it makes little sense to infer the current position as the resulting positional information is not sufficiently precise to allow for a reliable identification of (nearby) objects. Analogously, linguistic hedges cannot be used as the identification task calls for a single, crisp target object. These considerations apply also to the complex tasks of data collection and map interaction as these mainly rely on the identification task.

Unlike these tasks, the localization task generally allows for the application of all strategies. The only exception occurs in the case of self-localization, where induced frames of reference are not feasible: Since the user does not know her current position, she is unable to perform the corresponding actions. Inducing a frame of reference is only partially applicable in the complex task of giving (incremental) directions as well. In the case of incremental route instructions only turning instructions are suitable, as a relocation would lead to instructions from outside of the route perspective and confuse the user. Additionally, interactions such as the reduction of uncertainty or the exploration should not be used when guiding the user to a target location since the guidance task is an ongoing process. Repeated interactions at the end of each segment could seriously confuse the user, and could even lead to the total failure of the guidance process. Imagine, for example, a car driver approaching the end of a route segment at which point the system engages in a dialog to determine her current position. By the end of the dialog, the user is past the crossing where she

Task	Inference	Interaction	Adaptation
Identification	⊖	⊕	⊕ task relaxation induced frame of reference
Localization	⊕	⊕	⊕ task relaxation linguistic hedges (induced frame of reference)
Directions	⊕	⊖	⊕ task relaxation linguistic hedges (induced frame of reference)
Geo-encoding	⊕	◯	⊕ task relaxation
Data collection	⊖	⊕	⊕ task relaxation induced frame of reference
Map interaction	⊖	⊕	⊕ task relaxation induced frame of reference

Table 5.1: Complex tasks and positional information: $\oplus, \bigcirc, \ominus$ indicate whether a strategy applies well, somewhat, or not at all; entries in brackets apply only in some cases.

would have had to turn. However, induced frames of reference can be used in conjunction with complete route directions, i. e. the start location can be localized using any combination of reorientation and re-localization. Yet another concern limits the applicability of interaction strategies in the context of geo-encoding task. Depending on the task that initiated geo-encoding interacting with the user may either be feasible or not. If it was directly triggered by a user query – e. g. she employed a relational spatial localization – the user will probably not mind if the system first engages in a dialog before replying to her original request. However, if the geo-encoding was triggered by another process that was not initiated by the user, or if the user is completely unaware of the process (e. g. a process that is precomputing replies to potential future requests) then an interaction will most likely disturb the user.

5.4 Summary

In this chapter, we presented a number of strategies that enable a (mobile) system to adapt to common issues of real-world usage: lack of information, resource restrictions, and positional information of varying quality and availability. We first analyzed on an abstract level, in which ways a system can address the lack of situational information. The resulting categories consist of either ignoring missing information, or accessing alternative sources, or using standard values. Furthermore, the missing information can be inferred, or the underlying computations can be adapted. Finally, in an interactive system we can request information from the user. Based on this analysis, we showed how to apply these strategies to the complex tasks described in 4.3.

A second major issue in the context of real-world use is the handling of resource restrictions. In section 2.4 we review two different strategies to adapt to varying resource situations: adaptation of the presentation and the underlying computation. We first presented an analysis of how information in the context of complex tasks related to space can be presented to the user, and how these ways differ in terms of resource consumption. Based on this analysis, the selection of the presentation type can help to adapt to the current resource restrictions. Then, we pointed out how the rich information contained in a preverbal message enables the generating component to select *what* to present. This, in turn, has an impact on the resulting presentation, and therefore allows for further adaptation. Finally, we proposed several means to change the way in which task-related computations are performed, ranging from the gradual adjustment of the contents of the PVM to exploiting the inherent properties of multi-agent systems.

The third key factor addressed in this chapter was the availability and quality of positional information. We proposed a comprehensive approach to deal with this issue based on inference, interaction, and adaptation. In terms of inferring the user's current position we presented a knowledge-based dead reckoning mechanism that improves on the standard algorithm by considering situational factors as well as information from the world model. A key feature of the adaptation strategies that we introduced here lies in the cooperation with the user: instead of solely relying on sensor data and/or inference, we designed an interactive process that requests information from the user if no other sources are available. This process includes simple requests to select among a number of alternatives but also incorporates a sophisticated exploration algorithm that determines the user's current position through the visibility of world objects. In addition, we reviewed three strategies to adapt in case the position cannot be determined at the desired precision.

Finally, we showed how to model the entire process of adaptively determining the user's position in terms of the complex tasks and basic processes presented in chapter 4. The chapter concludes on an evaluation what strategies can be used in the context of which complex tasks, and what restrictions apply.

6 Implementation

There are several ways to validate a model such as the one presented in chapter 4: One alternative consists of empirically validating predictions derived from the model, which applies especially in the case of models that claim to mimic human behavior or reasoning. Depending on the degree of mathematical rigor that was used in defining the model, it may be possible to formally prove the correctness of the model in relation to a set of axioms. The way we selected for our model was to build a system that realizes the theories underlying our model. This has the advantage of proving the practical relevance of our approach while allowing for a later empirical evaluation of selected parts of the system *in a real-world setting.* While such an evaluation was outside the scope of this work, the implementation we describe in this chapter can serve as a starting point for such a study (see also 7.2.1).

A further reason why we favored an implementation over a purely empirical or mathematical validation was the breadth of the model. Instead of just analyzing a small subset of situated interaction on spatial topics, our goal was to design a model that covers most tasks in this realm. Consequently, a complete empirical evaluation would require a large number of studies on specific parts of the model which is beyond the scope of this thesis. Instead, the implementation enables us to validate the usefulness of the model in practical use with untrained users.

In this chapter, we will first review the general requirements for a system implementing a model such as the one we proposed in the previous chapter (6.1). Since multi-agent systems are a convenient means to address some of these, we will give a short overview over the corresponding terms and concepts in 6.2. Then, we will present the two host systems, SmartKom and Deep Map, which served as testbeds for the prototypical implementation (6.3). The agent implementing our model, SISTO, is described in the following section (6.4). In order to realize the adaptation strategies concerning positioning we then propose a generic architecture for handling positional information in section 6.5. An example journey through the city of Heidelberg illustrates the capabilities of prototypical system (6.6) and concludes the chapter.

6.1 Requirements

Cheverst et al. [Cheverst et al., 2000a, Cheverst et al., 2000c] have identified several key requirements for designing a mobile and interactive system. Through an expert walkthrough and an empirical study with tourists visiting the city of Lancaster they were able to support their claim that these requirements do indeed apply (at least in the context of a mobile tourist guide). According to their findings, these include:

- **flexibility**
  The users should be able to choose the way in which they access the information and services provided by the system. For example, some people visiting a city prefer guided tours while others like to explore it on their own.

- **context-sensitive information**
  The information presented to the user should be tailored to their current situation. In section 2.2, we list several relevant factors, and in 4.1.1, we discuss their impact on various tasks.

- **support for dynamic information**
  Information should be made available to the user whenever the situation deems this to be appropriate.

- **support for interactive services**
  The system should provide means to integrate external services such as online booking of accommodations.

In addition to these requirements, we have identified several further points, which we can either derive from the model presented in chapter 4, or the adaptation strategies proposed in chapter 5, or from more general considerations in the context of mobile assistance. More specifically, we can add the following requirements:

- **correspondence to model**
  The implementation should closely match the design and structure of the model in order to support its validation and to harvest the advantages of the model (such as extensibility and flexibility). Especially, the explicit interaction between various processes and tasks cannot only help to facilitate the implementation but also benefits the clarity of its design.

- **reusability**
  An implementation that realizes a model with a broad applicability should allow for easy reuse in order to validate the approach in other scenarios and systems and to avoid unnecessary re-implementations.

- **smart positioning**
  As we have seen in the previous chapters, positional information is a key factor determining the user's current situation. An implementation should take this into account and support the application of the adaptation strategies we described in 5.3.

On a more technical level, it is desirable to rely on standards when implementing a system based on the previously presented model (where this is feasible) in order to avoid unnecessary re-specification and to facilitate reuse. Furthermore, it can be advantageous (especially in a mobile setting) to provide architectural flexibility, i. e. allowing for a client-server architecture as well as for an integrated realization. Ideally, a system should support a gradual and flexible shifting of various (sub)components from local to remote execution.

6.2 Multi-agent systems

Multi-agent systems and agent-based computing are fairly recent additions to the field of computer science but have been rapidly gaining momentum in recent years [Weiss, 1999]. The underlying paradigm of autonomous entities interacting with its environment and/or siblings (see below for a more detailed definition) has proven beneficial in a variety of scenarios such as social simulations (see, for example, [JASSS, 2003]), electronic commerce [Preist et al., 2001], or software engineering [Jennings, 2000]. In a way, one can see agents as being the successors of 'objects' and object-oriented software engineering [Booch, 1994] as they do not only encapsulate some data and the corresponding methods to handle it but also have control over their own actions [Jennings, 1999]. This is a feature that we heavily relied on when implementing the model presented in the previous chapters. But before we review the advantages (and drawbacks) of multi-agent systems, we first need to define some basic terms in the following section.

6.2.1 Terms and Definitions

Due to the number of disciplines involved in agent research (computer science, artificial intelligence, business administration, social sciences, etc.), there are not only many possible applications but also several competing definitions of what exactly constitutes an agent. Some researchers define an agent in terms of its beliefs, desires, and intentions [Bratman et al., 1988], whereas other simply see it as a 'more autonomous object' (see below). A definition that captures well how the term will be used in this thesis is the following:

> an agent is an encapsulated computer system that is situated in some environment, and that is capable of flexible, autonomous action in that environment in order to meet its design objectives [Wooldridge, 1997]

Consequently, a *multi-agent system* is a team of several such agents that interact in order to fulfill the task(s) they have been designed for. A further important concept in this context is the *holon*. A holon behaves like any other agent in a multi-agent system but internally consists of a team of agents or further holons [Fischer, 1999]. This organization is a very efficient means to address increasing complexity.

Since there are some similarities between object- and agent oriented approaches, it is important to point out relevant differences. While both are based on information hiding and strictly defined interfaces (or interactions), Jennings [Jennings, 1999] lists four key points of discrepancy:

- Unlike agents, objects are passive in nature: they only react to incoming messages/requests.
- Objects have no choice of what to do: when a certain method is called, they perform the corresponding action, whereas agents can decide autonomously what to do upon reception of a message.
- Object-orientation does not support well the modeling of complex systems since individual objects and method invocation are too fine-grained to describe the interactions taking place.
- Organizational relationships are not easily specified and managed using object-orientation.

These arguments already hint at some of the properties that set multi-agent systems apart from other architectures such as client-server approaches or monolithic implementations. The following section will discuss these points in more detail.

6.2.2 Properties

Agent-based software engineering is a rather new discipline, which promises to cure some problems of 'traditional' software engineering (but unfortunately also creates a few new ones – see, for example, [Jennings, 2000, Wooldridge and Jennings, 1998]). In this section, we will focus on those properties of agent-based software engineering and multi-agent systems that are most relevant in the context of the implementation presented in chapter 6.

According to Jennings [Jennings, 1999], the three essential concepts of agent-based computing are agents, high level interaction, and organizational relationships. He cites Booch [Booch, 1994] for identifying three key techniques – namely *decomposition*, *abstraction*, and *organization*[1] – where these concepts help to improve on approaches such as object-orientation. Decomposition is the most basic technique for addressing large problems by dividing them into smaller, mostly independent sub-problems that are more manageable. Abstraction is the process of isolating relevant aspects of a problem in order to generate a simplified model of the system while disregarding other aspects. Jennings defines organization as "the process of identifying and managing interrelationships between the various problem solving components" [Jennings, 1999].

When decomposing a larger problem and then assigning the parts of it to different problem-solving components, agent-oriented software engineering offers the benefit of postponing the decision about interaction to run-time. This is especially advantageous in very complex and highly interactive systems – such as the one realizing the model and adaptation strategies presented in this thesis – since it is often extremely difficult and sometimes impossible to know a priori, which component is interacting with which one at any given point of time. In addition, the interaction is explicit, i. e. it takes place in a high level language such as KQML [Mayfield et al., 1996] or ACL [The Foundation for Intelligent Physical Agents, 2002], and is therefore more easily accessible to the human reader than method invocation. In terms of finding the right level of abstraction for the issues we address in this thesis, multi-agent systems are a very natural match for implementing the approaches presented in chapter 4 and 5, as we will see in the following sections.

Finally, an agent-based approach is very flexible concerning the organization of a complex system. Not only do most agent platforms provide means to dynamically discover agents that provide a certain service (e. g. FIPA [The Foundation for Intelligent Physical Agents, 2002] compliant platforms), but they also inherently support distributed computing. The first feature improves robustness as the failure of an agent can be detected and an alternative service provider can be selected to replace the failing agent. In conjunction with the second point, it also facilitates resource-adaptation because multiple agents can work on a problem in parallel, and/or move to a remote site to overcome resource restrictions. Furthermore, teams of agents can dynamically form to solve a given problem, and the inclusion of new agents is more readily achieved as in purely object-oriented systems.[2]

[1] Booch used the term 'hierarchy', which is subsumed by the term 'organization' according to Jennings.

[2] This is also a result of non-predefined interaction patterns.

6.3 Technical context

One important motivation for the development of the concepts, models, and strategies presented in the previous chapters was the intention to improve the interaction between a system providing services related to space and its human user. Within the research concerning intelligent user interfaces, it is a declared goal to relieve the person that uses a computer system of the burden to adapt to the concepts, behavior, and shortcomings of the system. Rather, it is upon the computer to try to adapt to user's abilities and preferences [Maybury and Wahlster, 1998]. The implementation of the ideas developed in the previous chapters was put to test in two real world systems in order to analyze whether it lives up to this ideal.

The following sections shortly introduce the host systems for the components that we implemented in the context of this work. In section 6.3.1, we introduce the Deep Map system, a mobile tourist guide for the city of Heidelberg, Germany. In section 6.3.2, we shortly present the SmartKom system, which is a large research effort to investigate future communication paradigms and devices.

6.3.1 Deep Map: a mobile tourist guide

The Deep Map system is the result of a joint research project hosted by the European Media Laboratory at Heidelberg, Germany [Malaka and Zipf, 2000]. Several German research institutes collaborated to develop a system that provides a tourist visiting Heidelberg with an easy-to-use system, which supports her in several ways during her visit. The project mainly focuses on simplifying human-computer interaction, e. g. by allowing for natural language input through a unified interface. This interface hides the complex services, which are accessed to fulfill the user's requests. A second major goal of Deep Map consists of creating a rich and detailed model of the domain. In order to achieve this, a high resolution map of Heidelberg was combined with 3D-data, historical information and a database containing further information such as names, and object properties (e. g. which roads are one way streets, or what businesses reside in a building).

The Deep Map system provides several services that tourists typically ask for. This includes incremental and complete route instructions, which are tailored to the specific user and context. Furthermore, the system is able to dynamically create maps of arbitrary regions in the Heidelberg area, and it allows for the manipulation of these maps (e. g., zooming, or panning). The identification of objects in the vicinity of the tourist is another service provided by Deep Map. It is also possible to request further information on arbitrary objects, and to learn about where they are located. The system can use relational localizations, maps, and further means to inform the tourist about her current position or the location of external objects. Finally, Deep Map supports the person in charge of maintaining the databases and GIS incorporated in the system by providing a special mode for the collection of new data in field. The user can access all these services using spoken natural language, or a graphical user interface optimized for the use on small screens.

Deep Map is realized as a multi-agent system that is organized in three conceptual layers: The *interface layer* comprises all agents, which interact more or less directly with the user. On the *cognition layer* reside all agents that are concerned with fulfilling the user's request. These agents rely on the services provided on the *knowledge layer* (see Figure 6.1).

The agents on the interface layer analyze the input of the user, and present the output of the

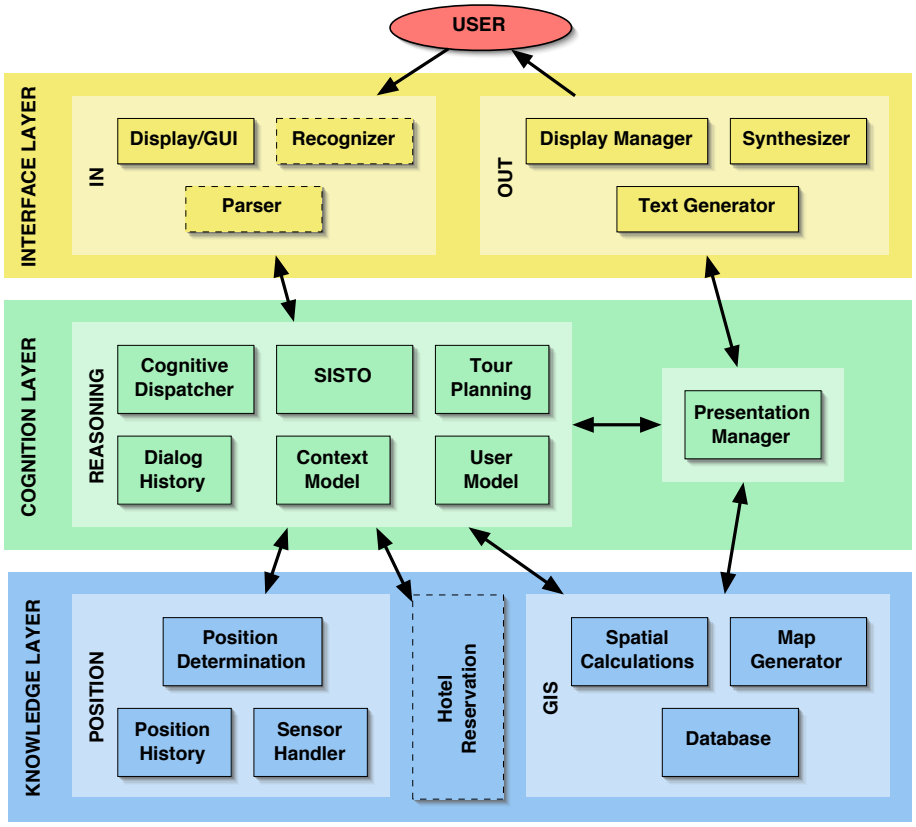


Figure 6.1: General architecture of the Deep Map system.

system to the user. In each group, three agents cooperate to realize these services. Since the input (as well as the output) can be multi-modal including (graphical) gestures, spoken language and written text, there are three components that convert the raw input into a higher internal representation. The *Display/GUI agent* captures pointing gestures, e. g. clicking with a pen on a touch screen, as well as textual input such as typed queries. The later one is passed to the *Parser agent*, which transforms natural language to a language independent representation format (see Section 6.4.2). The *Recognizer agent* handles spoken input by passing its hypotheses about what the user said to the Parser agent, which then performs the same transformation as it does for original textual input.[3] On the output side, there are three analogous agents. The *Text Generator agent* translates the semantic description of the reply utterance into a natural language phrase or text. The *Synthesizer agent* generates spoken utterances from textual data such as the results produced by the text generator. The *Display Manager* coordinates the coherent presentation of various the various parts. In addition to text and audio, it handles various types of graphical output, e. g. buttons, pictures, or maps.

[3]The recognizer and the parser have not yet been adapted to the most recent version of the underlying infrastructure. Therefore, input is currently only possible through a text-based interface.

On the cognition layer, several agents cooperate in order to determine the intention of the user's query/action, and to generate an appropriate reply. Once an an answer has been computed, the *Presentation Manager agent* generates a plan for the presentation of the corresponding content. Within the team of agents that are in charge of analyzing the user's queries and generating replies, the *Cognitive Dispatcher agent* coordinates the reasoning process. It analyzes the output of the agents on the interface layer that transform raw input to an internal representation. Based on the information contained therein the Cognitive Dispatcher determines the agent(s) that are most likely to be able to compute an appropriate answer. If several agents are needed to process the user's query, the Cognitive Dispatcher extracts suitable sub-queries and passes them to the appropriate agent. It collects the results and combines them into a coherent reply that is then sent to the Presentation Manager.

There are two main agents that provide the core services of Deep Map. The *Tour Planning agent* does not only compute routes through the Heidelberg area but also supports the user in tailoring a sightseeing tour to her need. The *SISTO* agent handles various tasks ranging from object identification to map control. In section 6.4 we will present a detailed description of this key component. In order to provide their services, SISTO and the Tour Planner rely on the information that is handled by the three auxiliary agents residing on the cognition layer. The *Dialog History agent* stores information about the interaction between the human user and the system, which is not restricted to their respective utterances, and can include arbitrary information that an agent wishes to associate with a given turn. The *Context Model agent* maintains a model of the user's current context, e. g. the current means of transportation, or actual weather conditions. Accordingly, the *User Model agent* stores information about the user.

A typical query will pass through the Cognitive Dispatcher to several reasoning agents, which in turn will request information from their siblings and/or the agents residing on the knowledge layer. Once an appropriate reply has been computed, the Cognitive Dispatcher passes it to the *Presentation Manager agent* that is in charge of deciding how to present it to the user. The Presentation Manager will then generate a presentation plan – possibly querying agents on the knowledge layer in the process – and assign the corresponding presentation realization tasks to the appropriate agents on the interface layer. Once the interface agents return their respective results, the Presentation Manager starts to communicate the entire presentation to the user.

On the knowledge layer, there are currently two main groups: agents in charge of handling the user's current position and several agents providing services around a Geographical Information System (GIS).[4] The first group consists of three members: The *Position Determination agent* serves as a proxy to 'external' agents asking for the user's current position, and it coordinates the effort to determine the position. The *Position History agent* keeps track of previous positions by storing all replies returned by the Position Determination agent and by periodically fetching measured positional data from the Sensor Handler. The *Sensor Handler* is connected to various sensors, i. e. a GPS receiver and an electronic compass, and provides the respective measurements to the agent community. Prior to being published, the readings are transformed from the raw data to a standard format that the other agents can understand. (In section 5.3, we give a detailed description of concepts underlying position determination, and in section 6.5, we present the corresponding implementation.)

[4] Previous versions of Deep Map also included a Hotel Reservation as a proof of concept.

The second group of agents on the knowledge layer consists of three agents that provide most of the information that the user wants to access. Once the agents on the cognition layer have determined what exactly the user wants to know, the GIS agents retrieve the desired information or generate it from the data stored in various databases. The *Spatial Calculations agent* performs basic geometric operations on geographical data such as searching a region for objects of a specific type or returning the outline of an object. The *Map Generator* dynamically generates maps from the data stored in the GIS, and allows for a wide range of parameters such as region boundaries, highlighting scheme, screen resolution, etc. The *Database agent* provides various information on arbitrary objects, e. g. the historical background, images, or textual annotations.

The entire system relies on a resource-aware agent infrastructure called *RAJA* (Resource-Aware Java Agent infrastructure) [Ding et al., 2001], which enables the distribution of agents over several platforms/computers. Furthermore, RAJA is built on top of a FIPA-compliant platform and therefore adherence to the FIPA standard [The Foundation for Intelligent Physical Agents, 2002]. Since Deep Map is entirely written in Java (except for some parts of the GIS and the Recognizer), it is basically platform independent.

6.3.2 SmartKom: (mobile) multi-modal interaction

Smartkom [Wahlster et al., 2001, Wahlster, 2002] is a large distributed research project lead by the German Research Center for Artificial Intelligence (DFKI). Its project partners stem from industry as well as from academia. The development process is organized around the iterative refinement of prototypes that demonstrate an increasing number of system features. A major scientific goal within the Smartkom project is the design of new computational methods for the seamless integration and mutual disambiguation of multi-modal interaction on the semantic and pragmatic level. This orientation is motivated by the idea to exploit the richness of 'natural' human interaction, which does not only include speech but also gestures and mimics.

In order to capture all this information and process it in an intelligent way, the system relies on a modular architecture based on the Verbmobil testbed [Wahlster, 2000]. Figure 6.2 shows a screenshot of the control interface to the SmartKom system. There are various components that handle tasks such as gesture recognition, language generation or audio recording. They are organized around multiple blackboards that store information related to a specific topics, e. g. speech recognition or presentation planning. All these modules are needed in order to realize multi-modal dialogue in three different scenarios:

- **SmartKom-Mobile**
  In this scenario, multi-modal interaction in a car and for a pedestrian user are investigated, e. g. in the context of incremental directions.

- **SmartKom-Home/Office**
  In this setting, the use of the system at home or at work is addressed, e. g. with consumer electronics or an electronic program guide (EPG)

- **SmartKom-Public**
  This scenario is aimed at investigating multi-modal interaction with a public system (similar to a phone booth), including services such as ticket reservation, phone calls, or email access.

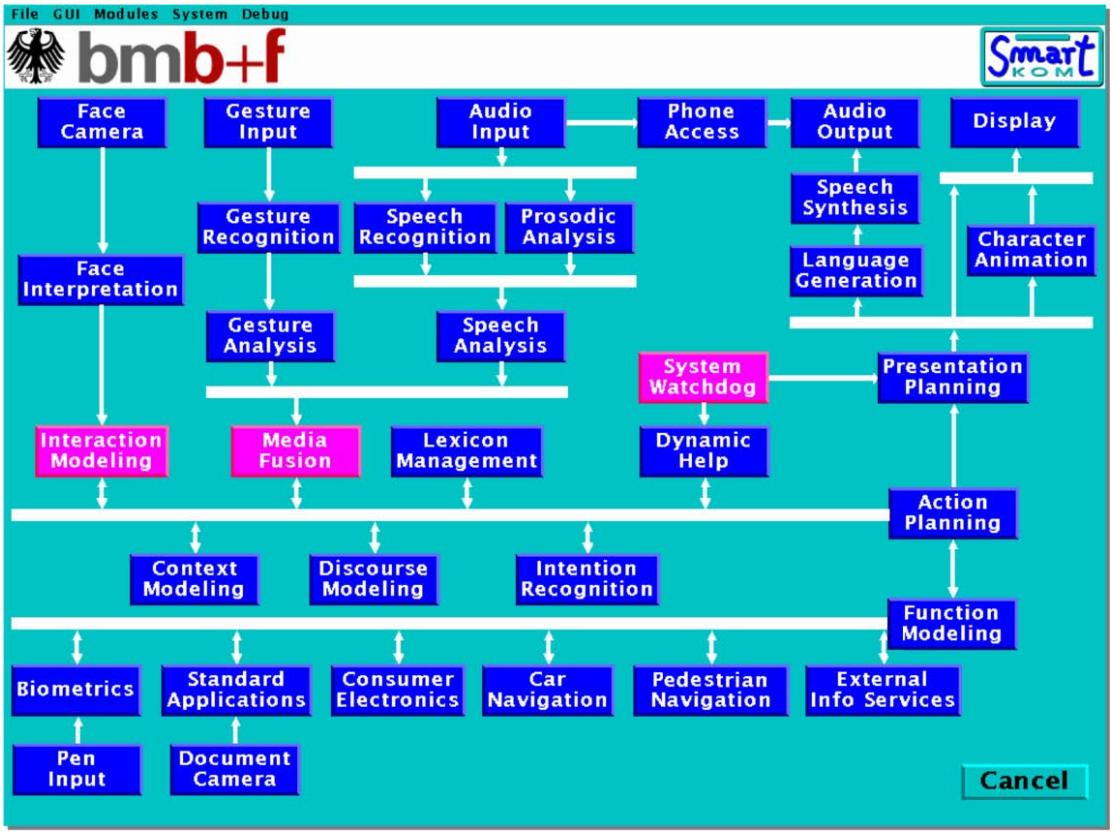
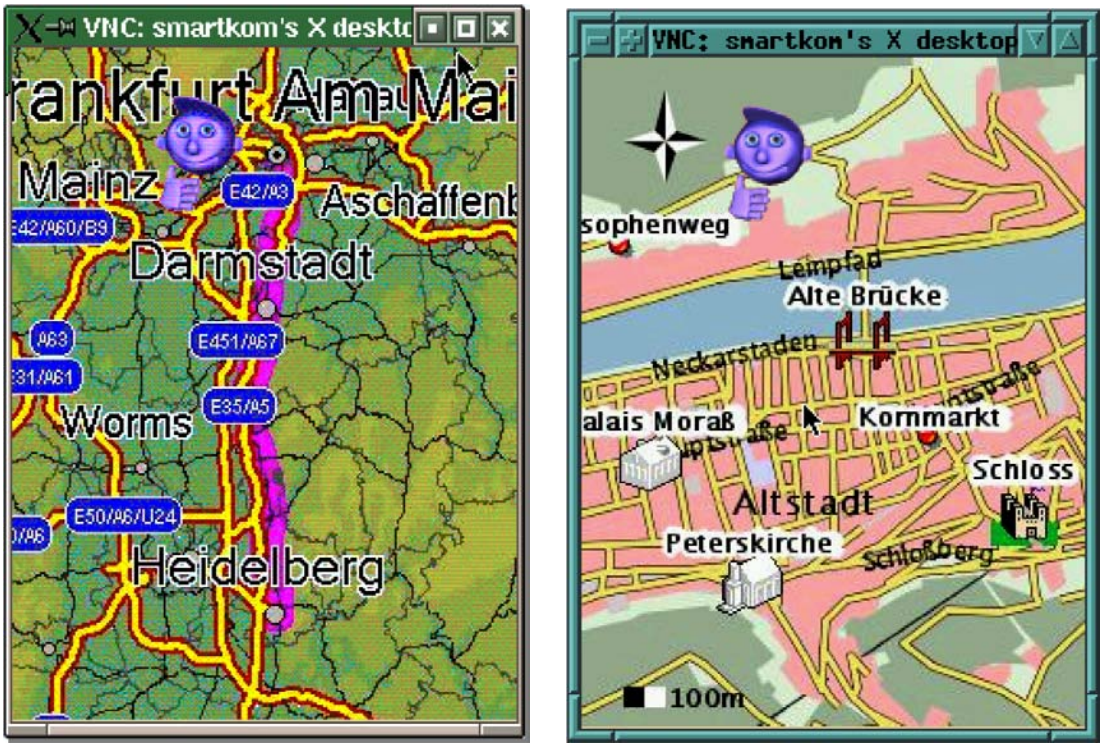


Figure 6.2: Architecture of the Smartkom system (screenshot of the control interface).

SISTO is one of the key components of the pedestrian navigation (shown in the lower right part of figure 6.2). This component is mainly used within the SmartKom-Mobile scenario, where it provides the user with incremental directions. A distinguishing feature of SmartKom in this context consists of the seamless transition from car navigation (shown in figure 6.3(a)) to pedestrian navigation (see figure 6.3(b)) by removing a PDA from its in-car cradle. Within the car, the PDA displays the driving directions, and outside the car, it generates incremental directions for its pedestrian user – thereby providing a consistent interface to two different services on a single device.

Inside the Pedestrian Navigation component, several agents interact to provide this service. A dedicated agent, the *NavPed Control Agent*, translates between internal messages and the rest of the SmartKom system. It is also in charge of coordinating and controlling the process of generating incremental instructions. In addition, there is a tour planning agent, which generates a tour according to the user's specifications, and a positioning agent, which keeps track of the user's current position and alerts the control agent when the user reaches the end of the current segment or when she leaves the route. Furthermore, there is a map agent and an agent that encapsulates a GIS, and that can also perform some basic geometrical computations.



(a) (b)

Figure 6.3: SmartKom: screenshots of car navigation (a) and pedestrian navigation (b).

SISTO provides two services within the Pedestrian Navigation component. On the one hand, it is used to divide the complete route into smaller segments that allow for the generation of focused maps suitable for a small display. On the other hand, it generates incremental directions based on these segments. The resulting output consists of preverbal messages, which are translated by the control agent into the corresponding SmartKom format (M3L – see below). In this context, SISTO mainly relies the GIS agent in order to access the geometrical information needed to perform the segmentation.

The modules of SmartKom are arranged around multiple blackboards (used to share information and to perform collaborative operations on these), and they are running in parallel as independent threads. The modules are realized in various programming languages (C, C++, Java, Prolog) on different operation systems (Linux, Windows NT), which are all integrated by the underlying infrastructure. Contributing to this tight integration was the design decision to use an XML-based markup language for the encoding of all shared information, which allows for the easy access by virtually any programming language. For example, the word lattice, results from media fusion, and the presentation plan are all represented using M3L (**M**ulti-**M**odal **M**arkup **L**anguage) [Wahlster et al., 2001].

6.4 SISTO

The agent SISTO provides its embedding system with various services related to space. These include in addition to the localization and identification of arbitrary objects (see 4.3.2 and 4.3.1) the management of routes as well as incremental and complete way descriptions. The agent also provides a service to select landmarks in a given region and to determine objects that the user most likely wants to learn more about. Furthermore, SISTO supports the interaction with maps (using natural language), the computation of landmarks, the in-field collection of data, and the translation between relational and metrical concepts.

The localization service takes as input a target object, which can be specified either by name, or by internal ID, or by using an anaphoric expression. If the specification is ambiguous or incomplete, SISTO first tries to identify the target object. Once it has determined the most likely candidate, it starts to search for a reference object, a two-point relation, and a frame of reference in order to generate a relational localization. The search for a reference object does not only take into account spatial factors – such as closeness to the user, closeness to the target object, or visibility – but does also consider other properties of candidates, e. g. color, amount of information available on an object, or whether it has been mentioned before. The resulting list of potential reference objects has to be combined with the possible two-point relations and frames of reference for proper evaluation. All localizations, which are defined in this way are assigned with a degree of applicability that combines the objects' rating with those for two-point relations within a given frame of reference. The localization receiving the highest value is the one that the localization service returns to the agent that initiated the process.

The services for identification, description, and landmark selection are closely related to one another. They mainly rely on object evaluation but assign different weights to the relevant factors, thus exploiting the flexibility of the underlying computational model (see chapter 4): While landmark selection disregards user- and context-related factors – as landmarks should be prominent for all people regardless of their current context – these factors are highly relevant in the case of identification and description. The main difference between identification and description lies in the intention of the triggering query: In the case of identification the asking agent wants to learn the name/identity of an entity, in the case of description the goal is to obtain further information. Therefore, the relevance of factors such as previously being mentioned or closeness to the user differ greatly. For example, while the fact that an object has been mentioned previously increases the probability that the user might want to find out more about it, it drastically lowers the likeliness of the user wanting to learn its name. Consequently, the identification service and the description service both return the object with the highest rating, but which was computed according to their different sets of weights. The result of the landmark service consists of a list of objects that contains the candidates with the highest ratings. The size of the list depends on the number of landmarks, which the agent sending the original query specified in its message.

The route management service handles several tasks related to routes. On the one hand, it can generate language independent route directions for arbitrary trajectories. The preverbal messages returned by this service (see 6.4.2) contain two-point localizations for the start and the end point of the trajectory as well as the entire trajectory. Additionally, it includes a localization using n-point relations (path relations) as well as a reorientation instructions and metric information about the length of the segment. On the other hand, the route management service also monitors

route progress, and reacts to various events such as deviating from the precomputed route, route cancellation, or stopping for a longer time. Finally, it is also possible to generate a (language independent) description of an entire route.

Since maps often play an important role in navigation, SISTO provides a service for interacting with these: The use can specify a target location or object, for which she wants to see a map, and she can also manipulate the map (using natural language commands), i. e. panning or zooming. Similar to the identification and description of objects, this service mainly relies on object evaluation. This is also true for the data collection service, which enables the user of the system to input data on objects in her environment using natural language and/or a textual interface.

In addition to the services described in the previous paragraphs, there are two further ones that are not triggered directly by the user: Firstly, SISTO can translate relational spatial descriptions such as `right-of church` into metrical concepts, i. e. spatial regions. Secondly, the system can compute a set of potential landmarks for a given region. Usually, these services are requested by other agents such as a hotel reservation agent that needs to extract a spatial region from a natural language description or a map agent that wants to enrich a map with further landmarks. In the following section, we will introduce the architecture that allows for all these services to be realized.

6.4.1 Architecture

Although other agents perceive SISTO as single agent, it really is a *holon*: a team of agents that cooperate but appear as a single entity to the outside [Fischer, 1999]. This architecture does not only have the advantage of concealing the complexity of certain tasks that SISTO performs, but it also provides external agents with a single access point. In addition, the structure of the entire multi- agent system is simplified. The system consists of three distinct types of internal agents, called *micro agents*, which all communicate through an internal message bus: basic micro agents, task micro agents, and the Dispatcher. *Task micro agents* handle complex tasks as described in section 4.3. There are currently nine task micro agents corresponding to one or more complex tasks.

- **Identify** is in charge of identifying arbitrary objects,
- **Describe** identifies objects, about which additional information is requested,
- **Localize** generates relational localizations,
- **TransToGeo** translates relational expressions into geometrical representations
- **RouteManager** manages incremental guidance/complete route descriptions,
- **Landmarks** dynamically computes potential landmarks in a given region,
- **Position** determines the user's current position,
- **MapInteraction** allows for verbal access to maps, and
- **Acquisition** provides means to collect data on world objects.

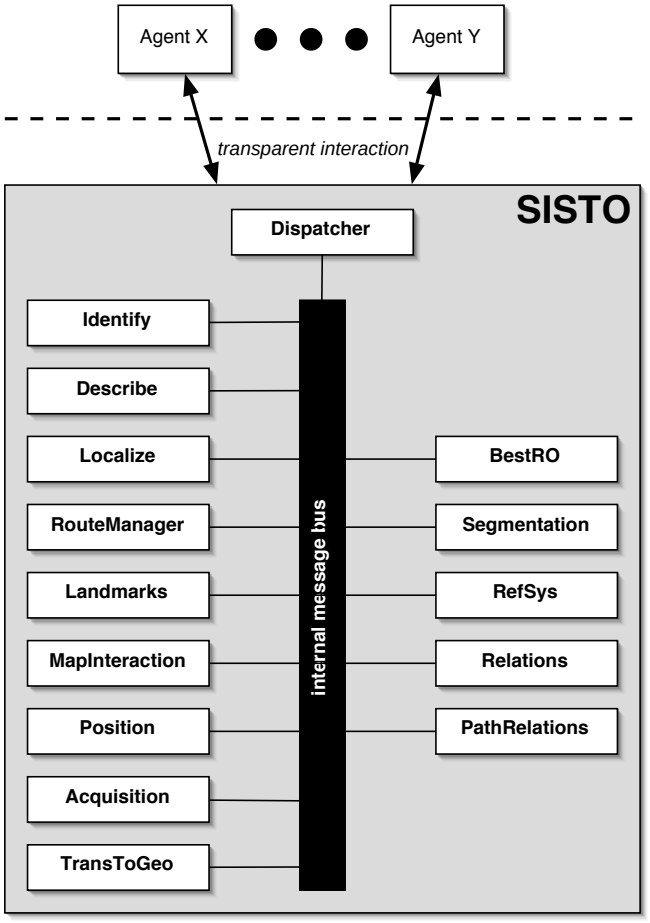


Figure 6.4: Internal architecture of SISTO.

The agents mentioned above are shown on the left side in figure 6.4, and mainly rely on the second type of micro agent, the *basic micro agents* (shown on the right), that provide fundamental services related to SISTO (see 2.5.2). There are again five agents of this type: *BestRO* evaluates real world objects such as buildings and streets along a number of spatial and non-spatial criteria. *RefSys* computes frames of reference based on the user's current position. The basic micro agent *Segmentation* partitions a longer trajectory according to several user- and context-related factors. *Relations* is in charge of computing two-point relations between arbitrary objects or geographical positions. Finally, *PathRelations* generates n-point (path) relations for trajectories and objects. All theses agents interact to solve their respective tasks.

The third group of micro agents only consists of a single agent, the *Dispatcher*, which is in charge of distributing incoming requests to the corresponding task micro agents that can handle them. It also manages the creation of new instances of micro agents as well as their destruction when they are no longer needed. In addition, the Dispatcher plays a key role in the adaptation to

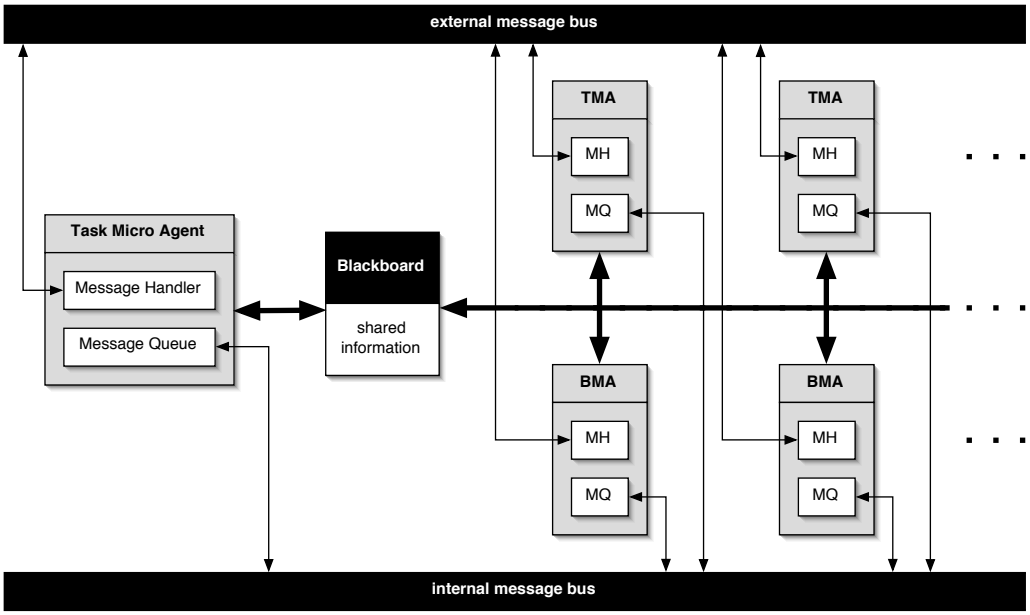


Figure 6.5: A team of internal micro agents: task micro agents (TMA) and basic micro agents (BMA) using message handlers (MH) and message queues (MQ) to communicate.

resource restrictions. In the current implementation, it only checks whether the available resources allow for the creation of further instances of a micro agent. If that is not the case, the corresponding request is put on hold until a corresponding instance finishes its current task and becomes available for allocation. However, more sophisticated adaptation strategies (cf. [Blocher, 1999], section 5.2) can easily be integrated into the Dispatcher, which has been specifically designed to allow for such extensions.

All agents run on their own thread, thereby performing concurrently.[5] Since most tasks cannot be addressed by a single agent, teams of agents are organized to cooperatively solve them. In order to achieve this, a task micro agent uses its knowledge about the task it handles to request the necessary micro agents from the Dispatcher, and assigns subtasks to the corresponding team members.

Figure 6.5 depicts a team of micro agents. Once the Dispatcher passes an incoming request to a corresponding task micro agent (TMA), this agent (shown on the left side of the blackboard) assembles the team (shown on the right side) according to its procedural knowledge about how to solve the task. A team can include both other task micro agents and basic micro agents (BMA), and it is organized around a central blackboard [Erman et al., 1980]. The blackboard stores information that is used by several agents and helps to prevent multiple request for the same data.[6]

[5] On a computer with a single CPU, the parallel execution is only simulated, e. g. using preemptive multitasking. On a machine with multiple CPUs, truly parallel execution is possible.

[6] It is important to highlight that the existence of multiple blackboards does not interfere with the principle of explicit interaction since only external request and the corresponding replies rely on it. The interaction between micro agent

Since several teams can exist at the same time, SISTO can be classified as a multi-blackboard system [Wahlster, 2001]. Each micro agent can communicate with its teammates through the internal message bus and with external agents through the external message bus. In the later case, the originating micro agent is concealed: external agents only see SISTO as the sender of the message. However, incoming replies to previous queries are automatically routed to the correct micro agent. A semi-autonomous message handler (MH) takes care of this routing by asynchronously sending and receiving messages on the external message bus. Sending internal messages is also an asynchronous process: each micro agent is equipped with a message queue (MQ) that is connected to the internal message bus. Since each agent runs on its own thread, it is possible to send and receive messages non-blockingly.

One of the main goals of the implementation was a direct correspondence between the entities and concepts of the model on the one hand and the components of the software on the other hand. We achieved this through the use of a multi-agent system where each complex tasks or basic process corresponds to a distinct agent. In order to distinguish between basic and complex tasks, we introduced the distinction between basic and task micro agents. While the first ones directly implement the basic processes described in 4.1, the later ones realize a superset of the complex tasks described in 4.3. The interactions that we identified as being necessary to perform these tasks and processes directly map to the messages exchanged between the members of an agent team. Therefore, the implementation allows for an easy realization of further complex tasks and contributes to the validation of the model.

In order to illustrate the close relationship between model and implementation, figure 6.6 shows the messages exchanged by the ‘Localize’ agent when computing a localization. The thick arrows directly correspond to the interactions in the model which were depicted in figure 4.16. Additional messages (marked with regular arrows) mainly serve the purpose of obtaining information that is required to perform reasoning, or to allocate resource needed in the process. For example, a message is sent to the GIS agent to obtain a list of objects that are located within specific regions such as the immediate vicinity of the user’s current position. The labels on the arrows correspond to the actions, which are defined in the internal XML-based ontology. In figure 4.16, task micro agents are shown as black boxes, basic micro agents are shown as gray boxes with boldface labels, and the Dispatcher is highlighted by underlining. The only ‘foreign’ agent, the GIS agent providing basic geometrical services, is depicted using a gray box with a label in regular type. It is worth mentioning that there are a number of further agents involved in this task which are left out since the corresponding interaction takes place within the context of Identify (e. g. when querying the context model).

When we compare the diagram in the figure to the one shown in figure 4.16, we observe a close correspondence between them: The basic interaction pattern is conserved and for each process/task in the model there is a corresponding agent in the implementation. The main difference lies in some additional interactions on the implementation side. For example, upon reception of a localization request (an ‘SALocalize’ action containing a partially filled preverbal message) the Localize agent first requests several agents from the Dispatcher using a message containing ‘ControlAction’ which specifies what agents are needed. The next step consists of identifying the target object by means of an ‘SAIdentify’ message containing the original description encoded in a

always consists of explicit messages sent over the internal message bus.

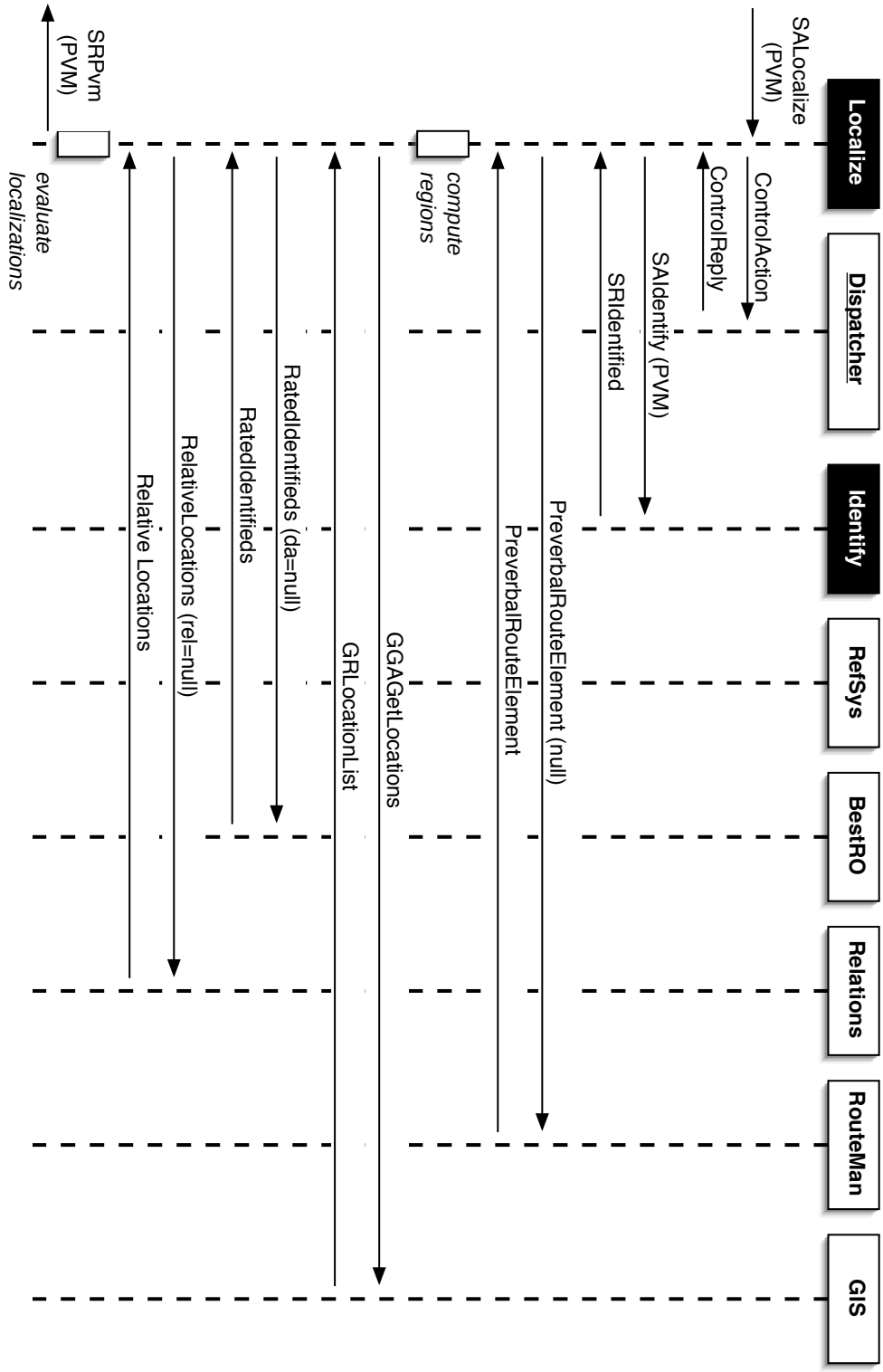


Figure 6.6: Agent interaction for computing a localization.

PVM. After the most likely target object has been determined, the Localize agent requests a frame of reference from the basic micro agent 'RefSys' (via a message containing an empty 'Perspective' object). Once Refsys returns this, the current route segment is asked from the 'RouteManager' task micro agent. Then, 'Localize' computes several regions based, for example, on the user's current position and the current route segment. These regions are passed to the GIS agent, which returns a list of objects located within their borders. This list of objects forms the set of possible reference objects. The 'BestRO' basic micro agent evaluates these before 'Relations' determines the spatial relations that apply best for each object. Finally, 'Localize' evaluates all resulting relational localizations and returns the most suitable one via a message containing a preverbal message, which encodes the localization. From this example, is it easy to see that implementation and model do indeed closely correspond to each other. The other complex tasks are defined analogously.

SISTO consists of roughly 27000 lines of 100% pure Java Code, and it is realized as an agent in a multi-agent system (RAJA). An Apple Macintosh PowerBook G3 running Mac OS X served as the prime development platform. Most test were run on a dual Athlon cluster node running Suse Linux 8.0, and on a standard PC running Windows 2000. Internally, SISTO relies also on a (tightly coupled) multi-blackboard multi-agent system, which is transparent for external agents. Since one of its embedding systems, Deep Map, is also entirely written in Java, we have been able to successfully run SISTO on Windows (NT, 2000, XP), Linux, Solaris/Sparc, and Mac OS X. Basically, any Java-compliant platform can serve as a host for the system.

6.4.2 Interaction and adaptation

In this section, we take a closer look at the internal workings and interactions of SISTO as well as at how it interacts with the rest of the system. One goal in designing SISTO was to achieve language independence when performing spatial reasoning. Therefore, we developed a concise representation format that allows for the encoding of a wide range of utterances related to spatial topics (see 4.2). The services provided by SISTO basically rely on a simple query-answer scheme: the user asks for information and the system provides her with it (if possible). At the beginning of 6.4, we listed several tasks that SISTO addresses. Of those, the user directly asks the system

- to identify objects in her vicinity,
- to provide further information on objects,
- to localize objects,
- to guide her to a specified location,
- to describe the entire route to a specified location,
- to tell her where she is, and
- to collect data on world objects.

In addition, she can control the system (e. g. stop the current operation), input data associated with arbitrary objects, and respond to the questions of the system, e. g. in the context of interactive positioning.

```
<complexType    name="PreverbalMessage">
   <complexContent>
   <extension base="dmo:Reply">
   <sequence>
      <element name="performative"
               type="dmo:Performative"
               minOccurs="0" maxOccurs="1"/>
      <element name="pvmTypes"
               type="dmo:PreverbalMessageType"
               maxOccurs="unbounded"/>
      <element name="pvmoGlobalGoal"
               type="dmo:PreverbalMessageObject"
               minOccurs="0"/>
      <element name="stprTwoPointRelation"
               type="dmo:SRTwoPoint" minOccurs="0"/>
      <element name="pvmoTwoPointRelation"
               type="dmo:PreverbalMessageObject"
               minOccurs="0"/>
      <element name="daTwoPointRelation"
               type="dmo:DegreeOfApplicability" minOccurs="0"/>
      <element name="tid"
               type="dmo:TourID" minOccurs="0"/>
      <element name="tourState"
               type="dmo:TourState" minOccurs="0"/>
      <element name="snpr" type="dmo:SRNPoint" minOccurs="0"/>
      <element name="pvmmoNPointRelation"
               type="dmo:PreverbalMessageObject"
               minOccurs="0"/>
      <element name="daNPointRelation"
               type="dmo:DegreeOfApplicability" minOccurs="0"/>
      <element name="stprStart"
               type="dmo:SRTwoPoint" minOccurs="0"/>
      <element name="pvmoStart"
               type="dmo:PreverbalMessageObject"
               minOccurs="0"/>
      <element name="daStart"
               type="dmo:DegreeOfApplicability" minOccurs="0"/>
      <element name="stprEnd"
               type="dmo:SRTwoPoint" minOccurs="0"/>
      <element name="pvmoEnd"
               type="dmo:PreverbalMessageObject"
               minOccurs="0"/>
      <element name="daEnd"
               type="dmo:DegreeOfApplicability" minOccurs="0"/>
      <element name="angle" type="dmo:SE2DAngle" minOccurs="0"/>
      <element name="metric" type="dmo:Distance" minOccurs="0"/>
      <element name="objectType"
               type="dmo:ObjectType" minOccurs="0" maxOccurs="1"/>
      <element name="attribute"
               type="dmo:Attribute" minOccurs="0" maxOccurs="1"/>
      <element name="choices"
               type="dmo:PreverbalMessageObject"
               minOccurs="0" maxOccurs="unbounded"/>
      <element name="text" type="string" minOccurs="0"/>
   </sequence>
   </extension>
   </complexContent>
</complexType>
```

speech act and type of content

general target of utterance

relational localization

reference to entire route and its state

localization using path relations

relational localization of the start of a segment

relational localization of the end of a segment

description of a route segment

(turning) angle and distal info.

used for data collection to encode object type, attributes, and choices

for error handling

Figure 6.7: Data structure used to encode a preverbal message.

The questions corresponding to these requests as well as the replies that the system generates have to be properly encoded in the data structure representing the preverbal message. Figure 6.7 shows the XML representation that directly corresponds to the internal object format. (Appendix B gives a detailed description of the internal encoding of all interactions with the user.) This representation not only allows for the easy translation to other formats (such as the one realized within the pedestrian navigation of SmartKom), but is also more readable to a human operator. In addition, it facilitates filtering and post-processing, which we exploited in designing the debugging interface to SISTO and Deep Map.

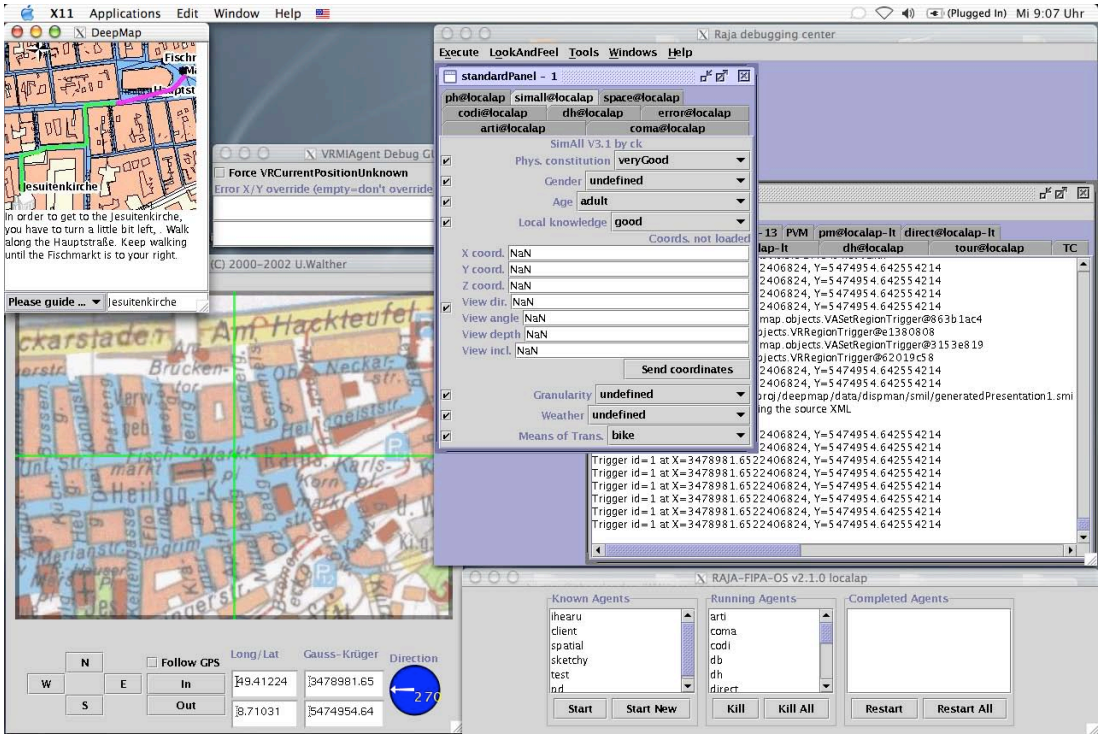
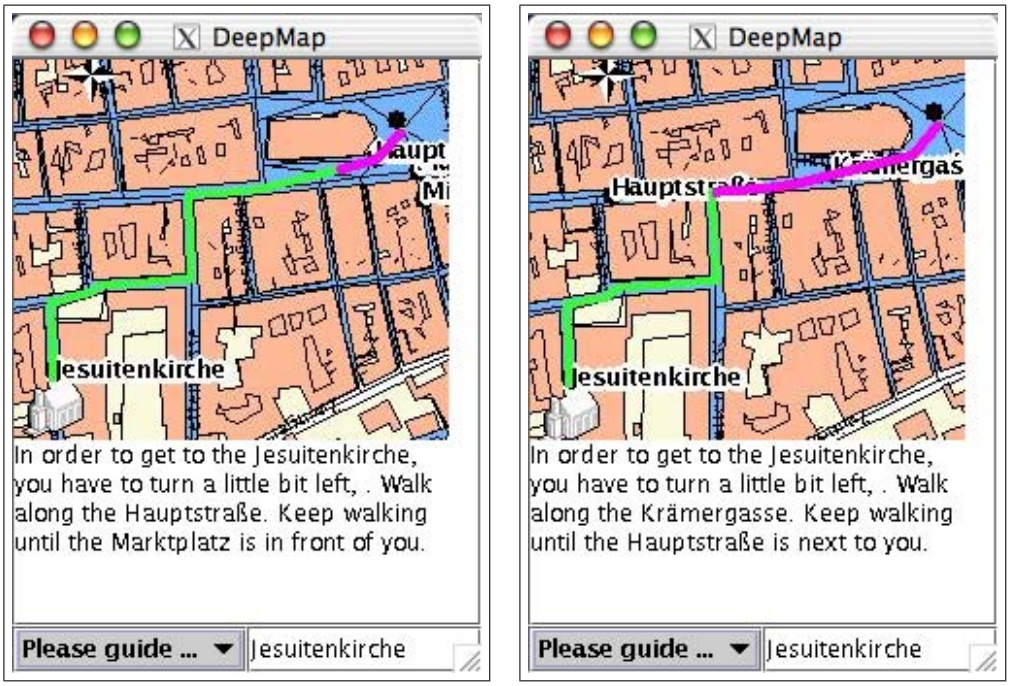


Figure 6.8: Screenshot of the debugging and development environment of SISTO and Deep Map.

Figure 6.8 shows SISTO running on Deep Map. In the upper left corner of the picture, the system's user interface is depicted as it would appear on her PDA. The GPS manager is shown in the lower left part; clicking on the map simulates the user being located at the corresponding position in Heidelberg. Her view direction can be adjusted using the dial controller in the lower right corner of the GPS manager's window. In order to simulate outages and imprecise sensor readings, there is an additional window (shown next to the output window, and above the GPS manager), where we can manually enter the error associated with the X- and Y-coordinate returned by a GPS receiver. In addition, there is a checkbox to turn off the GPS simulation, which is equivalent to a GPS outage for the rest of the system. Since the GPS manager is also used to connect the GPS receiver to the system, its simulated output is indistinguishable from 'real' sensor readings for the other agents of the Deep Map system.



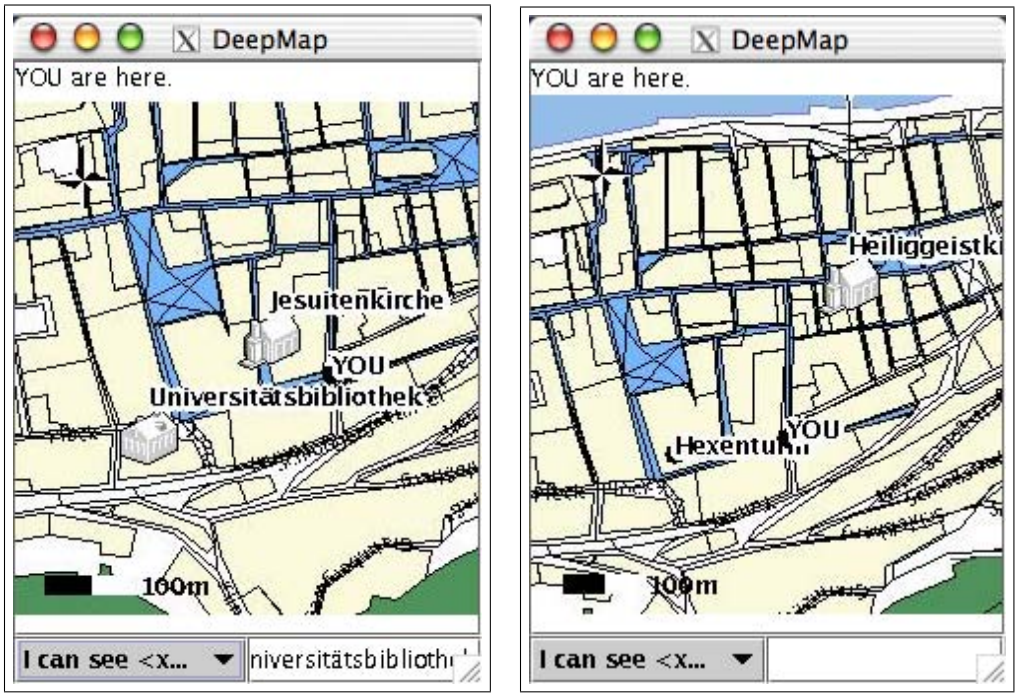
(a) A slow walker (b) A bike rider

Figure 6.9: Incremental guidance: adaptation to user properties

The lower right corner of figure 6.8 depicts the control interface for the entire agent system. It shows the agents that are currently running, those that have been shut down as well as the ones that have not yet been started. In the upper right corner of the screenshot, the main internal interface for debugging and optimizing purposes is shown (labeled 'RAJA debugging center'). It consists of two main parts: On the left side, there is an internal window (labeled 'standardPanel-1') that contains the graphical interfaces of the individual agents that are currently running. For example, in the figure, the interface to the user and context model simulation is shown, where the operator can adjust factors such as the age or constitution of the user as well as her current means of transportation. On the right side of the internal interface window, there is the internal window that collects all debugging messages that the system outputs. These messages are sorted according to which agent they came from.

Although the system currently only considers a few situational factors, they are used to adapt all services it provides. In figure 6.9 an example for this adaptation is shown in the context of incremental guidance. Two factors that play an important role in the process of segmentation – the subdivision of a long route into smaller parts that are well suited for incremental guidance – are the means of transportation and (depending on the means of transportation) the physical constitution. In both scenarios depicted in the figure, we assume that the user is located on the market and asks the system for guidance to the Jesuitenkirche. However, while the user is a pedestrian of

poor physical constitution in case (a), she is riding a bike in case (b) and of average physical constitution. Consequently, SISTO adapts the length of the segment in order to take into account the difference in speed. Figure 6.9(a) shows the much shorter segment generated for the slow walker, and figure 6.9(b) the much longer segment generated for the biker. (Figure 6.15(a) depicts the corresponding segment for an average pedestrian such as Alice.)



(a) Visitor 1 (b) Visitor 2

Figure 6.10: Interactive positioning: adaptation to interaction history

A further example for the adaptation to situational context is shown in figure 6.10. The two screenshots show the output of the interactive positioning that we presented in section 5.3. In this example, we assume two visitors who do not differ in terms of their inherent properties, and who are both located at the same position. In both cases, there is no positional information from the GPS or the position history. Hence, both visitors went through the interaction described in 5.3 to help the system in determining their current position. However, due to the fact that each visitor has a distinct interaction history with the system, and has seen different objects prior to asking for her current location, the generated you-are-here-maps are different as well. Even though both visitors are located at the same position, there are different objects on the maps shown in the figure, and the map region differs as well. Since visitor 1 only visited the Jesuitenkirche and the Universitätsbibliothek in the surrounding area, they are included in figure 6.10(a) and the region depicted in the map is smaller than in figure 6.10(b). In the latter case, the region is larger in order to include the objects that visitor 2 has seen in the area surrounding her current position.

6.5 A generic positioning architecture

In previous chapters we have argued that positional information is a key factor in mobile computing in general as well as in determining the user's current situation. Consequently, we first presented an analysis of what factors determine a position in 6.5. In chapter 4, we proposed a model for interaction on spatial topics, which employs the preverbal message to encode related interactions independent of the target language or modality. In 5.3, we then introduced several strategies to adapt to positional information of varying quality and to the absence of any information, which were based on the previously proposed approach. In this section, we now present a generic architecture that realizes the aforementioned methods in a generic way, and therefore allows for its application in various scenarios.

However, we first need to determine the requirements such an implementation will have to fulfill. On the one hand, it should be transparent for other components that require positional information, whether the result stems from direct measurement. The complex interactions that may be necessary to determine the current position should be hidden from querying agents to facilitate their realization. On the other hand, the implementation should be efficient since positional information is frequently used in a mobile system. Otherwise, the positioning component could slow down the entire system.

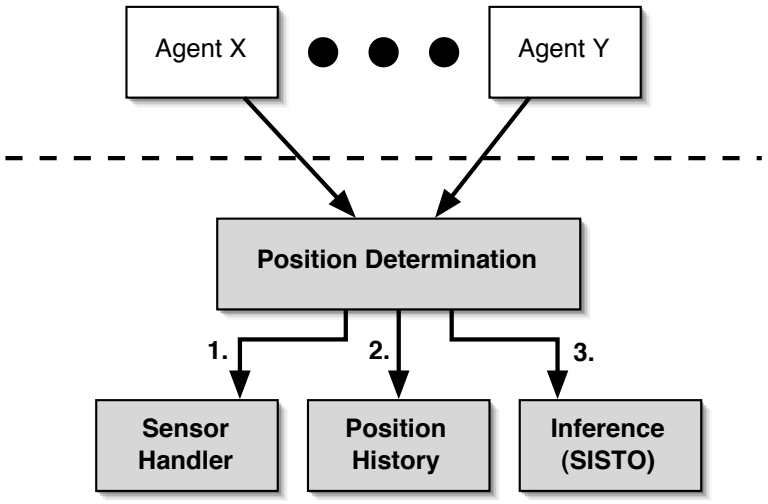


Figure 6.11: A generic architecture for handling positional information.

In order to address these requirements we chose a holonic architecture as shown in figure 6.11: External agents always send their queries to the agent 'Position Determination', which serves as 'proxy' for positional information and coordinates the process of determining the current position. The requests sent by external components specify what kind of positional information (e. g. heading) is needed, at what precision, and of what recency. These corresponding replies are computed using a three layer approach: The 'Position Determination' agent first checks whether the sensor handling component 'Sensor Handler' can provide the requested information. If that is the

Trigger event	Spatial triggers	Temporal triggers	Combined triggers
entering	*enter trigger* fire when user enters region	*time trigger* fire at a given time	*timed enter trigger* fire when user enters a region at a given time
exiting	*exit trigger* fire when user leaves a region	*time trigger* fire at a given time	*timed exit trigger* fire when user leaves a region at a given time
staying	N/A	N/A	*stop trigger* fire when the user's position does not change for a given interval of time
moving	N/A	N/A	*change trigger* fire when the user's position does change during a given interval of time

Table 6.1: Trigger classification: Each type may either be persistent or temporary

case, the corresponding reply is sent back to the external agent. If sensor readings do not meet the requirements specified in the original query, 'Position Determination' accesses the 'Position History' to see whether its records contain information that meets the query criteria. If that is the case, the reply can be generated from that. Otherwise, the procedures described in 5.3 have to be applied. The agent that realizes these was described in 6.4.

This architecture not only enables external agents to simply request positional information (from 'Position Determination') without the need to know the complex mechanisms behind it but also avoids unnecessary computation: Since the query is successively passed through three layers – from the fastest to the computationally most expensive one – and 'aborted' once a sufficient reply is determined, the least expensive layer is always selected.

In order to further increase the efficiency of the approach, we designed a new concept to address a very common problem in handling positional information: Often, a component wants to know when the user approaches a certain position, or when she leaves a certain region. A frequently used method to implement this consists in the corresponding agent regularly requesting the current position of the user, even though she may never get to that position. If there are several agents that require positional information, they may produce a significant amount of unnecessary 'polling traffic'.

Therefore, we developed a *spatiotemporal trigger* concept that avoids these problems. The basic idea lies in not continuously asking for a the current position but telling the positioning system to send a notification message when the user reaches a certain position[7]. This reduces the required traffic tremendously: instead of a steady stream of position requests and queries going back and forth, a single initiating query is sent as well as the corresponding reply containing the requested positional information. Table 6.1 lists the different categories of spatiotemporal triggers

[7]This concept resembles the one proposed in the context of leaving 'spatial notes' (e. g. [Hohl et al., 1999]) where users can leave virtual notes at arbitrary places that can later be seen by anyone passing that location.

we identified, which can be distinguished according to the following criteria:

- **region- and/or time-based**
  A trigger can either be tied to a predefined region or to a temporal interval. In the first case, the trigger fires when the relative location of the user to the defining region changes. In the later case, it either fires at a given time or interval (e. g. every two seconds), or during predefined interval. A combination of both types triggers a message when the spatial and the temporal criteria are met, e. g. when the user enters a region within a given time span.

- **trigger event**
  In case of a spatial trigger, there are two qualitatively different events: the user may either enter or exit the region associated with the trigger. Analogously, the user may 'enter' or 'exit' an interval of time in the temporal case. Furthermore, there are two main spatio-temporal events: the user may not move for a given amount of time, or her position may change within a given interval of time.

- **persistence**
  Orthogonally to the previous two criteria, we can distinguish between triggers that fire a defined number of times and persistent triggers that are active until they are removed.

Not all triggers listed in table 6.1 were employed to realize the prototypical system presented in this chapter. Figure 6.12 shows a few examples of spatiotemporal triggers that are key ingredients in generating incremental directions: The arrow corresponds to a path segment that has been selected by the path segmentation process described in 4.1.5. In order to react appropriately to the motion of the user after she has been instructed for this segment, four different triggers are installed. An *exit trigger* is associated with the immediate environment of the segment (see figure 6.12(a)) which automatically notifies the route manager in case the user leaves the proposed route. The route manager can then initiate the computation of a modified route and generate new instructions for the current position of the user. In addition, there is a *stop trigger* (shown in figure 6.12(b)), which fires when the user stops to move on the segment for a certain amount of time. In this case, the route manager can either repeat the last directions, or generate a new one for the remaining part of the segment. Finally, there is an *enter trigger* associated with the end of the segment. When the user enters the corresponding region (see figure 6.12(c)) the route manager knows that she is approaching the end of the segment, and starts to work on the directions for the next segment.

The corresponding regions are shown in three distinct sub-figures for clarity reasons only – in reality they do overlap which further increases the flexibility in defining under what conditions a trigger should fire. In addition to the route manager, the position history agent does also employ a spatio-temporal trigger: A persistent *time trigger* is installed that fires every two seconds, thereby providing the position history agent with regular updates on the user's current position. While the other triggers listed in table 6.1 are currently not used in the system, they can nevertheless allow to realize various mobile applications in an efficient way.

For example, *timed enter and exit triggers* allow for the easy realization of time- and space-dependent adaptive presentation of information: Installing such a trigger at the entrance of a museum would enable the system to automatically notify a user of the beginning of guided tours only

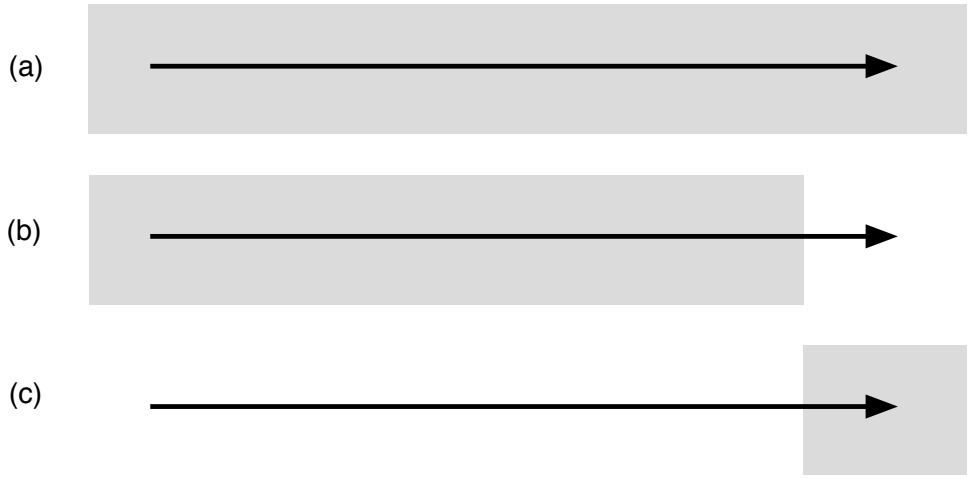


Figure 6.12: Examples for spatiotemporal triggers.

when she is at the right place at the right time. *Stop and change triggers* can help to support a user model agent in learning the user's interests: Associating either one with a number of sights provides the user model with information about how much time the user spent at a given sight, which is a key factor in determining what the user finds interesting. These are just a few examples for the usefulness of the concept of spatio-temporal triggers. Generally, this approach allows for an efficient association of events and information with spatio-temporal constraints. Unlike other approaches, the consumption of bandwidth and computational resources is drastically reduced since activity only takes place when the specified constraints are met. Furthermore, no special infrastructure (e. g. [Hohl et al., 1999]) is needed to realize the services mentioned above.

6.6 A journey through Heidelberg

In order to provide a better insight into the capabilities and features of the implementation, this section will follow a tourist who is visiting the city of Heidelberg, Germany. We will show how the system can support her on her journey through the old town, and we will highlight in what respect the assistance goes beyond what a paper tourist guide (or most other navigational assistant systems) can achieve.

Let us assume that Alice is on a trip through Europe, and that she has decided that Heidelberg is one of the cities she would like to visit while she is in Germany. She arrives at Heidelberg at Karlstor station where she disembarks the train. She starts her journey by walking a little bit along the river Neckar until she turns toward the city, and follows a few of the narrow aisles of the old town. After some time she arrives at the market, where she sees a big old church.

(a) (b)

Figure 6.13: Object identification and description.

Since she has always been fascinated with sacral buildings and their history, she is curious about this church. Consequently, she pulls out her mobile assistant and asks “What’s this?”. The system replies by showing a presentation such as depicted in figure 6.13(a), which consists of a picture of the Heiliggeist church, a textual part identifying the church, and a spoken output repeating the textual output. Had Alice been using a paper guide, she would have had to perform

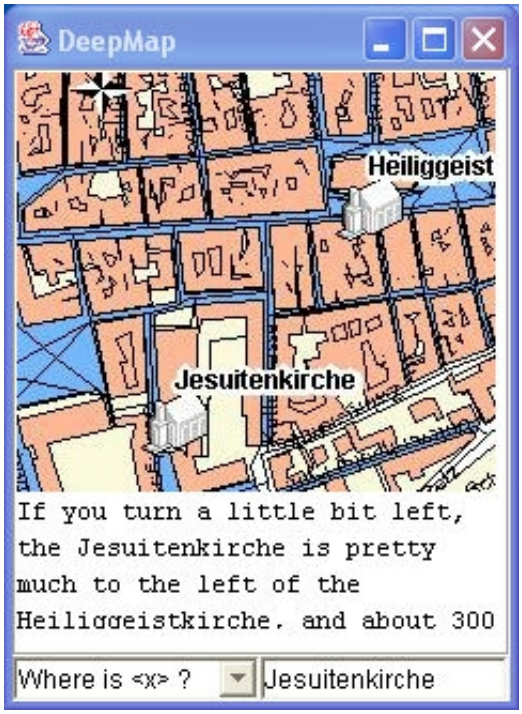


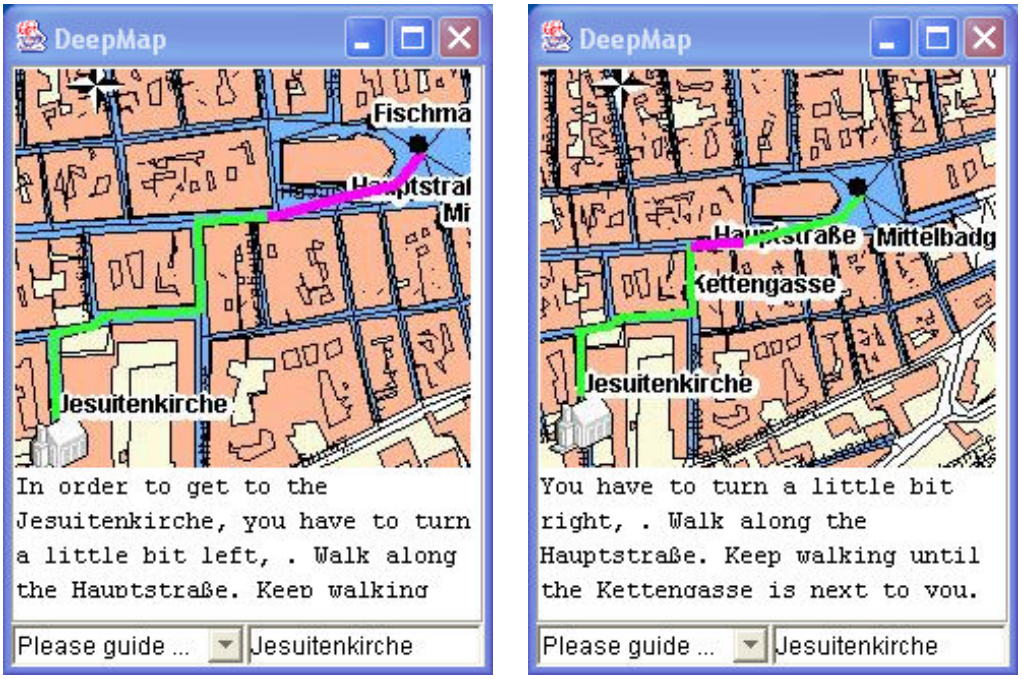
Figure 6.14: Localization

several steps in order to learn what church she is facing: First, she would have had to determine her own position and orientation using a map, and external information sources such as street signs or other people. If the map has several sights highlighted on it, she would have had to match the caption/legend with the symbols of the map. If the map does not have this specific sight on it, she would have had to browse through the entire guide to either find a matching picture or description.

But for the moment, Alice's interest has been awaken and she wants to know more about this church. So, she says "Tell me more about it.", and her assistant replies with a slide show (see figure 6.13(b), where each slide consist of a picture and a longer part of text giving background information on the Heiliggeist church. Again, had she been using a paper tourist guide, she would have had to use the index to find the page describing the church and then browse to the corresponding description. However, she needs to know the name of the church to do so. Otherwise, she would have to browse through the entire guide trying to find a matching picture or description, or she would have to perform the same steps as listed for the identification case.

Once Alice has learned enough on the Heiliggeist church, she remembers another church from her studies prior to her trip, and she decides that she would like to see it as well, if it is not too far away. In order to do so, she asks her mobile assistant "Where is the Jesuitenkirche?". The resulting reply consists of a map along with textual and spoken output (see figure 6.14). On the map, both churches are highlighted, the Heiliggeist church for being the anchor object of the localization and the Jesuiten church for being the target object. The verbal description does not only include a qualitative localization but also a metric distance, which helps Alice decide whether to go there

or not. Note that the exact location of the Jesuitenkirche is somewhat hard to describe in a few words. Therefore, the system generates a linguistic hedge (to signal its user that the Jesuitenkirche is not exactly to the left) as well as an induced frame of reference (to improve the applicability of the corresponding relation). This kind of adaptivity to the user is obviously impossible when using a paper guide.



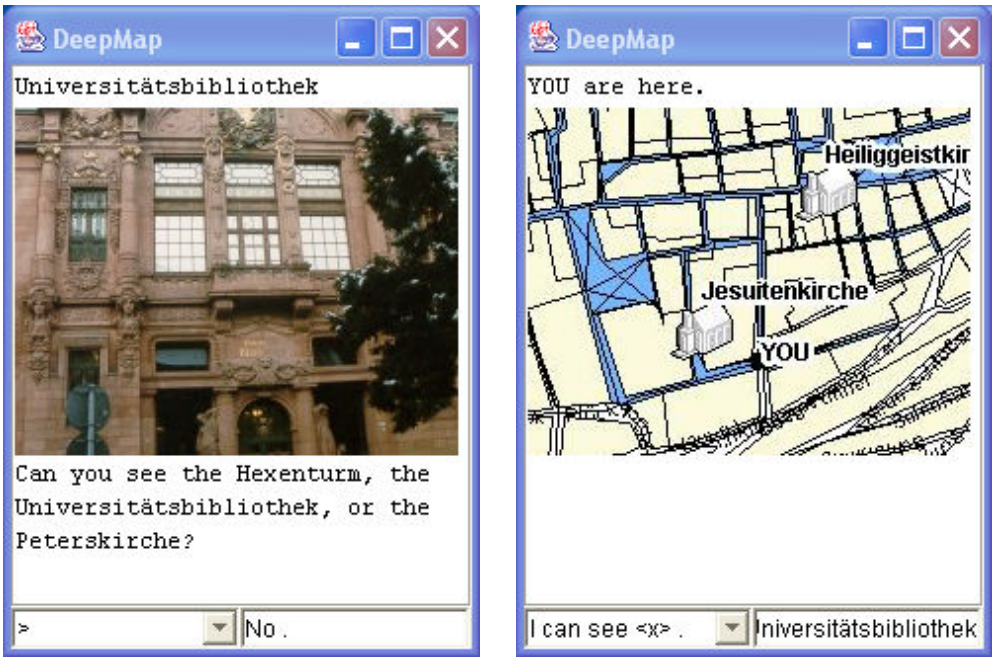
(a) (b)

Figure 6.15: Incremental guidance.

When Alice learns that the church is pretty close to her current location, she decides to go there and have a look at it. So, she instructs her mobile assistant "Please guide me to it." The system responds again by a map with verbal output (see figure 6.15(a)). The given directions are both displayed and spoken, the former for reference while she is following the segment and the later to support navigation without having to look at the display all the time. Using a paper guide, she would have to rely on a city map and her location awareness. Alice, however, follows the instructions she received from her assistant, and when she arrives at the end of the route segment described in the directions, she is automatically presented with directions for the next segment (see figure 6.15(b))[8]. Had she been using a paper guide, she would have had to continuously match her position in the real world with a city map.

[8]Should Alice leave the route, her assistant would automatically compute a new route and provide her with adapted directions to enable her to reach her destination from her new position.

With the help of the system, Alice makes her way to the Jesuiten church. After learning more about the history of that building, she puts away her guide and goes on a sightseeing tour through the scenic aisles of the old town. While she is leisurely wandering around the city, she suddenly realizes that is has gotten late, and that her train will leave in half an hour. Even worse, she has no idea of where she is at the moment. She pulls out the navigational assistant and asks “Where am I?”. Since she is standing in a narrow aisle, there is no GPS signal and thus, her position cannot be determined from sensor readings. So, the system replies with a question asking “What street are you in?”. Alice is rather surprised by this response but nevertheless looks around for a street sign. Sure enough, there is one on the wall across the street, and so she replies “I am on Seminarstraße.”. This provides the system with some rough information about her location. In order to determine it more precisely the assistant generates a very limited number of questions such as the one shown in figure 6.16(a), asking whether Alice can see certain buildings. The presentation also includes a slideshow of photographs of those buildings along with their name. This facilitates Alice’s task and she quickly answers all the questions. In the end, the system generates a personalized you-are-here map, which does not only contain her current location but also objects that are familiar to her (see figure 6.16(b)). Had Alice been using a paper guide, she would have had to rely on the index of street names and a city map. Alice looks a the map and realizes with relief that she will make it in time to catch her train. She asks her system for guidance to the Karlstor station and on her way back even finds the time to buy a souvenir for her husband.



(a) (b)

Figure 6.16: Interactive positioning.

6.7 Summary

In this chapter, we presented details related to the implementation of our model (and adaptation strategies), which we introduced in the previous chapters. We first reviewed several key requirements in designing a mobile and interactive system. On the one hand, such a system should allow the user to access the information in several ways according to her personal preferences. On the other hand, it is important to include context-sensitive as well as dynamic information. Furthermore, the system should support the inclusion of external/interactive services. Furthermore, it is desirable that the architecture of the system mirrors the underlying model and that is supports reuse. In a mobile setting, the intelligent treatment of positional information is another key factor. We also gave a short introduction to multi-agent systems, which provide means to address some of the above listed requirements.

Secondly, we reviewed the technical context of the prototypical implementation. We introduced Deep Map, an intelligent mobile tourist guide for the city of Heidelberg, and Smartkom, a large system exploring new ways to improve and support various types of communication, e. g. through multi-modal interaction. We then presented SISTO, the prototypical implementation of our approach, which provides various services within both Deep Map and Smartkom. It is realized as a multi-agent system that consists of multiple agents of different types. These agents directly correspond to the basic and complex processes modeled in chapter 4. Upon reception of a query, teams form dynamically in order to generate the corresponding reply. This holonic structure is hidden from external agents, which only interact with a single entity (SISTO).

In this chapter, we also introduced a generic architecture for the handling of positional information. It is based on a similar holonic scheme, where the complexity of the task is hidden from other components of the system: The user's current position is determined in three steps by first checking sensor readings, then querying the position history, and finally performing various reasoning and interaction with the user. Another key factor in streamlining the handling of positional information lies in the concept of spatio-temporal triggers, which we presented as well. By assigning certain events (such as the user entering) with regions and/or time, we were able to drastically reduce the overhead of keeping track of the user's position and to reacting appropriately.

In order to illustrate the fitness of our proposed approach, this chapter concludes on an example journey, where we follow a fictive tourist on a journey through the city of Heidelberg. In the process, we show the interaction between the user and the system and point out some key features of our model.

7 Conclusion

There are several contributions of this thesis, which we shortly summarize in this chapter. These include a comprehensive and extensible model for the integration of situational factors into spatial reasoning, a systematic analysis (and implementation) of induced frames of reference, and a language-independent representation format for situated interaction on spatial topics. In addition, we presented a comprehensive approach (and implementation) for determining and handling positional information based on sensor data, inference, and interaction with the user.

On a practical level, the model itself as well as the prototypical implementation allow for the rapid development of systems incorporating interaction on spatial topics. For example, the prototypes supports the mobile collection of data on arbitrary real-world objects. In the following, we will summarize the most relevant results of this thesis in detail, and point out several opportunities for further research based on our findings.

7.1 Scientific contributions

Previous approaches in spatial reasoning focused mainly on geometrical and spatial properties and constellations. Since they neglect non-spatial factors during computation, one of the main goals of this thesis was to investigate ways to take situational factors into account during the reasoning process. The pitfalls and specific problems of real world use of mobile systems are further key issues that are often ignored and that we addressed in this work. Specifically, the following results are worth highlighting:

- **An analysis of situational factors impacting interaction on spatial topics**
  In chapter 2, we presented an analysis of what factors can have an impact in the context of interaction on spatial topics. These factors comprise user- and context-related factors as well as the available resources and positional information. In addition, the task the user is performing also plays an important role.

- **A model for situated spatial reasoning**
  Based on an analysis of what basic processes contribute to the realization of complex tasks related to space, we designed a computational model for situated interaction on spatial topics, which takes into account the situational factors that we previously identified. Due to its flexibility, the inclusion of additional factors as well as the adaptation to new empirical findings are as easy to achieve as is the compensation for the (temporary) unavailability of situational information.

- **A modular approach for designing complex spatial tasks**
  Using the aforementioned model, we were able to conceptualize several complex tasks related to space which also involved user interaction. The proposed approach not only facilitates the design of these complex tasks but also reveals the interactions required to perform them. In addition, the resulting model lends itself to a modular and reusable implementation (see below).

- **A language- and media-independent representation format**
  In order to support a broad range of applications, we designed a representation format that allows for unified encoding of all interactions between the system and its user. In addition to its independence of a specific natural language, the proposed format also supports further media and modi (aside from verbal means). We illustrated this flexibility by reviewing various presentations for route directions that can be generated from a single preverbal message, which ranged from verbal instructions over cartographic presentations to three-dimensional animations.

- **Several adaptation strategies for real-world use**
  The use of mobile systems in a real-world scenario poses some challenging problems such as the unavailability of information or severe resource restrictions. We presented several strategies that help to address these issues using the basic model we proposed. Some of these strategies were based on a systematic analysis of generic approaches for addressing the lack of information.

- **A comprehensive approach for the determination of the user's position**
  Since the determination of the user's current position is a key factor in a mobile scenario, we presented an integrated approach to handle positional information. In addition to sensor data, our model includes knowledge based dead reckoning as well as the interaction with the user.

- **An empirically grounded model for n-point relations**
  A series of experiments on the applicability of n-point relations (path relations) provided us with new insights on the corresponding concepts in human reasoning. Specifically, we found that the proximity between anchor and target object has a major impact, which we used to extend the model presented in [Kray and Blocher, 1999].

- **A systematic analysis of induced frames of reference**
  We introduced a new dimension in the categorization of frames of reference by analyzing the way in which a frame is established. Previously, only the immediate establishment was considered but inducing a frame through meta-communicative acts opens up some interesting opportunities. The first implementation of this concept in SISTO was used, for example, in handling imprecise positional information and to improve the quality of localizations.

In addition to these contributions to the theory of situated interaction on spatial topics, we were able to make several advances in the field of practical application. These are based on an extensive implementation not only of the core model but also of further auxiliary agents. More specifically, the following contributions are worth highlighting in this context:

- **A prototypical implementation**
  Within an intelligent mobile tourist guide, we implemented a prototypical system that realizes several key points of the model presented in this thesis as well as some adaptation strategies. During the development of the system, several extension were realized and easily integrated, thereby supporting the claim of flexibility and extensibility of the underlying model. The use of SISTO in two different host systems is further evidence in this context. In addition, we conducted some preliminary tests with the system in the city of Heidelberg.

- **A generic architecture for agent-based spatial reasoning**
  Since one of the embedding systems (Deep Map) as well as the core implementation of our model (SISTO) are agent-based, we were able to illustrate the suitability of multi-agent systems for spatial reasoning. More specifically, we were able to directly map the modular model as well as the complex tasks to a multi-agent system, and to integrate the resulting system into two host systems (SmartKom and Deep Map).

- **Support for mobile data collection**
  The prototypical implementation allows for the (mobile) collection of data on arbitrary objects through a combination of spatial reasoning (e. g. object identification) and ontology-based data entry. The collected data is stored in an XML format that directly corresponds to the internal format of the database, and therefore facilitates the process of maintaining it and keeping it up to date.

- **A generic architecture for the handling of positional information**
  In mobile applications, the position of the user is often one of the key factors to keep track of. We proposed a generic architecture for handling positional information in such a scenario that is transparent for the rest of the system, which facilitates the use of positional information. In addition, its three level approach (sensor data, position history, knowledge based determination) not only allows for the adaptation to varying quality and availability of positional information but also for its application to various domains. Due to its agent-based implementation, the latter task is further facilitated (such as in the case of SISTO, which was used in two different host systems).

While the implementation of the model and adaptation strategies presented in this thesis (SISTO) is only a research prototype, it is – to our best knowledge – the first system to realize the following two features in a real-world application:

- **Determination of the user's position through a dynamically optimizing query-answer paradigm**
  Unlike most previous systems, SISTO can determine the user's current position even in the absence of any information from the sensors or a position history. While there are a few systems (e. g. GUIDE and LoL@ – see 3.3 and 3.6) that incorporate a rudimentary mechanism for asking the user in order to determine her position, SISTO goes beyond that in several ways. On the one hand, it dynamically generates questions based on what knowledge is available, for example street names or recent sensor data. The questions are not only optimized in terms of quickly reducing the number of potential positions but also in terms of minimizing the number of interactions required to unambiguously determine the user's position. On the other hand, the process dynamically reacts to the answers that the user provides. Each answer is analyzed before the next question is generated in order to assure that the following answers allow for an optimal reduction of the search space and the length of the remaining dialog.

- **Generation of optimized localizations through frames of reference induced by meta-communicative acts**
  A second distinguishing feature of SISTO consists of its ability to generate localizations using an induced frame of reference (see 4.1.2). This enables SISTO to better cope with spatial constellations, where there is no combination of a relation and an anchor object that precisely describes the location of a target object. In these cases, previous systems often resorted to other means such as metrical descriptions or combined relations. SISTO however allows for the modification of the frame of reference in such a situation, thereby not only providing a further means of adaptation but also a way to optimize localizations in general: Even if a localization is already fairly precise, the use of an induced frame of reference allows for a further improvement of the corresponding utterance. Hence, SISTO can generate very precise localizations even when faced with 'difficult' constellations that do not afford 'simple' localizations.

7.2 Opportunities for further research

Since the approach that we propose in this thesis is specifically designed to be flexible and extensible, it facilitates extensions on various levels. In addition to these extensions, the interdisciplinary field of mobile assistance provides several promising areas for further research that became apparent during the work on this thesis. In the following, we will shortly review some key future research directions that can build upon the results of our work.

7.2.1 Empirical evaluation

While the model we proposed supports the inclusion of empirical findings (for example, through the task-dependent adjustment of weights), a complete investigation of the impact that each situational factor has on each basic process was beyond the scope of this thesis. This has only been partially addressed in the context of the computation of two- and n-point relations. Obviously, a broad empirical study on this for all basic processes would greatly benefit the practical application of the model. But even partial results can incrementally be incorporated as they become available – such as the results from the study reported in the appendix, which informed the process of modeling the path relations. In addition, empirical research in this direction will most certainly yield further situational factors to be included into the reasoning process, which would also contribute to the quality of applications based on our model.

Since there now is a prototypical implementation, it would also be very interesting to evaluate it in terms of user friendliness and to determine the performance of users on typical tasks compared to scenarios without an assistive system or with only a paper 'assistant'. Not only would the results from such a study help to further improve the system but it would probably also yield further relevant situational factors. Additionally, a field test could contribute to the identification of the relative importance of the various situational factors that the system already considers during computation.

7.2.2 Extension of the model and further applications

The model itself can be extended in several ways: On the one hand, it naturally lends itself to the inclusion of further situational factors. On the other hand, the modeling of additional complex tasks based on the ones presented and the basic processes described in 4.1 can help to get further insights in terms of relevant factors and the inherent interactions required to perform these tasks. We expect similar results from an application of our model to other scenarios such as robot navigation or tour planning.

As we have mentioned in section 4.3.3, tour planning is a complex and computationally demanding process. In order to generate tours that match for example the user's preferences and abilities as well as her current situation, numerous factors have to be taken into account in addition to purely geometrical ones such as the number of turns or the overall length. The model we introduced in this thesis (and parts of the prototypical implementation) can help to address several issues in this context. One way to improve on 'regular' tour planning lies in the incremental evaluation of potential tours using a process similar to the one described in 4.3.3. For example, the evaluation could then include additional factors such as the number of suitable anchor objects near

the route, the quality of instructions for this route, or the number and shape of the corresponding segments.

There is another interesting way to exploit the modular character of the model: Since the basic processes are separated conceptually as well as on the implementation level, it is easy to compare competing models for the realization of those processes in terms of their practical performance. For example, by replacing the computational approach for the establishment of two-point relations we presented in 4.1.3 with RCC-8 [Randell et al., 1992] we can evaluate the performance of these models against a set of tasks that the user has to perform. The results of such a study can not only help to select the mechanism best suited for a given purpose but may also yield valuable information for the improvement of adaptation strategies and the development of further ones.

7.2.3 Comprehensive model of positional information

The work presented in this thesis does include several relevant aspects in terms of handling and classifying positional information. In addition, we introduced a comprehensive model as well as a generic architecture for the handling of positional information. However, there are a few important points where the model and implementation could be improved. The modeling of sensors that gather positional information is one key aspect that could help to vastly improve the robustness of an approach against problems such as sensor outages. Among other advantages, a sensor model would enable the system to derive information from the fact that a sensor is failing or returning impossible values. For example, if the GPS looses contact with all satellites simultaneously, we might conclude that the user either entered a building or a narrow alley. This knowledge could help to significantly reduces the number of locations, where the user may possible be. (This example is a simplification for illustration purposes: a system would have to consider the possibility of a sensor failure as well, e. g. a broken cable.) A further extension of the model for positional information is also related to the inclusion of sensor models: If we dispose of those models and a history of sensor data, multiple sensors can be used to compensate missing information from other sensors. Additionally, this allows for the explicit reasoning about sensor readings, which, for example, may help to detect false readings and possibly even to correct these.

7.2.4 Analysis of human-computer collaboration in a mobile setting

In the context of position determination, we designed an approach that relies on interaction with the user in case the system is unable to compute the user's current location with sufficient precision and/or confidence. Especially in a mobile setting (such as new location based services evolving in the context of UMTS), this paradigm is a promising approach to address problematic situations, where today many systems simply fail to provide their service. Since mobile devices are inherently more limited in terms of computational resources (such as bandwidth, computing power or memory), this approach allows for the compensation of some problems such as the loss of the connection to a server. Generally, when a system is faced with a (sub-)problem it can solve with the information at hand, interacting with the user is a means to avoid failure that is rarely seen in current intelligent system. A more general theory of collaborative robustness through interaction may help to improve this situation and enable more user-friendly systems. In this context, the benefits of providing a service despite missing information have to be weighted against the increase

in interaction. An analysis of the corresponding trade-offs is of general interest in the context of intelligent interactive systems.

Our approach for interactively determining the user's current position can also be extended in several ways. Especially in non-urban environment it could be beneficial to allow for more open questions such "What can you see?". A further promising extension lies in exploiting the camera that is integrated in many current mobile phones: by analyzing snapshots of the user's environment, the system may be able to infer her current environment through a combination of interaction and image recognition. In addition, a probabilistic modeling of the user's replies can help to address the problem of oversight: if the visibility matrix consisted of probabilities instead of crisp values, overlooking sights would only result in decreasing their probability instead of eliminating them entirely. Consequently, this modification would enable the system to recover from accidentally false replies such as "I cannot see X." while standing in front of X.

7.2.5 Plan recognition through position tracking

Positional information, its determination and handling was one main topic of this thesis. The comprehensive approach we presented in this context not only includes various adaptation strategies but also a generic architecture for managing positional information. Especially in ubiquitous and mobile computing or in intelligent environments, knowing where the user is (and was) can help to infer what her current intention or plan is, and where she may want to go next. For example, if someone is constantly entering and leaving shops of certain type (e. g. book stores), a system could conclude that she is looking for a book. Based on this assumption, it could then proactively support the user, e. g. by showing a map of book stores in her current environment and highlighting those that she has not been to yet.

Our approach already provides the basic components to realize such a scenario: the three level architecture not only includes direct sensor data but also a position history as well as advanced inferential services to determine the user's current position. The spatiotemporal trigger concept we presented in section 6.5 allows for an abstract description of regions as well as events associated with them – such as entering, leaving, or remaining there. In addition, a plan recognition component could also contribute to inferring the user's current position by providing hypotheses about where she may want to go next.

Bibliography

[Abowd et al., 1997] Abowd, G., Atkeson, C., Hong, J., Long, S., Kooper, R., and Pinkerton, M. (1997). Cyberguide: A mobile context-aware tour guide. *ACM Wireless Networks*, (3):421–433.

[Allen, 1981] Allen, G. L. (1981). A developmental perspective on the effect of "subdividing" macrospatial experience. *Journal of Experimental Psychology: Human Learning and Memory*, 7:120–132.

[André, 1995] André, E. (1995). *Ein planbasierter Ansatz zur Generierung multimedialer Präsentationen*, volume 108 of *DISKI - Dissertationen zur Künstlichen Intelligenz*. Infix, St. Augustin.

[André et al., 1986] André, E., Bosch, G., Herzog, G., and Rist, T. (1986). Characterizing trajectories of moving objects using natural language path descriptions. In *Proceedings of the 7th European Conference on Artificial Intelligence (ECAI)*, volume 2, pages 1–8, Brighton, UK.

[André and Rist, 1995] André, E. and Rist, T. (1995). Generating coherent presentations employing textual and visual material. *AI Review*, 9(2/3):147–165.

[Anegg et al., 2002] Anegg, H., Kunczier, H., Michlmayr, E., Pospischil, G., and Umlauft, M. (2002). LoL@: designing a location based UMTS application. *Elektrotechnik und Informationstechnik*, 119(2):48–51.

[Asthana et al., 1994] Asthana, A., Cravatts, M., and Krzyzanowski, P. (1994). An indoor wireless system for personalized shopping assistance. In *IEEE Workshop on Mobile Computing Systems and Applications*, Santa Cruz, CA, US.

[Austin, 1962] Austin, J. L. (1962). *How to Do Things with Words*. Oxford University Press, Oxford.

[Baddeley, 1986] Baddeley, A. D. (1986). *Working Memory*. Oxford University Press, Oxford.

[Baus, 2002] Baus, J. (2002). *Ressourcenadaptierende hybride Personennavigation*. PhD thesis, Computer Science Department, University of Saarland, Saarbrücken. (To appear in the DISKI series in 2003).

[Baus et al., 1999] Baus, J., Butz, A., and Krüger, A. (1999). One way interaction: interactivity over unidirectional links. In *Proceedings of I3 Workshop on Adaptive Design of Interactive Multimedia Presentations for Mobile Users*.

[Baus et al., 2002] Baus, J., Krüger, A., and Wahlster, W. (2002). A resource-adaptive mobile navigation system. In Gil, Y. and Leake, D. B., editors, *IUI 02 – 2002 International Conference on Intelligent User Interfaces*, pages 15–22, San Francisco, California. ACM Press.

[Benelli et al., 1999] Benelli, G., Bianchi, A., Marti, P., Not, E., and Sennati, D. (1999). HIPS: Hyper-interaction within physical space. In *Proceedings of IEEE Multimedia Systems '99*, pages 1075–1078, Florence, Italy. IEEE.

[Berendt, 1999] Berendt, B. (1999). *Representation and precessing of knowledge about distances in environmental spaces: A computational model of inferred route distances investigating their qualitative and quantitative determinants*, volume 197 of *DISKI - Dissertationen zur Künstlichen Intelligenz*. Infix, Sankt Augustin.

[Berendt et al., 1998] Berendt, B., Barkowsky, T., Freksa, C., and Kelter, S. (1998). Spatial representation with aspect maps. In Freksa, C., Habel, C., and Wender, K. F., editors, *Spatial Cognition*, volume 1404 of *LNCS*, pages 313–336, Berlin, Heidelberg, New York. Springer.

[Bianchi and Zancanaro, 1999] Bianchi, A. and Zancanaro, M. (1999). Tracking user's movements in an artistic physical space. In *Proceedings of I3 Annual Conference*, pages 103–106, Siena.

[Blocher, 1999] Blocher, A. (1999). *Ressourcenadaptierende Raumbeschreibung: Ein beschränkt-optimaler Lokalisationsagent*. PhD thesis, Computer Science Departement, University of Saarland, Saarbrücken, Germany.

[Booch, 1994] Booch, G. (1994). *Object-oriented analysis and design with applications*. Addison Wesley, Boston, U.S.A.

[Bosch i Creus, 2002] Bosch i Creus, G. (2002). 3D environments in wireless information devices. Master's thesis, Espoo-Vantaa Institute of Technology, Espoo.

[Bratman et al., 1988] Bratman, M. E., Israel, D., and Pollack, M. (1988). Plans and resource-bounded practical reasoning. *Computational Intelligence*, 4(4):349–355.

[Bronstein and Semendjajew, 1979] Bronstein, I. N. and Semendjajew, K. A. (1979). *Taschenbuch der Mathematik*. Verlag Harri Deutsch, Thun and Frankfurt am Main, reprint of the 20th edition edition.

[Butz et al., 2001] Butz, A., Baus, J., Krüger, A., and Lohse, M. (2001). A hybrid indoor navigation system. In *Proceedings of IUI 2001*, pages 25–32, New York. ACM Press.

[CARC, 2002] CARC (2002). Web page of the Cyber Assist Research Center (CARC). http://www.carc.aist.go.jp.

[Carroll, 1946] Carroll, L. (1946). *Alice in Wonderland*. Grosset and Dunlap.

[Casakin et al., 2000] Casakin, H., Barkowsky, T., Klippel, A., and Freksa, C. (2000). Schematic maps as wayfinding aids. *Lecture Notes in Computer Science*, 1849:54–71.

[Casner, 1991] Casner, S. M. (1991). Task-analytic approach to the automated design of graphic presentations. *ACM Transactions on Graphics*, 10(2):111–151.

[Cheverst et al., 2000a] Cheverst, K., Davies, N., Friday, A., and Efstratiou, C. (2000a). Developing a context-aware electronic tourist guide: Some issues and experiences. In *Proceedings of CHI 2000*, pages 17–24, Netherlands.

[Cheverst et al., 2002] Cheverst, K., Davies, N., and Mitchell, K. (2002). Exploring context-aware information push. *Personal and Ubiquitous Computing*, 6(4):276–281.

[Cheverst et al., 2000b] Cheverst, K., Davies, N., Mitchell, K., and Efstratiou, C. (2000b). Using context as a crystal ball: Rewards and pitfalls. In *Proceedings of Workshop on 'Situated Interaction in Ubiquitous Computing' at CHI 2000*, pages 12–16.

[Cheverst et al., 2000c] Cheverst, K., Davies, N., Mitchell, K., and Friday, A. (2000c). Experiences of developing and deploying a context-aware tourist guide: the GUIDE project. In *Mobile Computing and Networking*, pages 20–31, Boston, U.S.A.

[Cheverst et al., 2000d] Cheverst, K., Davies, N., Mitchell, K., and Smith, P. (2000d). Providing tailored (context-aware) information to city visitors. In Brusilovsky, P., Stock, O., and Strapparava, C., editors, *Adaptive Hypermedia and Adaptive Web-Based Systems, International Conference, AH 2000, Trento, Italy*, pages 73–85, Berlin, Heidelberg, New York. Springer.

[Cheverst et al., 1999] Cheverst, K., Mitchell, K., and Davies, N. (1999). Design of an object model for a context sensitive tourist guide GUIDE. *Computers and Graphics*, 23(6):883–891.

[Cohn, 1996] Cohn, A. G. (1996). Calculi for qualitative spatial reasoning. In Calmet, J., Campbell, J. A., and Pfalzgraf, J., editors, *Artificial Intelligence and Symbolic Mathematical Computation*, volume LNCS 1138, pages 124–143, Berlin, Heidelberg, New York. Springer.

[Cormen et al., 1989] Cormen, T. H., Leiserson, C. E., and Rivest, R. L. (1989). *Introduction to Algorithms*. MIT Press.

[Dey and Abowd, 1999] Dey, A. K. and Abowd, G. D. (1999). Towards a better understanding of context and context-awareness. Technical Report GIT-GVU-99-22, College of Computing, Georgia Institute of Technology, Atlanta, Georgia, U.S.A.

[Ding et al., 2001] Ding, Y., Malaka, R., Kray, C., and Schillo, M. (2001). RAJA: a resource-adaptive Java agent infrastructure. In Müller, J. P., André, E., Sen, S., and Frasson, C., editors, *Proceedings of the Fifth International Conference on Autonomous Agents*, pages 332–339, Montreal, Canada. ACM Press.

[Downs and Stea, 1977] Downs, R. and Stea, D. (1977). *Maps in Minds: Reflections on Cognitive Mapping*. Harper and Row, New York.

[Egenhofer and Franzosa, 1991] Egenhofer, M. and Franzosa, R. (1991). Point-set topological spatial relations. *International Journal of Geographical Information Systems*, 5(2):161–174.

[Erman et al., 1980] Erman, L. D., Hayes-Roth, F., Lesser, V. R., and Reddy, D. R. (1980). The Hearsay-II speech-understanding system: Integrating knowledge to resolve uncertainty. *ACM Computing Surveys*, 12(2):213–253.

[Estrin et al., 2001] Estrin, D., Girod, L., Pottie, G., and Srivastava, M. (2001). Instrumenting the world with wireless sensor networks. In *Proc. of International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2001)*, pages available online at http://www.isi.edu/scadds/papers/ICASSP–2001.ps (as of April 2003), Salt Lake City, Utah.

[Fillmore, 1982] Fillmore, C. (1982). *Speech, place, and action: Studies in deixis and related topics*, chapter Towards a descriptive framework for spatial deixis, pages 31–59. John Wiley & Sons.

[Fischer, 1999] Fischer, K. (1999). Holonic multiagent systems – theory and applications. In *Proceedings of the 9th Portuguese Conference on Progress in Artificial Intelligence (EPIA-99)*, volume 1695 of *LNAI*, pages 34–48. Springer, Berlin, Heidelberg, New York.

[Fontaine and Denis, 1999] Fontaine, S. and Denis, M. (1999). The production of route instructions in underground and urban environments. In Freksa, C. and Mark, D. M., editors, *Spatial Information Theory (Proceedings of COSIT 99)*, pages 83–94, Berlin, Heidelberg, New York. Springer.

[Frank, 1998] Frank, A. U. (1998). Formal models for cognition - taxonomy of spatial location description and frames of reference. In Freksa, C., Habel, C., and Wender, K., editors, *Spatial cognition – An interdisciplinary approach to representation and processing of spatial knowledge*, pages 293–312. Springer, Berlin, Heidelberg, New York.

[Freksa, 1992] Freksa, C. (1992). Temporal reasoning based on semi-intervals. *Artificial Intelligence*, 54(1):199–227.

[Freksa, 1997] Freksa, C. (1997). *Spatial and Temporal Structures in Cognitive Processes*, pages 379–387. LNCS 1337. Springer, Berlin, Heidelberg, New York.

[Freksa et al., 2000] Freksa, C., Moratz, R., and Barkowsky, T. (2000). *Spatial Cognition II: Integrating Abstract Theories, Empirical Studies, Formal Methods, and Practical Applications*, volume 1849 of *LNAI*, chapter Schematic Maps for Robot Navigation, pages 100–114. Springer, Berlin, Heidelberg, New York.

[Freundschuh and Egenhofer, 1997] Freundschuh, S. M. and Egenhofer, M. J. (1997). Human conceptions of spaces: Implications for geographic information systems. *Transactions in GIS*, 2(4):361–375.

[Gapp, 1994] Gapp, K.-P. (1994). Basic meanings of spatial relations: Computation and evaluation in 3D space. In *Proceedings of AAAI-94*, pages 1393–1398, Seattle, WA.

[Gapp, 1995] Gapp, K.-P. (1995). An empirically validated model for computing spatial relations. In Wachsmuth, I., Rollinger, C.-R., and Brauer, W., editors, *KI-95: Advances in Artificial Intelligence. 19th Annual German Conference on Artificial Intelligence*, pages 245–256, Berlin, Heidelberg, New York. Springer.

[Gapp, 1997] Gapp, K.-P. (1997). *Objektlokalisation: ein System zur sprachlichen Raumbeschreibung*. Kognitionswissenschaft. Deutscher Universitäts Verlag, Wiesbaden.

[Grice, 1975] Grice, H. P. (1975). Logic and conversation. In Cole, P. and Morgan, J. L., editors, *Syntax and Semantics, 3: Speech Acts*, pages 41–58. Academic Press, New York.

[Guhe et al., 2000] Guhe, M., Habel, C., and Tappe, H. (2000). Incremental event conceptualization and natural language generation in monitoring environments. In *Proceedings of the First International Conference on Natural Language Generation (INLG)*, pages 85–92.

[Habel, 1989] Habel, C. (1989). Zwischen-Bericht. In Habel, C., Herweg, M., and Rehkämpfer, K., editors, *Interdisziplinäre Beiträge zu Sprache und Raum*, pages 37–69. Niemeyer, Tübingen.

[Hanßmann, 1980] Hanßmann, K.-J. (1980). Sprachliche Bildinterpretation für ein Frage-Antwort-System. Technical Report Ifi-hh-m-74/80, FB Informatik, Universität Hamburg.

[Harter and Hopper, 1994] Harter, A. and Hopper, A. (1994). A distributed location system for the active office. *IEEE Network Magazine*, 8(1):62–70.

[Herrman and Grabowski, 1994] Herrman, T. and Grabowski, J. (1994). *Sprechen – Psychologie der Sprachproduktion*. Spektrum Akademischer Verlag, Berlin, Heidelberg.

[Herskovits, 1986] Herskovits, A. (1986). *Language and Spatial Cognition - An Interdisciplinary Study of the Prepositions in English*. Cambridge University Press, Cambridge, UK.

[Hohl et al., 1999] Hohl, F., Kubach, U., Leonhardi, A., Rothermel, K., and Schwehm, M. (1999). Next century challenges: Nexus - an open global infrastructure for spatial-aware applications. In *Proceedings of Mobicom '99*, pages 249–255, Seattle, WA. ACM.

[Ittelson, 1973] Ittelson, W. (1973). *Environment and Cognition*, chapter Environment perception and contemporary perceptual theory, pages 1–19. Seminar Press, New York.

[Jameson, 2001] Jameson, A. (2001). Modeling both the context and the user. *Personal Technology*, 5(1):29–33.

[Jameson, 2002] Jameson, A. (2002). Usability issues and methods for mobile multimodal systems. In *Proceedings of the ISCA Tutorial and Research Workshop on Multi-Modal Dialogue in Mobile Environments*, page available online at http://dfki.de/~jameson/pdf/ids02.jameson.pdf (as of April 2003), Kloster Irsee, Germany. Summary of a keynote address.

[Jameson and Buchholz, 1998] Jameson, A. and Buchholz, K. (1998). Einleitung zum Themenheft "Ressourcenadaptive kognitive Prozesse" [Introduction to the special issue on "Resource-Adaptive Cognitive Processes"]. *Kognitionswissenschaft*, 7:95–100.

[Jansen-Osmann, 1998] Jansen-Osmann, P. (1998). *Kognition von Distanzen - laborexperimentelle Untersuchungen in virtuellen Umgebungen (distance cognition - laboratory-experimental investigations in virtual environments)*. PhD thesis, University of Duisburg, Department of Educational Science, Duisburg.

[JASSS, 2003] JASSS (2003). The Journal of Artificial Societies and Social Simulation. http://jasss.soc.surrey.ac.uk/JASSS.html.

[Jenkins et al., 2000] Jenkins, L., Myerson, J., Joerding, J. A., and Hale, S. (2000). Converging evidence that visualspatial cognition is more age-sensitive than verbal cognition. *Psychology and Aging*, 15(1):157–175.

[Jennings, 1999] Jennings, N. R. (1999). Agent-based computing: Promise and perils. In Dean, T., editor, *Proceedings of IJCAI 1999*, pages 1429–1436. Morgan Kaufmann.

[Jennings, 2000] Jennings, N. R. (2000). On agent-based software engineering. *Artificial Intelligence*, 177(2):277–296.

[Kainz et al., 1993] Kainz, W., Egenhofer, M. J., and Greasley, I. (1993). Modeling spatial relations and operations with partially ordered sets. *International Journal of Geographical Information Systems*, 7(3):215–229.

[Keeney et al., 1976] Keeney, R. L., Raiffa, H., and Meyer, R. F. (1976). *Decisions with multiple objectives*. Wiley, New York.

[Kirasic, 2000] Kirasic, K. C. (2000). Age differences in adults' spatial abilities, learning envrionmental layout, and wayfinding behavior. *Spatial Cognition and Computation*, (2):117–134.

[Klabunde et al., 1999] Klabunde, K., Glatz, D., and Porzel, R. (1999). *Representations and Processes in Language Production*, chapter An Anatomy of a Spatial Description, pages 89–116. Deutscher Universitäts Verlag, Wiesbaden.

[Klatzky, 1998] Klatzky, R. (1998). Allocentric and Egocentric Spatial Representations: Definitions, Distinctions, and Interconnections. In Freksa, C., Habel, C., and Wender, K., editors, *Spatial cognition – An interdisciplinary approach to representation and processing of spatial knowledge*, pages 1–17. Springer, Berlin, Heidelberg, New York.

[Knauff et al., 2002] Knauff, M., Schlieder, C., and Freksa, C. (2002). From rat-research to multifunctional spatial assistance systems. *Künstliche Intelligenz*, 4(16):5–9.

[Kobsa et al., 1986] Kobsa, A., Allgayer, J., Reithinger, N., Harbusch, K., and Wahlster, W. (1986). Combining deictic gestures and natural language for referent identification. In *Proceedings of the 11th International Conference on Computational Linguistics*, pages 356–361, Bonn, Germany.

[Kobsa and Wahlster, 1989] Kobsa, A. and Wahlster, W., editors (1989). *User Models in Dialog Systems*. Springer, Berlin, Heidelberg, New York.

[Kray, 1998] Kray, C. (1998). Ressourcenadaptierende Verfahren zur Präzisionsbewertung von Lokalisationsausdrücken und zur Generierung von linguistischen Hecken. Master's thesis, Computer science departement, University of Saarland, Saarbrücken.

[Kray et al., 2001] Kray, C., Baus, J., Zimmer, H., Speiser, H., and Krüger, A. (2001). Two path prepositions: along and past. In Montello, D. R., editor, *Spatial Information Theory (Proceedings of COSIT'01)*, pages 263–277, Berlin, Heidelberg, New York. Springer.

[Kray and Blocher, 1999] Kray, C. and Blocher, A. (1999). Modeling the basic meanings of path relations. In *Proceedings of the 16th IJCAI. Morgan Kaufmann, San Francisco, CA*, pages 384–389.

[Kray et al., 2003] Kray, C., Laakso, K., Elting, C., and Coors, V. (2003). Presenting route instructions on mobile devices. In Johnson, W. L., André, E., and Domingue, J., editors, *Proceedings of IUI 03*, pages 117–124, Miami Beach, FL. ACM Press.

[Kray et al., 2002] Kray, J., Li, K. Z. H., and Lindenberger, U. (2002). Age-specific changes in task-switching components: The role of task uncertainty. *Brain and Cognition*, (49):363–381.

[Krieg-Brückner and Röfer, 1998] Krieg-Brückner, B. and Röfer, T. (1998). A taxonomy of spatial knowledge for navigation and its application to the Bremen autonomous wheelchair. In Freksa, C., Habel, C., and Wender, K. F., editors, *Spatial Cognition – An interdisciplinary approach to representation and processing of spatial knowledge*, pages 373–398, Berlin, Heidelberg, New York. Springer.

[Krüger, 2000] Krüger, A. (2000). *Automatische Abstraktion in 3D-Graphiken*. Number 232 in DISKI - Dissertationen zur Künstlichen Intelligenz. Akademische Verlags-Gesellschaft AKA, Berlin.

[Krüger and Maaß, 1997] Krüger, A. and Maaß, W. (1997). Towards a computational semantics of path relations. In *Proceedings of the Workshop "Language and Space" at AAAI'97*, pages 101–109, Providence, USA.

[Laakso, 2002] Laakso, K. (2002). Evaluating the use of navigable three-dimensional maps in mobile devices. Master's thesis, Helsinki University of Technology, Helsinki.

[Lakoff, 1973] Lakoff, G. (1973). Heges: A study of meaning criteria and the logic of fuzzy concepts. *Journal of Philosophical Logic*, 2:458–508.

[Lankenau and Röfer, 2002] Lankenau, A. and Röfer, T. (2002). Mobile robot self-localization in large-scale environments. In *Proceedings of the IEEE International Conference on Robotics and Automation 2002 (ICRA-2002)*, pages 1359–1364, Washington, DC. IEEE.

[Lee et al., 2000] Lee, B.-S., Cai, W., Turner, S. J., and Chen, L. (2000). Adaptive dead reckoning algorithms for distributed interactive simulation. *International Journal of Simulation*, 1(1–2):21–34.

[Leśniewski, 1931] Leśniewski, S. (1927–1931). O podstawach matematyki (on the foundations of mathematics). *Przeglad Filozoficzny (Philosophical Review)*, (30–34).

[Levasseur and Veron, 1991] Levasseur, M. and Veron, E. (1991). *Ethnographie de l'exposition*. BPI/Centre Pompidou, 2e édition edition.

[Levelt, 1989] Levelt, W. J. M. (1989). *Speaking: from intention to articulation*. MIT Press, Cambridge, MA.

[Long et al., 1995] Long, A. C., Narayanaswamy, S., Burstein, A., Han, R., Lutz, K., Richards, B., Sheng, S., Brodersen, R. W., and Rabaey, J. (1995). A prototype user interface for a mobile multimedia terminal. In *Conference companion on Human factors in computing systems*, pages 81–82, New York. SIGCHI, ACM Press.

[Long et al., 1996] Long, S., Kooper, R., Abowd, G. D., and Atkeson, C. G. (1996). Rapid prototyping of mobile context-aware applications: The Cyberguide case study. In *Mobile Computing and Networking*, pages 97–107.

[Lynch, 1960] Lynch, K. (1960). *The Image of the City*. MIT Press, Cambridge, MA.

[Maaß, 1999] Maaß, W. (1999). *Von visuellen Daten zu inkrementellen Wegbeschreibungen in dreidimensionalen Umgebungen*, volume 214 of *DISKI - Dissertationen zur Künstlichen Intelligenz*. infix, St. Augustin.

[Maaß et al., 1995] Maaß, W., Baus, J., and Paul, J. (1995). Visual grounding of route descriptions in dynamic environments. In *Proc. of AAAI Fall Symposium on Computational Models for Integrating Language and Vision*, MIT, Cambridge, MA. AAAI.

[Maaß and Schmauks, 1998] Maaß, W. and Schmauks, D. (1998). MOSES: Ein Beispiel für die Modellierung räumlicher Leistungen durch ein Wegbeschreibungssystem. *Zeitschrift für Semiotik 20*, pages 91–103.

[Malaka and Zipf, 2000] Malaka, R. and Zipf, A. (2000). Deep Map - challenging IT research in the framework of a tourist information system. In Fesenmaier, D. R., Klein, S., and Buhalis, D., editors, *Information and communication technologies in tourism 2000*, pages 15–27. Springer, Berlin, Heidelberg, New York.

[Marr, 1982] Marr, D. (1982). *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. W. H. Freeman and Co., New York.

[Masolo and Vieu, 1999] Masolo, C. and Vieu, L. (1999). Atomicity vs. infinite divisibility in space. In Freksa, C. and Mark, D. M., editors, *Spatial Information Theory (Proceedings of COSIT 99)*, pages 235–250, Berlin, Heidelberg, New York. Springer.

[Maybury and Wahlster, 1998] Maybury, M. T. and Wahlster, W. (1998). *Readings in Intelligent User Interfaces*, chapter Intelligent User Interfaces: An Introduction, pages 1–14. Morgan Kaufman Press, San Francisco, CA.

[Mayfield et al., 1996] Mayfield, J., Labrou, Y., and Finin, T. (1996). *Intelligent Agents II*, chapter Evaluating KQML as an agent communication language, pages 347–360. LNAI 1037. Springer, Berlin, Heidelberg, New York.

[McCarthy and Buvač, 1998] McCarthy, J. and Buvač, S. (1998). Formalizing context (expanded notes). In Aliseda, A., van Glabbek, R. J., and Westerståhl, D., editors, *Computing Natural Language*, volume 81 of *CSLI Lecture Notes*, pages 13–50. Center for the Study of Language and Information, Standford University, Stanford, CA.

[Merriam Webster, 2002] Merriam Webster (2002). Online dictionary. http://www.m-w.com.

[Montello, 1993] Montello, D. R. (1993). Scale and multiple phychologies of space. In Frank, A. and Campari, I., editors, *Spatial Information Theory: A theoretical basis for GIS*, pages 312–321, Berlin, Heidelberg, New York. Springer.

[Mori, 2003] Mori, A. (2003). Toward an open platform for pervasive computing environment - UBKit. Talk at the Cyber Assist Consortium Second International Symposium.

[Mukerjee, 1997] Mukerjee, A. (1997). *Representation and Processing of Spatial Expressions*, chapter Neat vs Scruffy: A Review of Computational Models for Spatial Expressions, pages 1–36. Lawrence Erlbaum Associates.

[Nishimura et al., 2002] Nishimura, T., Itoh, H., Yamamoto, Y., and Nakashima, H. (2002). A compact battery-lesss information terminal (CoBIT) for location-based support systems. In *Procceedings of SPIE 2002*, pages 80–86.

[Not et al., 1998] Not, E., Petrelli, D., Sarini, M., Stock, O., Strapparava, C., and Zancanaro, M. (1998). Hypernavigation in the physical space: Adapting presentations to the user and to the situational context. *The New Review of Hypermedia and Multimedia*, 4:33–45.

[Not et al., 2000] Not, E., Petrelli, D., Stock, O., Strapparava, C., and Zancanaro, M. (2000). The environment as a medium: Location-aware generation for cultural visitors. In *Proceedings of the Workshop on Coherence in Generated Multimedia (held in conjunction with INLG-2000)*, Mitzpeh Ramon, Israel. (also available as IRST-Technical Report 001-17, November 2000).

[Oppermann and Specht, 1999] Oppermann, R. and Specht, M. (1999). A nomadic information system for adaptive exhibition guidance. In *International Cultural Heritage Meeting (ICHIM99)*, pages 103–109, Washington D.C.

[Oppermann et al., 1999] Oppermann, R., Specht, M., and Jaceniak, I. (1999). Hippie: A nomadic information system. *Lecture Notes in Computer Science*, 1707:330–333.

[PEACH, 2003] PEACH (2003). Personal experience with active cultural heritage. http://peach.itc.it/home.html.

[Picard, 1997] Picard, R. W. (1997). *Affective Computing*. MIT Press, Cambridge, MA.

[Porzel et al., 2002] Porzel, R., Meyer-Klabunde, R., and Jansche, M. (2002). *Spatial Language. Cognitive and Computational Perspectives*, chapter Generating Spatial Descriptions from a Cognitive Point of View, pages 185–208. Kluwer Academic, Dordrecht.

[Poslad et al., 2001] Poslad, S., Laamanen, H., Malaka, R., Nick, A., Buckle, P., and Zipf, A. (2001). CRUMPET: Creation of user-friendly mobile services personalised for tourism. In *Proceeding of 3G 2001 - Second international conference on 3G mobile communication technologies*, pages 28–32, London, UK.

[Pospischil et al., 2002] Pospischil, G., Umlauft, M., and Michlmayr, E. (2002). Designing LoL@, a Mobile Tourist Guide for UMTS. In Paterno, F., editor, *Proceedings of Mobile Human-Computer Interaction 2002*, pages 140–154, Berlin, Heidelberg, New York. Springer.

[Preist et al., 2001] Preist, C., Byde, A., and Bartolini, C. (2001). Economic dynamics of agents in multiple auctions. In Müller, J. P., André, E., Sen, S., and Frasson, C., editors, *Proceedings of the Fifth International Conference on Autonomous Agents*, pages 545–551, Montreal, Canada.

[Priyantha et al., 2000] Priyantha, N. B., Chakraborty, A., and Balakrishnan, H. (2000). The Cricket location-support system. In *Proc. of the Sixth Annucal ACM International Conference on Mobile Computing and Networking*, pages 32–43, New York. ACM Press.

[Randell et al., 1992] Randell, D., Cui, Z., and Cohn, A. (1992). A spatial logic based on regions and connection. In *Proc. 3rd Int. Conf. on Knowledge Representation and Reasoning*, pages 165–176, San Mateo. Morgan Kaufmann.

[Raubal and Worboys, 1999] Raubal, M. and Worboys, M. (1999). A formal model for the process of wayfinding in built environments. In Freksa, C. and Mark, D. M., editors, *Spatial Information Theory (Proceedings of COSIT 99). Springer, Berlin*, pages 381–399, Berlin, Heidelberg, New York. Springer.

[Renz and Nebel, 1999] Renz, J. and Nebel, B. (1999). On the complexity of qualitative spatial reasoning: A maximal tractable fragment of the region connection calculus. *Artificial Intelligence*, 1–2(108):448–455.

[Richter, 2001] Richter, K.-F. (2001). Ein Verfahren zur Bestimmung geeigneter Standorte für You-Are-Here-Karten in Außengeländen. Master's thesis, Arbeitsbereich Wissens- und Sprachverarbeitung, Fachbereich Informatik, Universität Hamburg, Hamburg.

[Rothermel, 2003] Rothermel, K. (2003). SFB "Umgebungsmodelle für mobile kontextbezogene Systeme". http://www.nexus.uni-stuttgart.de.

[Sadalla and Magel, 1980] Sadalla, E. K. and Magel, S. G. (1980). Perception of traversed distances. *Environment and Behavior*, 12:65–79.

[Sashima and Kurumantani, 2002] Sashima, A. and Kurumantani, K. (2002). Seamless context-aware information assists based on multiagent cooperation. In *Proc. of The second International Workshop on Agent-based Approaches in Economic and Social Complex Systems (AESCS02)*, pages 99–46, Tokyo.

[Schäfer, 2001] Schäfer, R. (2001). Rules for using multi-attribute utility theory for estimating a user's interests. In Henze, N., editor, *Online Proceedings of ABIS-Workshop 2001*, pages available online at http://www.kbs.uni–hannover.de/~henze/ABIS_Workshop2001/final/Schaefer_final.pdf (as of April 2003), Dortmund.

[Schirra, 1994] Schirra, J. (1994). *Bildbeschreibung als Verbindung von visuellem und sprachlichem Raum – Eine interdisziplinäre Untersuchung von Bildvorstellungen in einem Hörermodell*, volume 71 of *DISKI - Dissertationen zur Künstlichen Intelligenz*. Infix, St. Augustin, available online (as of April 2003) at http://www.computervisualistik.de/~schirra, English edition.

[Schmauks, 1998] Schmauks, D. (1998). Kognitive und semiotische Ressourcen für die Wegfindung. *Kognitionswissenschaft, Sonderheft zum Sonderforschungsbereich (SFB) 378*, 7(3):124–129.

[Schmidt et al., 1999] Schmidt, A., Beigl, M., and Gellersen, H.-W. (1999). There is more to context than location. *Computers and Graphics*, 23(6):893–901.

[Schweizer et al., 1998] Schweizer, K., Herrmann, T., Janzen, G., and Katz, S. (1998). The route direction effect and its constraints. In Freksa, C., Habel, C., and Wender, K., editors, *Spatial Cognition – An interdisciplinary approach to representation and processing of spatial knowledge*, pages 19–38. Springer, Berlin, Heidelberg, New York.

[Searle, 1969] Searle, J. R. (1969). *Speech Acts: An Essay in the Philosophy of Language*. Cambridge University Press, Cambridge.

[SmartKom Research Consortium, 2002] SmartKom Research Consortium (2002). SmartKom. http://www.smartkom.org.

[Sorrows and Hirtle, 1999] Sorrows, M. E. and Hirtle, S. C. (1999). The nature of landmarks in real and electronic spaces. In Freksa, C. and Mark, D. M., editors, *Spatial Information Theory (Proceedings of COSIT 99)*, pages 37–50, Berlin, Heidelberg, New York. Springer.

[Specht and Oppermann, 1999] Specht, M. and Oppermann, R. (1999). User modeling and adaptivity in nomadic information systems. In *Proceedings of the 7th GI-Workshop "Adaptivität und Benutzermodellierung in interaktiven Softwaresystemen" (ABIS)*, pages available online at http://www–mmt.inf.tu–dresden.de/projekte/TELLIM/Members/Joerding/abis99/EndPaper/specht_18.ps (as of April 2003), Magdeburg.

[Sun Microsystems, Inc., 2002] Sun Microsystems, Inc. (2002). Java Servlet Technology. http://java.sun.com/products/servlet/index.html.

[The Foundation for Intelligent Physical Agents, 2002] The Foundation for Intelligent Physical Agents (2002). FIPA Specifications. http://www.fipa.org/repository/index.html.

[The UMTS Forum, 2003] The UMTS Forum (2003). Website of the UMTS Forum. http://www.umts-forum.org.

[Tolman, 1948] Tolman, E. C. (1948). Cognitive maps in rats and men. *Psychological Review*, (55):189–208.

[Tversky, 1993] Tversky, B. (1993). Cognitive Maps, Cognitive Collages, and Spatial Mental Models. In *Spatial Information Theory. A Theroretical Basis for GIS, COSIT'93*, pages 14–24.

[Tversky and Lee, 1998] Tversky, B. and Lee, P. U. (1998). How Space Structures Language. In Freksa, C., Habel, C., and Wender, K., editors, *Spatial cognition – An interdisciplinary approach to representation and processing of spatial knowledge*, pages 157–177. Springer, Berlin, Heidelberg, New York.

[Tversky and Lee, 1999] Tversky, B. and Lee, P. U. (1999). Pictural and Verbal Tools for Conveying Routes. In Freksa, C. and Mark, D., editors, *COSIT 99: Spatial Information theory: cognitive and computational foundations of geographic information science*, pages 37–50, Berlin, Heidelberg, New York. Springer.

[von Winterfeldt and Edwards, 1986] von Winterfeldt, D. and Edwards, W. (1986). *Decision analysis and behavioral research*. Cambridge University Press, Cambridge, New York, Melbourne.

[Wahlster, 1977] Wahlster, W. (1977). Die Repräsentation von vagem Wissen in natürlichsprachlichen Systemen der Künstlichen Intelligenz. Technical Report 38, Institut für Informatik, Universität Hamburg, Hamburg.

[Wahlster, 1991] Wahlster, W. (1991). *Intelligent User Interfaces*, chapter User and Discourse Models for Multimodal Communication, pages 45–67. ACM Press, New York.

[Wahlster, 2000] Wahlster, W. (2000). *Verbmobil: Foundations of Speech-to-Speech Translation*. Springer, Berlin, Heidelberg, New York.

[Wahlster, 2001] Wahlster, W. (2001). Robust translation of spontaneous speech: A multi-engine approach. In Nebel, B., editor, *Proceedings of IJCAI '01*, pages 1484–1493, San Francisco. Morgan Kaufmann.

[Wahlster, 2002] Wahlster, W. (2002). Smartkom: Fusion and fission of speech, gestures, and facial expressions. In *Proceedings of the 1st International Workshop on Man-Machine Symbiotic Systems*, pages 213–225, Kyoto, Japan.

[Wahlster, 2003] Wahlster, W. (2003). SmartKom: Modality Fusion for a Mobile Companion based on Semantic Web Technologies. In *CyberAssist Consortium Second International Symposium*, pages A–1–A–23, Tokyo.

[Wahlster et al., 2001] Wahlster, W., Reithinger, N., and Blocher, A. (2001). SmartKom: Multimodal communication with a life-like character. In *Proceedings of EuroSpeech 2001*, volume 3, pages 1547 – 1550, Aalborg, Danmark.

[Wahlster and Tack, 1997] Wahlster, W. and Tack, W. (1997). Ressourcenadadaptive kognitive Prozesse. In Jarke, M., Pasedach, K., and Pohl, K., editors, *Informatik 97 – Informatik als Innovationsmotor*, pages 51–57, Berlin, Heidelberg, New York. Springer.

[Want et al., 1992] Want, R., Hopper, A., Falcao, V., and Gibbons, J. (1992). The Active Badge location system. *Transactions on Information Systems*, 10(1):91–102.

[Want et al., 1995] Want, R., Schilit, B., Adams, N., Gold, R., Petersen, K., Goldberg, D., Ellis, J., and Weiser, M. (1995). On overview of the ParcTab ubiquitous computing experiment. *IEEE Personal Communications*, 2(6):28–43.

[Weiss, 1999] Weiss, G. (1999). *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. MIT Press.

[Werner et al., 1997] Werner, S., Krieg-Bruckner, B., Mallot, H. A., Schweizer, K., and Freksa, C. (1997). Spatial cognition: The role of landmark, route, and survey knowledge in human and robot navigation. In Jarke, M., editor, *Informatik '97 GI Jahrestagung*, pages 41–50, Berlin, Heidelberg, New York. Springer.

[Wooldridge, 1997] Wooldridge, M. (1997). Agent-based software engineering. *IEE Proceedings on Software Engineering*, 144(1):26–37.

[Wooldridge and Jennings, 1998] Wooldridge, M. and Jennings, N. R. (1998). Pitfalls of agent-oriented development. In Sycara, K. P. and Wooldridge, M., editors, *Proceedings of the 2nd International Conference on Autonomous Agents (Agents'98)",*, pages 385–391, New York. ACM Press.

[World Wide Web Consortium, 2002a] World Wide Web Consortium, T. (2002a). Extensible Markup Language (XML). http://www.w3.org/XML/.

[World Wide Web Consortium, 2002b] World Wide Web Consortium, T. (2002b). The Extensible Style Sheet Language (XSL). http://www.w3.org/Style/XSL/.

[World Wide Web Consortium, 2002c] World Wide Web Consortium, T. (2002c). HTTP – Hypertext Transfer Procol. http://www.w3.org/Protocols/.

[World Wide Web Consortium, 2002d] World Wide Web Consortium, T. (2002d). Hypertext Markup Language (HTML). http://www.w3.org/MarkUp/.

[Yoshida, 2001] Yoshida, E. (2001). Deictic expressions and discourse segment in English and Japanese task-oriented dialog. In Ito, M., editor, *Online Proceedings of the Theoretical and Applied Linguistics Postgraduate Conference*, page available online at http://www.ling.ed.ac.uk/˜pgc/archive/2001/etsuko01.pdf (as of April 2003), Edinburgh, UK. University of Edinburgh.

[Zadeh, 1965] Zadeh, L. A. (1965). Fuzzy sets. *Information and Control*, 8:338–353.

[Zadeh, 1972] Zadeh, L. A. (1972). A fuzzy-set theoretic interpretation of linguistic hedges. *Journal of Cybernetics*, 3(2):4–34.

[Zilberstein, 1996] Zilberstein, S. (1996). Using anytime algorithms in intelligent systems. *AI Magzine*, (17):73–83.

[Zimmer et al., 2001] Zimmer, H., Speiser, H., Baus, J., and Krüger, A. (2001). Critical features for the selection of verbal description for path relations. *Cognitive Processing*, 2(2–3):389–411.

A An empirical study on path relations

In this appendix, we present the results from an empirical study on the meaning and underlying concepts of two German path prepositions “entlang” (along) and “vorbei” (past). The results informed the process of modeling path relations that we presented in chapter 4.

Within the Collaborative Research Center SFB 378 “Resource-adaptive cognitive processes”, several studies concerning the use of path relations/prepositions were conducted ([Kray et al., 2001, Zimmer et al., 2001]). In the following, we will report on two of these, which investigated the production of the German path prepositions “entlang” (along) and “vorbei” (past) and the corresponding trajectories.

Table A.1 summarizes the experimental setup for experiment one. After the subjects returned the completed set of items, we scanned all the drawings for post-processing and evaluation, and used a custom Java-based software to compute critical parameters and to align and superimpose all drawings for each item (see figure A.1). These parameters characterize the course of the trajectories in three regions: FA, the area in front of (below) the anchor object; NA, the one next to the anchor object, and BA, the area behind (above) the anchor object (cf. figure A.2). We calculated the distance of the trajectory t to the anchor object in discrete steps, and interpolated the area between the trajectory and anchor object in the regions of interest. In the context of the questions that are of interest here, region NA is most relevant: If parallelism and proximity are significant in the case of “along”, the prototypical trajectory drawn to characterize “along” should be closer to the anchor object than the one drawn in the case of “past”, and its distance to the anchor object within NA should only vary minimally.

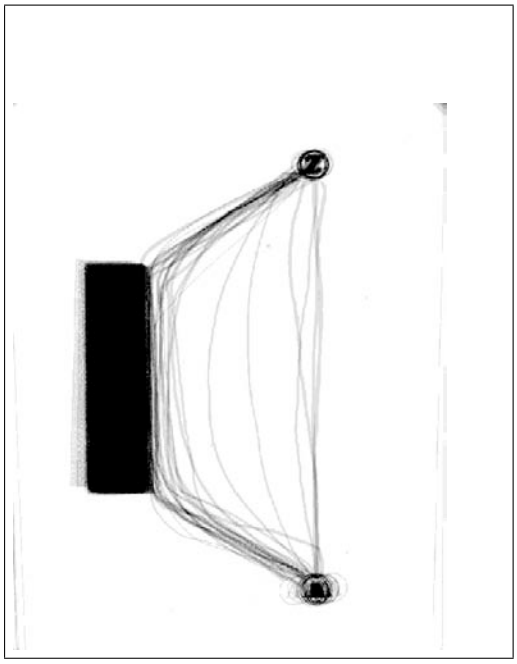
This is what we did indeed observe. On average, the entrance point of the trajectory drawn for “along” into NA (18 mm) lie closer to the anchor object than the one for “past” (49 mm) with $t(27) = 2.4, p < .05$. This pattern also applied for the exit point (17 mm vs. 49 mm) as well as for the average distance within NA (13 mm vs. 47 mm). Finally, within NA the variance was smaller in the case of “along” (5.02) than it was in the case of “past” (7.14) with $t(27) = 2.39, p < .05$. Apparently, subjects moved closer to the anchor object when they had “along” in mind than they did imagining “past”. They kept a constant distance relative to anchor object in both cases. We observed a similar pattern for the L-shaped anchor object (cf. figure A.1(c) and (d)). However, this case also illustrated that the parallel course of the trajectory that we recorded for “past” in the previous example was accidental: While the subjects still moved closer to the anchor object in the case of “along” (12 mm) than in the case of “past” (35 mm) – with $t(26) = 9.27, p < .001$ – they followed the shape of the anchor object only in the case of “along”.[1] The results from this experiment suggest that parallelism and proximity are important concepts for the discrimination of the two German path prepositions “entlang” (along) and “vorbei” (past). To verify this hypothesis we designed a second experiment, where subjects had to produce one of the two prepositions, and where we systematically varied the shape of the anchor object, and the shape/curvature of the trajectory between the start and the end point. Table A.2 explains the setup of the second experiment.

In order to more closely investigate the importance of these parallelism and closeness, we computed the frequency of use of either path preposition as well as the latencies of speech production. For the subsequent analysis, we focussed on two specific item layouts: In case A, the anchor object was a rectangle, and we compared three different trajectories. Two trajectories were parallel to the outline of the anchor object. From the two parallel ones, trajectory *t1* was closer to the anchor object than trajectory *t2* . The third trajectory *t4* violated the concept of parallelism. In

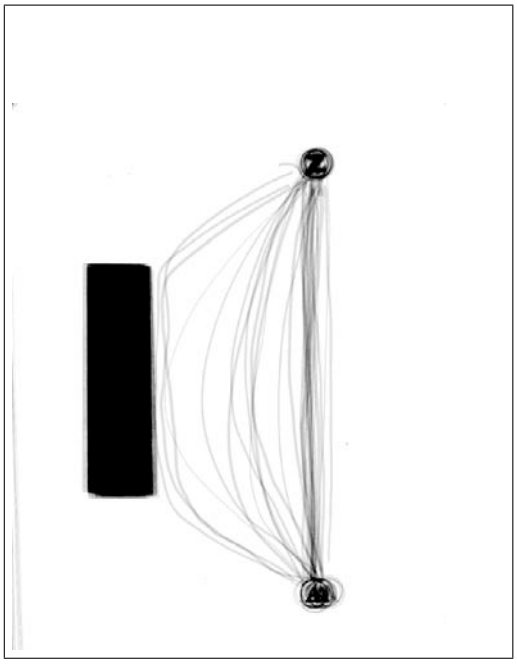
[1] A more detailed description of the results, including the other conditions that were realized in this experiment can be found in [Zimmer et al., 2001].

Subjects	The 28 participants of this experiment were students of the Saarland University. The subjects were all native speakers of German, and they were not paid for their participation.
Material	The experiment consisted of a paper and pencil test. On each sheet, there was an anchor object, a start point and an end point as well as a literal route description, e. g., ”Gehe entlang des Gebäudes” (Go along the building). The subjects were also presented with a rating scale, on which they could mark how easy (or difficult) they found the task of drawing the trajectory between start and end point. (We will not report on these rating here.) Each item was printed on a DIN A4-sized sheet of paper. At the top of the page, there was the written description. Below this, there was a framed box (16 cm by 20 cm) and the rating scale at the bottom of the sheet. Within the framed box, the anchor object was displayed as well as the start point and the end point of the trajectory the subjects had to draw. We will report the results for two different anchor objects: The first one was a rectangle (2 cm by 8 cm), the second one consisted of two rectangles (2 cm by 8 cm and 3 cm by 2 cm), which were arranged to form an “L-shaped object standing on its head” (cf. figure A.1). In the case of the plain rectangle, the start point was located 6 cm to the right and 4 cm below the lower right corner of the anchor object. The end point was 6 cm to the right and 3 cm above the upper right corner of the anchor object. In case of the L-shaped object the start point was 2.5 cm from the right and 4 cm below the lower right corner, the end point 3 cm the the right and 3 cm above the upper right corner. Along with each of these two items we gave one of the following two instructions: “Go along the building” or “Go past the building”. The resulting four items of interest were tested in conjunction with other items, which differed in the shape of the anchor object and in the accompanying instructions, e. g., “Go along the river” or “Go around the tower”. Altogether, there were 36 different items, each on separate sheet of paper. These 36 sheets were randomly shuffled and combined with a general instruction for the experiment
Procedure	The subjects were tested in two groups at the beginning of two lectures on computer science. Every subject received the 36 different items and the instructions for the experiment. They were told to read the instructions carefully and to wait for the start signal. After the signal was given, they had to draw what they thought is the best matching trajectory between the start and the end point for each of the given combinations of the anchor objects and descriptions. Additionally, they had to judge the difficulty of the task using the rating scale.

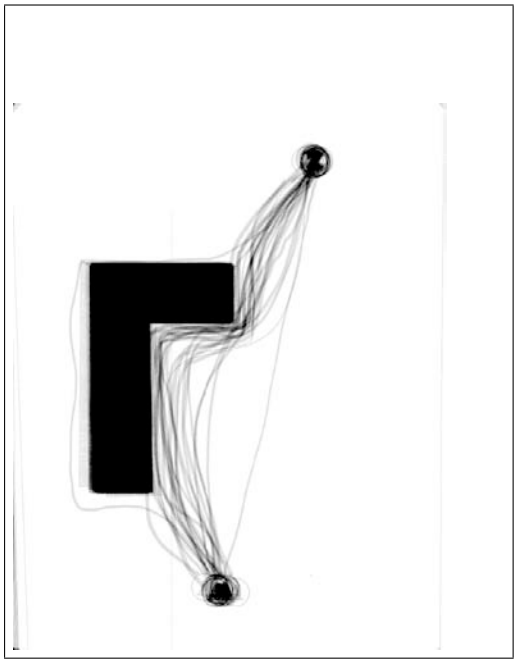
Table A.1: Description of experiment one



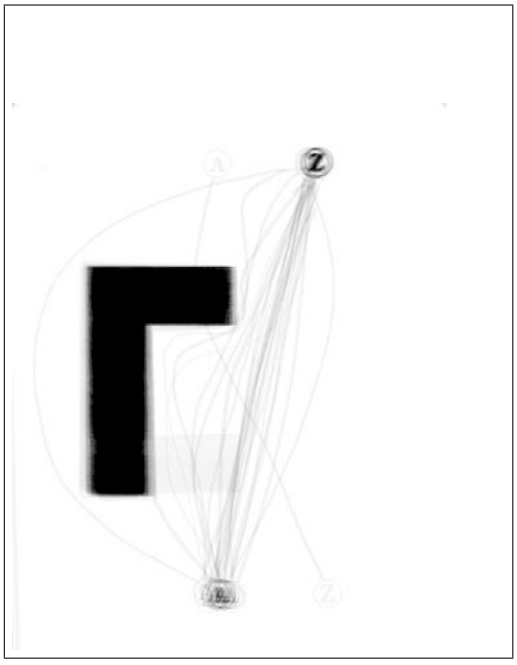
(a) “Go along the building”



(b) “Go past the building”



(c) “Go along the building”



(d) “Go past the building”

Figure A.1: The four cases of interest: Each picture shows a superimposition of the trajectories produced by all participants. Actual items were 16 cm by 20 cm. (from [Kray et al., 2001])

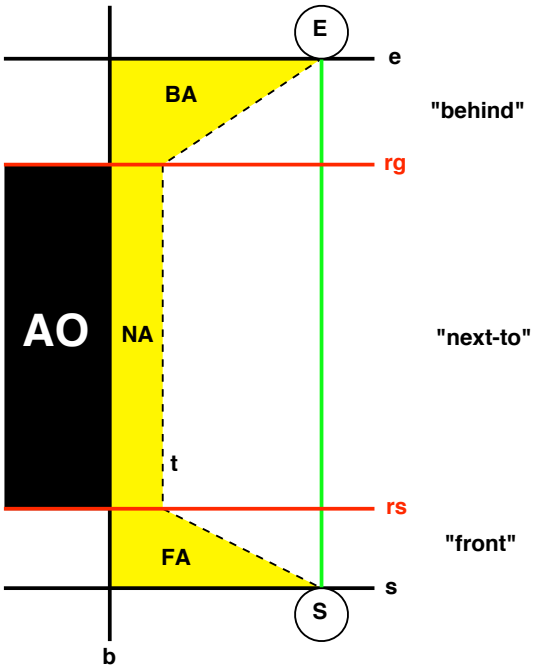
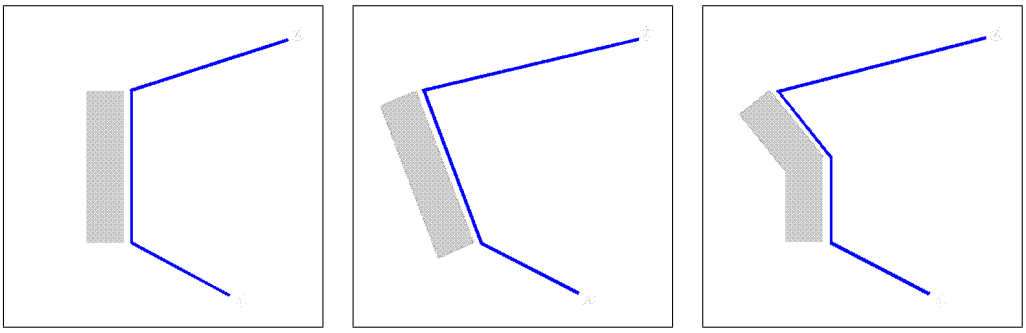


Figure A.2: Schematic description of the regions used for the analysis of path relations (adapted from [Kray et al., 2001])

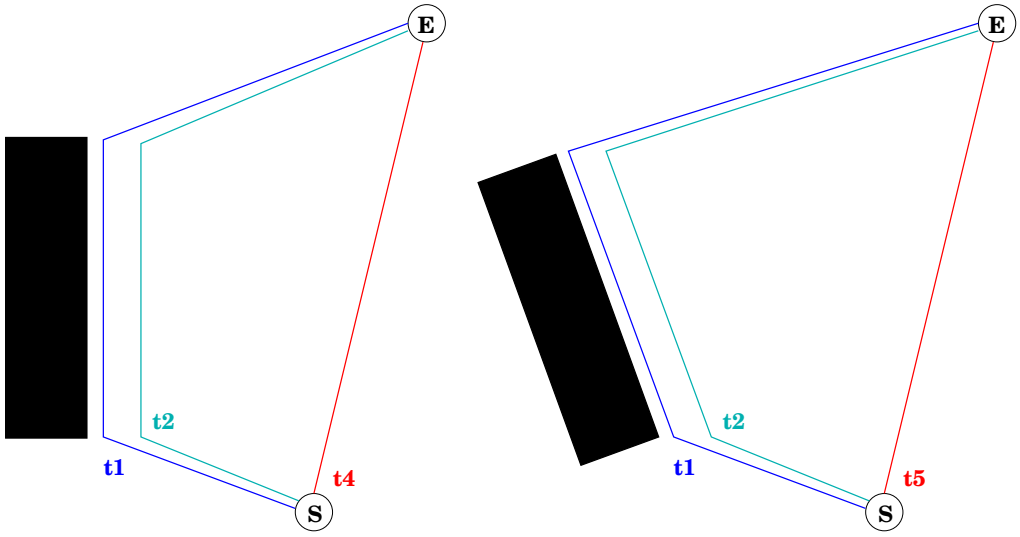


(a) (b) (c)

Figure A.3: A overview of the items used in the second experiment (from [Kray et al., 2001])

Subjects	Sixteen students of the Saarland University took part in the experiment. All subjects were native German speakers and were paid for their participation.
Material	Each item consisted of an anchor object, a start point, an end point, and a trajectory connecting those two points. We designed three different reference objects: a simple rectangle (2 x 8 cm), a rectangle of the same size, but tilted 20 degrees to the left, and another rectangle, which was bent in the middle to form a 160 degree angle. The start points for the trajectories were located 3 cm in front of (below) the anchor object's lower right corner and 2 cm respectively 6 cm to right of it (cf. figure A.3). The corresponding end points were always located 3 cm behind (above) and 9 cm to the right of the anchor object's upper right corner. Trajectories were drawn as lines of 1.5 mm width. For all items there was a mirrored counterpart with start and end points on the left side of the anchor object. Thus, 24 different items were used. They differed in the kind of anchor objects, in the location of the trajectory start point, and in the trajectory's shape/curvature. The goal of systematically varying these variables was to reveal the importance of the aforementioned concepts of parallelism and closeness in the discrimination of the two path prepositions. The items were displayed on a 17 inch computer screen, with subjects seated one meter in front of it. The experiments were controlled by an IBM compatible PC running a Java 3D application, that was specifically built for the trials.
Procedure	Subjects were seated in front of the computer screen. Each trial started with a short warning signal (a beep). One second later, the subjects saw one of the items. They had been instructed to describe aloud and as fast as possible the curvature of the trajectory in relation to the anchor object using one of the two German path prepositions "entlang" (along) and "vorbei" (past). No other descriptions were permitted. The subjects' speech production triggered a voice key, which in turn caused the item to disappear from the screen, and the subjects' choice to be recorded. After a pause of one second, the next trial began. Times were measured between the presentation of an item and the moment the subject started to reply.

Table A.2: Description of experiment two



(a) Case A: Anchor object and the trajectories *t1*, *t2*, and *t4*

(b) Case B: Anchor object and the trajectories *t1*, *t2*, and *t5*

Figure A.4: Relevant items in the preposition production condition (from [Kray et al., 2001])

case B, the anchor object was the tilted rectangle. Again, we had two parallel trajectories *t1* and *t2*, where the first one was closer to the anchor object than the second one. The third trajectory *t5* again violated the concept of parallelism. Figure A.4 shows the corresponding anchor objects and trajectories. (During the experiment, additional trajectories were used, which are not shown in the picture.)

The average frequencies of selecting "along" are reported in table A.3. A 2×3 analysis of variance of this data with the factors 'type of item' (Case A or B) and 'course of trajectory' (parallel and close, parallel and distant, and non-parallel) yielded a significant effect for the trajectory's course with $F(2, 30) = 47.16, p < .001$. Post hoc comparisons showed that the frequencies were the same for the two parallel cases, and that they were much higher than in the nonparallel case. The production latencies (cf. table A.4) were compared using the same two factors in a 2×2 analysis; we excluded the latencies for 'along' with a nonparallel trajectory due to the insufficient amount of data for this case. This analysis also yielded a significant effect for the course of the trajectory with $F(1, 13) = 8.42, p < .05$, which demonstrates that subjects produced "along" faster when describing the closer trajectory than in the case of the more distant one. From these results we can conclude that parallelism to the shape of anchor object is necessary precondition for the use of "along". We can also infer that closeness has only a weak effect on selection, but the production latencies are slightly shorter for trajectories closer to the anchor object.

A further set of items consisted of a combination of an anchor object and several trajectories that showed a different degree of deviation from parallelism with respect to the shape of the anchor object. Figure A.5 shows the corresponding items of interest: the tilted rectangle served as the

	Parallel trajectories		Non-parallel trajectory
	Distance		
	Close	Far	
Case A	87.5	77.3	10.9
Case B	89.9	87.5	3.9

Table A.3: Percentages of subjects producing “along”

	Distance	
	Close	Far
Case A	744	790
Case B	709	748

Table A.4: Latencies of subjects producing “along” for parallel trajectories (in ms)

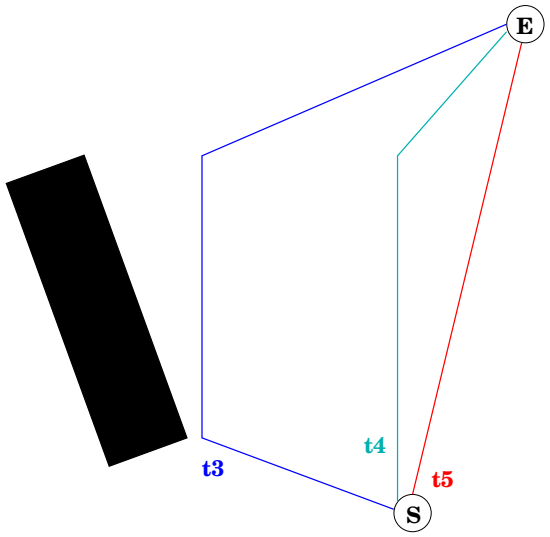


Figure A.5: Case B: Anchor object and trajectories *t3, t4* and *t5* (from [Kray et al., 2001])

	Non-parallel		Departing
	Distance		
	Close	Far	
Case B, frequencies (in %)	66.4	82.0	95.3
Case B, latencies (in ms)	896	848	824

Table A.5: Percentages and latencies of subjects producing “past” for items in figure A.5

	Start point far		Start point close
	Distance		
	Close	Far	Trajectory far
Parallel then straight	61.7	47.7	47.7
Parallel then departing	52.3	43.0	40.6

Table A.6: Percentages of subjects producing “along” for items in figure A.7

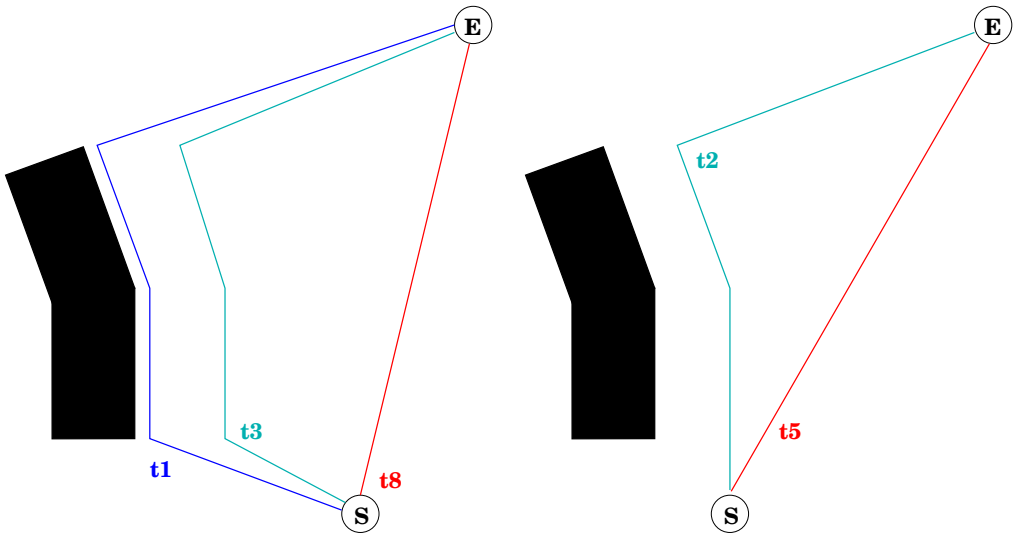
anchor object, and there were three trajectories *t3, t4, t5*, which deviated from being parallel with the shape of the anchor object. While trajectories *t3, t4* were at least partially parallel to each other – but not to the anchor object – *t3* is located closer to the anchor object than *t4*. Trajectory *t5* leads away from the anchor object, and is neither parallel to any other trajectory nor to the anchor object. The resulting frequencies and speech production latencies for the selection of “past” are listed in table A.5.

In case of the non-parallel trajectories, “past” was more frequently used than “along” (see table A.5). An analysis of the frequencies and the production latencies for “past”[2] in a one-way analysis with three levels – (1) close and passing, (2) distant and passing, and (3) departing – again revealed a significant effect for the course of the trajectory: $F(2, 30) = 6.66, p < .01$ for frequencies, and $F(2, 26) = 5.74, p < .01$ for latencies. “Along” was used less frequently and more slowly in the case of the passing trajectory that was close to the anchor object, than in case of the other two trajectories.

In order to test whether we can replicate these effects with a differently shaped anchor object, we first investigated five further trajectories. Three of them were parallel, of which one was close to the anchor object (*t1*). The two others (*t2, t3*) were located farther away (at the same distance) but had distinct start points, which were located in different distances. They shared these start points with the two non-parallel trajectories (*t5, t8*) Figure A.6 shows the resulting items. Again, we observed a replication of our results: The parallel trajectories were described using “along” (89, 90 and 88 %, respectively), while the non-parallel ones were not (both 6 %) with $F(4, 60) = 75.80, p < .001$.

In a next step, we then investigated trajectories that were only partially parallel to the anchor object. We manipulated the distances of the trajectories as well as the course in their non-parallel part: In the region formerly denoted as NA, only one half of the trajectory was parallel to the anchor object, while the other one was not. The latter part was either passing or departing (cf.

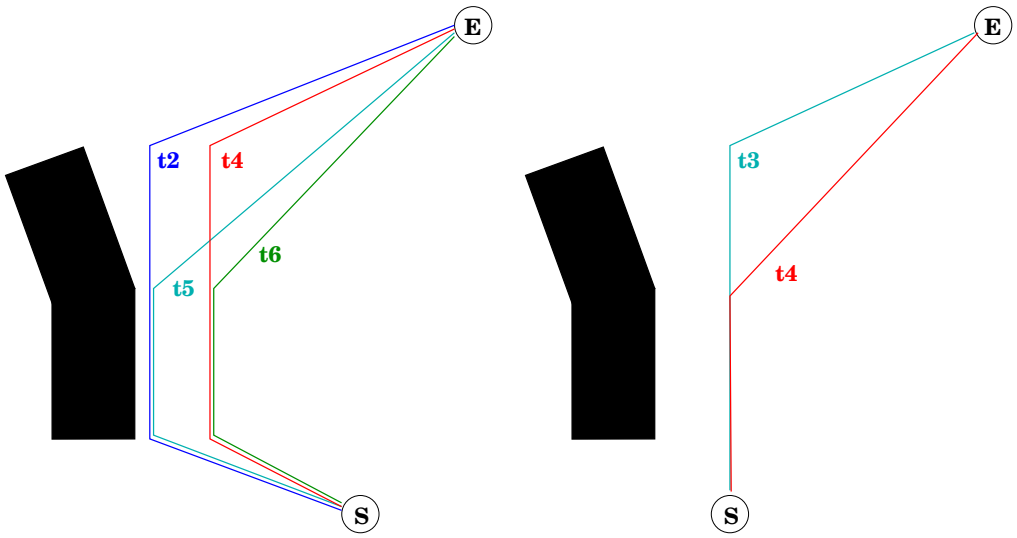
[2] those for “along” were $1 - f(past)$



(a) Case C: anchor object and trajectories *t1*, *t3*, and *t8*

(b) Case D: anchor object and trajectories *t2*, and *t5*

Figure A.6: Parallel vs. parallel/departing trajectories (from [Kray et al., 2001])



(a) Case C: anchor object and trajectories *t2*, *t4*, *t5* and *t6*

(b) Case D: anchor object and trajectories *t3*, and *t4*

Figure A.7: Partially parallel trajectories (from [Kray et al., 2001])

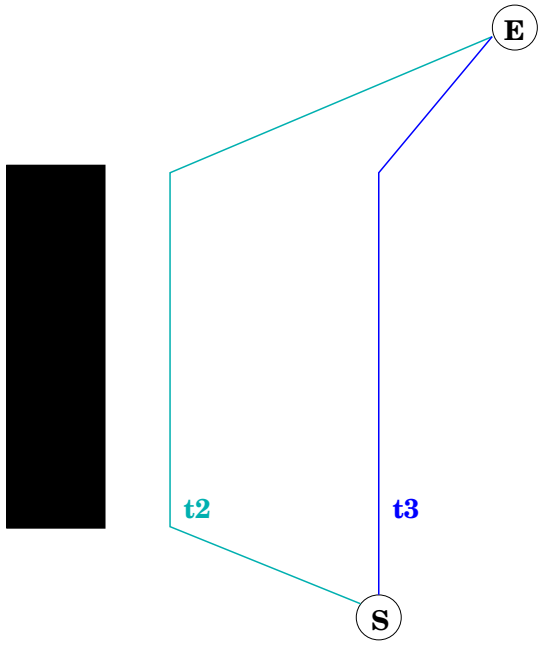


Figure A.8: Illustration of the distance threshold (from [Kray et al., 2001])

figure A.7). The production frequencies of "along" were recorded and are listed in table A.6 depending on the courses of the trajectories. We first compared these frequencies in a 2×3 analysis of variance with the factors 'degree of parallelism' (2) and 'distance' (3). In this analysis, only the degree of parallelism was significant with $F(2, 30) = 4.49$, $p < .05$. "Along" was produced more often to describe the proximal trajectories (57%) than to describe one of the two more distant trajectories (46% and 45%). No other effect was significant. We were unable to analyze the latencies: the number of remaining data was too small due to the case-wise deletion of subjects in the repeated measurement analysis.

We then combined these partially parallel conditions and compared them with the other conditions, where trajectories were either completely parallel or non-parallel. We analyzed the data in a one-way analysis with the following levels: (1) completely parallel, (2) partially parallel and passing, (3) partially parallel and departing, and (4) nonparallel. The corresponding frequencies were 89%, 52%, 54%, and 6%; their difference was highly significant with $F(3, 45) = 26.91$, $p < .001$. The parallel trajectories were more often described using "along" than any other trajectories. The partially parallel ones did not differ in terms of the frequency, which was still higher than for the non-parallel trajectories. In the latter case, the subjects almost exclusively used "past".

We can draw two main conclusions from the series of experiments that we presented in the previous paragraphs. Firstly, there is strong evidence that a necessary precondition for the applicability of "along" consists of the parallelism between the corresponding trajectory and the adjacent part of the outline of the anchor object. In the path production experiment, the subjects were willing to take great detours in order to assure that the trajectory they drew was at least partially parallel to the anchor object. This thesis was supported by the results of the subsequent trials,

where subjects were asked to describe trajectories using one of the two path prepositions. Secondly, the distance between the anchor object and the trajectory influenced the selection in several ways. While closeness did yield faster response times in case of parallel trajectories, there were also trials where proximity induced a higher percentage of subjects choosing "along" in case of partially parallel trajectories (cf. [Zimmer et al., 2001]). This implies that closeness is a secondary criterion that is called upon in cases where the degree of parallelism is not high enough to justify the selection of "along". Furthermore, the consistent effort that most subjects in the path production condition put into assuring the closeness between trajectory and anchor object, suggests that the proximity between these two is a key factor.

The comparison of two specific items from the path production experiment also indicates that parallelism is not sufficient in order to select "along". Figure A.1 shows these two items : the subjects were given the description "Go along the building" in case (a) and "Go past the building" in case (b). Obviously, superimposed trajectories are parallel to the adjacent outline, even though the trajectories in case (b) were produced to depict "past". A plausible reason for the parallelism in case (b) lies in the specific constellation: the most direct route from source to target is a straight line that happens to be parallel to the anchor object. This *accidental parallelism* may raise some problems for the computational modeling of "along" as its applicability seems to depend also on alternate routes, i. e. the structure of the surrounding environment and the corresponding set of possible routes.[3]

A second important observation concerns the degree of closeness: while parallel trajectories closer to the anchor object yielded faster response times (at a similar selection rate) than trajectories that were farther away, there seems to be a threshold distance. Once a trajectory is farther away than that, "along" is almost never chosen. In figure A.8 both trajectories are equally parallel to the reference object, yet "along" is selected by 77% for *t2*, but by only 43% for *t3*. The determination of the threshold value and relevant factors that influence it are subject of further research. As a first approximation, we included the two-point relation `near` into the computation of basic path relations in our model. However, due to the observation we reported in the previous paragraphs, it is clear that there is no exact mapping from the path relation `follow` to the preposition "along" (although such a mapping may serve as a starting point to identify further aspects of the modeling.)

Throughout the different trials, "past" seemed to be the less specific case. "Past" was only consistently chosen, when the trajectory led straight from source to target (ignoring the shape of the anchor object), or when it led away from the reference object. Otherwise, there was no clear trend as to when "past" was preferred over "along". These observations can be interpreted in several ways. On the one hand, "past" may have a less specific meaning, whereas "along" is defined more crisply. Therefore, "past" is only chosen when the more crisp case does not apply. However, there is also evidence against this interpretation as production latencies were similar in cases, where both prepositions were applicable (cf. [Zimmer et al., 2001]). On the other hand, there may be inconsistent perceptions on the interpersonal level of what "past" means. In figure A.1(b), most subjects drew a straight line from source to target when asked to illustrate "past". However, quite

[3]For example, when driving a car "past" a building, there may be only one route, which happens to be parallel to the building. Since the car cannot drive, e. g., over stairs or through other buildings, the street is the only object that affords driving on it in this environment.

a few drew lines closer to the building, which was the most frequent behavior in the "along" condition. Finally, "past" may be the *default* preposition subjects used, when they just wanted to establish a relation between the trajectory and the anchor object, and were unable or unwilling to specify it in more detail.

From a different perspective, one may argue that the applicability of "along" depends on the *intention* of the producer. By using it instead of the less specific case of "past", a pragmatic goal is achieved such as making sure that the listener gets to see a certain sight, or does not get lost.[4]. This argument can also explain the effects of distance that we observed: once a threshold distance is passed, the intention behind the use of "along" can no longer be fulfilled. This may also be the reason why subjects went to great detours in order to approach the anchor object when drawing trajectories for "along" (cf. figure A.1(a)).

In our experiment, we relied mainly on simple trajectories that were rather short and incorporated few turns. However, when a trajectory gets longer and more complex it may be hard or impossible to find a single path relation (or preposition) to describe it well. This is one issue, which the process described in the section 4.1.5 can address.

[4]This may be the case, for example, when the area is crowded and the object to walk along allows the listener to constantly reassure that she is on the right way.

B Encoding of interactions

In section 4.2, we presented a language independent format for representing interactions on spatial topics – the preverbal message (PVM). The corresponding data structure in the implementation was shortly introduced in section 6.4.2. In this appendix, we provide a detailed description of how all interactions between the user and SISTO on Deep Map were encoded using the preverbal message.

Question type	Slot	Value(s)	Comment
all	`performative`	`request`	Asking a question is equivalent to requesting an answer.
What is this ⟨*A*⟩ ?	`types`	`identification`	The goal is to identify ⟨*A*⟩.
	`globalGoal`	⟨*A*⟩	E. g., a name or an anaphoric expression
Tell me more about ⟨*A*⟩!	`types`	`description`	The goal is to get further information on ⟨*A*⟩.
	`globalGoal`	⟨*A*⟩	E. g., a name or an anaphoric expression
Where is ⟨*A*⟩?	`types`	`localization`	The goal is to learn the location of ⟨*A*⟩.
	`globalGoal`	⟨*A*⟩	E. g., a name or an anaphoric expression
Guide me [from ⟨*A*⟩] to ⟨*B*⟩!	`types`	`path`	The goal is to get incremental guidance [from ⟨*A*⟩] to ⟨*B*⟩.
	`globalGoal`	⟨*B*⟩	E. g., a name or an anaphoric expression
	`pvmoStart`	⟨*A*⟩	E. g., a name or an anaphoric expression
How do I get [from ⟨*A*⟩] to ⟨*B*⟩?	`types`	`completePath`	The goal is to get a description of the entire path [from ⟨*A*⟩] to ⟨*B*⟩.
	`globalGoal`	⟨*B*⟩	E. g., a name or an anaphoric expression
	`pvmoStart`	⟨*A*⟩	E. g., a name or an anaphoric expression

Table B.1: Slot allocation in PVM data structure for questions

Table B.1 shows a general overview on how the requests of the user are encoded. Obviously, the corresponding performative is `request` in theses cases. Currently, SISTO on Deep Map allows for five distinct questions or request that the user can ask the system:

- **identification**
  The user wants to learn the name of an object and asks the system "What's this?" – possibly including an object type such as in "What is this church?". The corresponding preverbal message type is `identification` and further information on the target object is passed in the slot `globalGoal`.

- **description**
  While this use case is similar to the identification, the user's main intention here is not to learn the name of an object but to get further information on it, i. e. by asking "Tell me more about ⟨*A*⟩.". The PVM type hence is `description`, and ⟨*A*⟩ is stored in `globalGoal`.

- **localization**
  If the user asks the system "Where is ⟨*A*⟩?", she most likely want to know its location. Again, the target object ⟨*A*⟩ is stored in `globalGoal` but the PVM type is `localization`.

- **incremental guidance**
  The key function of a navigational assistant consists of guiding its user to her destination. "Guide me to ⟨*B*⟩." is a question that will trigger this service. The corresponding PVM type is `path`, and the target is stored in `globalGoal`.[1]

The corresponding replies to the identification and description request are shown in table B.2. They consist of the same entries, except for the performative, which is `inform`, and `globalGoal`, which contains a complete reference to the target object. This information allows for retrieving further information on the target object from the database, e. g. images, or historical data.

Reply	**Slot**	**Value(s)**	**Comment**
all	`performative`	`inform`	The answer to a question is equivalent to providing the requested information.
This is ⟨*A*⟩	`types`	`identification`	PVM type corresponds to original query.
	`globalGoal`	⟨*A*⟩	contains name, id, and further information on ⟨*A*⟩
Here is more information on ⟨*A*⟩: ...	`types`	`description`	PVM type corresponds to original query.
	`globalGoal`	⟨*A*⟩	contains name, id, and further information on ⟨*A*⟩, which can be used to access further information in the database, GIS, etc.

Table B.2: Slot allocation in PVM data structure for identification and description

[1] In some cases it may make sense to include an origin ⟨*A*⟩, e. g. when the user is on a virtual journey. Within the implementation presented in chapter 6, such a case does not occur.

The reply to a localization request is more complicated (see table B.3). While the performative is again `inform`, the preverbal message type corresponds to the query (`localization`) and the complete reference to the target object is stored in `globalGoal`, we also have to encode the relational localization: The spatial (two-point) relation is stored in `stprTwoPointRelation` and the corresponding anchor object in `pvmoTwoPointRelation`. We also assign a degree of applicability to the complete relational proposition, which is passed in `daTwoPointRelation`. In addition, the metric distance from anchor to target object as well as a turn angle are stored in `metric` and `angle`. The latter is used to encode induced frames of reference (non-zero values), or an ego-centric frame of reference (angle equals zero).

Reply	**Slot**	**Value(s)**	**Comment**
$\langle A \rangle$ is to the $\langle rel \rangle$ of $\langle B \rangle$.	`performative`	`inform`	The system provides the user with information.
	`types`	`localization`	PVM type corresponds to original query.
	`globalGoal`	$\langle A \rangle$	contains name, id, and further information on $\langle A \rangle$
	`stprTwoPointRelation`	$\langle rel \rangle$	spatial two-point relation $\langle rel \rangle$
	`pvmoTwoPointRelation`	$\langle B \rangle$	the anchor object $\langle B \rangle$ of relation $\langle rel \rangle$
	`daTwoPointRelation`	$da(\langle A \rangle, \langle rel \rangle, \langle B \rangle)$	the degree of applicability for the relational proposition $(\langle A \rangle, \langle rel \rangle, \langle B \rangle)$
	`metric`	$dist(\langle A \rangle, \langle B \rangle)$	distance from $\langle A \rangle$ to $\langle B \rangle$
	`angle`	$\langle turn \rangle$	turning angle (in case of a induced frame of reference)

Table B.3: Slot allocation in PVM data structure for a localization

In table B.4, the encoding scheme for directions is shown, which is employed for incremental as well as for complete guidance. It contains all information to provide the user with instructions (such as the one shown on the left side of the table) for a single route segment. Performative, PVM types, and the global goal are filled in analogy to the previous three use cases. However, in order to generate rich and detailed route instructions, the PVM contains three two-point relations and one path relation, which each consist of an anchor object, a relation, and a degree of applicability. The corresponding target objects are the start and the end point of the route as well as the entire route in case of the two-point relations. The entire route is also the target object of the path relation. The frames of reference are deduced from the route perspective. Finally, the length of the segment is stored in `metric`, and the turn that is required at the beginning of the segment in order to align oneself with the segment, is encoded in `angle`.

Reply	**Slot**	**Value(s)**	**Comment**
In order to get to $\langle G \rangle$, you have to turn $\langle dir \rangle$. Then, $\langle A \rangle$ is on your $\langle rel_A \rangle$ and walk $\langle rel_B \rangle$ $\langle B \rangle$. $\langle C \rangle$ will be on your $\langle rel_C \rangle$. Keep walking until $\langle D \rangle$ in on your $\langle rel_D \rangle$.	`performative`	`inform`	The user is informed about the next segment $S(S_1...S_n)$ of the path she requested.
	`types`	`path`	PVM type corresponds to original query.
	`globalGoal`	$\langle G \rangle$	contains name, id, and further information on $\langle G \rangle$
	`stprStart`	$\langle rel_A \rangle$	two-point relation $\langle rel_A \rangle$
	`pvmoStart`	$\langle A \rangle$	the anchor object $\langle A \rangle$ of relation $\langle rel_A \rangle$
	`daStart`	$da(\langle S_1 \rangle, \langle rel_A \rangle, \langle A \rangle)$	the degree of applicability for the relational proposition $(\langle S_1 \rangle, \langle rel_A \rangle, \langle A \rangle)$
	`snpr`	$\langle rel_B \rangle$	n-point relation $\langle rel_B \rangle$
	`pvmoNPointRelation`	$\langle B \rangle$	the anchor object $\langle B \rangle$ of relation $\langle rel_B \rangle$
	`daNPointRelation`	$da(\langle S \rangle, \langle rel_B \rangle, \langle B \rangle)$	the degree of applicability for the relational proposition $(\langle S \rangle, \langle rel_B \rangle, \langle C \rangle)$
	`stprTwoPointRelation`	$\langle rel \rangle$	two-point relation $\langle rel \rangle$
	`pvmoTwoPointRelation`	$\langle C \rangle$	the anchor object $\langle C \rangle$ of relation $\langle rel_C \rangle$
	`daTwoPointRelation`	$da(\langle S \rangle, \langle rel_C \rangle, \langle C \rangle)$	the degree of applicability for the relational proposition $(\langle S \rangle, \langle rel_C \rangle, \langle C \rangle)$
	`stprEnd`	$\langle rel_D \rangle$	two-point relation $\langle rel_D \rangle$
	`pvmoEnd`	$\langle D \rangle$	the anchor object $\langle D \rangle$ of relation $\langle rel_D \rangle$
	`daEnd`	$da(\langle S_n \rangle, \langle rel_D \rangle, \langle D \rangle)$	the degree of applicability for the relational proposition $(\langle S_n \rangle, \langle rel_D \rangle, \langle D \rangle)$
	`metric`	$dist(\langle S_1 \rangle, \langle S_n \rangle)$	length of segment
	`angle`	$\langle dir \rangle$	turning angle (reorientation)

Table B.4: Slot allocation in PVM data structure for (incremental) guidance

In section 5.3, we introduced an interactive mechanism to determine the user's current positions. Table B.5 lists the questions and requests that the system asks the user in this context. The most simple one is the request accompanying a map that the user can click on to communicate his current position to the system ("Please locate yourself."). While this means was not realized in the current implementation, the other three questions listed in the table were. The first one is a request to confirm the current location ("Are at $\langle A \rangle$?"), which is encoded using the corresponding

performative (confirm) and a localization consisting of the target object ⟨A⟩ in globalGoal and the two-point relation stprNextTo. If there are several alternatives ("Are you at ⟨A_1⟩, ⟨A_1⟩, ... or ⟨A_n⟩?"), the target objects are stored in choices and the PVM type choosing is added.

The key question used to interactive determine the user's current position – "Can you see ⟨A⟩?" – corresponds to a preverbal message with a performative confirm and a PVM type identification. The target object is again stored in globalGoal. In case there are several alternatives ("Can you see ⟨A_1⟩, ⟨A_1⟩, ... or ⟨A_n⟩?"), the target objects are passed in choices, and the choosing PVM type is added.

Utterance type	**Slot**	**Value(s)**	**Comment**
Please locate yourself.	performative	request	Asking a question is equivalent to requesting an answer.
	types	localization	The user is asked to specify her own location.
Are you at ⟨A⟩?	performative	confirm	The user is asked to confirm that she is near ⟨A⟩.
	types	localization	
	globalGoal	⟨A⟩	
	twoPointRelation	stprNextTo	
Are you at ⟨A_1⟩, ⟨A_1⟩, ... or ⟨A_n⟩?	performative	confirm	The user is asked to confirm that she is near ⟨A_i⟩.
	types	localization, choosing	The user is asked to choose among several localizations.
	twoPointRelation	stprNextTo	
	choices	{⟨A_1⟩..⟨A_n⟩}	List of choices
Can you see ⟨A⟩?	performative	confirm	The user is asked to confirm that ⟨A⟩ is visible from her current location.
	types	identification	
	globalGoal	⟨A⟩	
Can you see ⟨A_1⟩, ⟨A_1⟩, ... or ⟨A_n⟩	performative	confirm	The user is asked to confirm that ⟨A_i⟩ is visible from her current location.
	types	identification, choosing	
	choices	{⟨A_1⟩..⟨A_n⟩}	

Table B.5: Slot allocation in PVM data structure for questions in the context of interactive positioning

Utterance type	**Slot**	**Value(s)**	**Comment**
Yes.	`performative`	`inform`	Confirmation.
	`types`	`agreement`	
No.	`performative`	`inform`	Disconfirmation.
	`types`	`disagreement`	
I do not know.	`performative`	`inform`	Reply if neither confirmation nor disconfirmation is possible.
	`types`	`indetermination`	
I am at ⟨*A*⟩.	`performative`	`inform`	The user states that she is located near ⟨*A*⟩.
	`types`	`localization`	
	`globalGoal`	⟨*A*⟩	
	`twoPointRelation`	`stprNextTo`	
I can see ⟨*A*⟩.	`performative`	`inform`	The user states that she can see ⟨*A*⟩.
	`types`	`identification`	
	`globalGoal`	⟨*A*⟩	

Table B.6: Slot allocation in PVM data structure for utterances for positioning

In table B.6, the preverbal messages corresponding to the user's replies in the context of interactive positioning are listed. The three basic replies ("Yes.", "No." and "I do not know.") are encoded using the performative `inform` and the corresponding preverbal message types `agreement`, `disagreement` and `indetermination`. The user can also state directly that she is located near an object, which is translated into a preverbal message, which resembles the one shown in table B.3 for the system's output in case of a localization – except for metrical data, turn angle, and the degree of applicability. The latter ones are not used for encoding the user's utterance.

An even greater similarity exists between the PVM used to encode "I can see ⟨*A*⟩." and the one that encodes the reply to an identification request (shown in table B.2). If only one target object is given, both PVMs are equal. In case the user can see several objects, these are passed as `choices` in analogy to the corresponding query listed in table B.5.

A further service provided by SISTO is the support for mobile data collection. The user can enter or leave the data collection modus by saying "Start data collection." and "Stop data collection." (see table B.7). The dialog to collect data is highly structured: the system first asks the user to identify the target object, and then, whether she wants to subdivide it into parts. The latter question allows the user to collect data on large unstructured objects such as city blocks. If the user does not unambiguously specify the target object, the system can ask her to select it from a list of potential targets. Similarly, if the user specified more than one part, she is asked to select one from a list. While the partitioning is purely qualitative in nature (i. e. the user cannot spatially subdivide an object), the user can record unstructured annotations for each part while she is collecting data. Later on – when she is adding the collected data to the database – these annotations can be used to model the parts accordingly, e. g. using the underlying GIS.

Utterance type	Slot	Value(s)	Comment
Start data collection.	performative	request	The user initiates data collection.
	types	dataCollection	
Stop data collection.	performative	request	The user stops data collection.
	types	dataCollection, disagreement	
Please specify target object.	performative	request	The user is asked to specify the object about which she wants to collect data.
	types	identification	
Do you want to edit $\langle A_1 \rangle$, $\langle A_2 \rangle$, ... or $\langle A_n \rangle$, or a part of them?	performative	request	The user is asked to select the object that she wants to edit.
	types	choosing, editPartial	The user is asked to choose among several objects or parts of them.
	choices	$\{\langle A_1 \rangle..\langle A_n \rangle\}$	List of choices
How many subdivision of $\langle A \rangle$ shall be created?	performative	request	The user is asked to specify the number of subdivision of $\langle A \rangle$.
	types	queryValue, editPartial	
	globalGoal	$\langle A \rangle$	Target object
Do you want to edit $\langle P_1 \rangle$, $\langle P_2 \rangle$, ... or $\langle P_n \rangle$?	performative	request	The user is asked to select the object part that she wants to edit.
	types	choosing	The user is asked to choose among several objects parts.
	choices	$\{\langle P_1 \rangle..\langle P_n \rangle\}$	List of choices

Table B.7: Slot allocation in PVM data structure for data acquisition (specification of target object)

Utterance type	**Slot**	**Value(s)**	**Comment**
Now specifying attributes for object ⟨*A*⟩.	`performative`	`inform`	The user is informed that data is now being collected about ⟨*A*⟩.
	`types`	`queryValue`	
	`globalGoal`	⟨*A*⟩	
Please specify attribute.	`performative`	`request`	The user is asked to specify the attribute that she wishes to fill.
	`types`	`queryObject`	
	`attributes`	{*undefined*}	
Please specify value for attribute ⟨*A*⟩.	`performative`	`request`	The user is asked to specify the value of attribute ⟨*A*⟩.
	`types`	`queryValue`	
	`attributes`	{⟨*A*⟩}	

Table B.8: Slot allocation in PVM data structure for data acquisition (data entry)

The PVMs used to encode the interactions mainly rely on two further preverbal message types: `dataCollection` is used to signal the beginning and end of data collection, `editPartial` marks those PVMs that are related to partitioning the target object, and `queryValue` indicates that the corresponding PVM encodes a question for a value (such as the number of parts). In order to distinguish queries for attributes from those asking for values, there is an additional preverbal message type `queryObject`, that is used to encode questions such as the one asking for the next attribute that the user want to specify. Table B.7 and B.8 give a complete list of the PVMs that encode the interactions specific to the data collection task.

The user's replies to the questions listed in those tables are either localizations (e. g. "(I want to collect data on) the object in front of me."), identifications such as "(I want to collect data on) the Peterskirche."), or the input of attributes and values. The first two are encoded in the same way as shown previously, the latter ones correspond to the last two PVMs shown in table B.8 with the exception of the performative, which is of course `inform`, and `attributes`, which contain the attribute or value that the user entered.

Symbols

A

B

C

D

E

F

G

H

I

L

M