D2.3.2: Final Pedestrian Behaviour Component

Alexandre Albore, Johan Boye, Morgan Fredriksson,
Jana Götze, Joakim Gustafson, Jürgen Königsmann

Distribution: Public

SpaceBook
Spatial & Personal Adaptive Communication Environment: Behaviors & Objects & Operations & Knowledge
270019 Deliverable 2.3.2

July 1, 2013

Project funded by the European Community
under the Seventh Framework Programme for
Research and Technological Development

The deliverable identification sheet is to be found on the reverse of this page.

Project ref. no.	270019
Project acronym	SpaceBook
Project full title	Spatial & Personal Adaptive Communication Environment: Behaviors & Objects & Operations & Knowledge
Instrument	STREP
Thematic Priority	Cognitive Systems, Interaction, and Robotics
Start date / duration	01 March 2011 / 36 Months

Security	Public
Contractual date of delivery	M28 = June 2013
Actual date of delivery	July 1, 2013
Deliverable number	2.3.2
Deliverable title	D2.3.2: Final Pedestrian Behaviour Component
Type	Software
Status & version	1.0
Number of pages	27 (excluding front matter)
Contributing WP	2
WP/Task responsible	KTH
Other contributors	
Author(s)	Alexandre Albore, Johan Boye, Morgan Fredriksson, Jana Götze, Joakim Gustafson, Jürgen Königsmann
EC Project Officer	Franco Mastroddi
Keywords	spatially aware systems, navigation, GPS, dialogue, user modeling, cognitive modeling, AI planning

The partners in SpaceBook are:

Umeå University	UMU
University of Edinburgh HCRC	UE
Heriot-Watt University	HWU
Kungliga Tekniska Högskola	KTH
Liquid Media AB	LM
University of Cambridge	UCAM
Universitat Pompeu Fabra	UPF

For copies of reports, updates on project activities and other SPACEBOOK-related information, contact:

The SPACEBOOK Project Co-ordinator:
Dr. Michael Minock
Department of Computer Science
Umeå University
Sweden 90187
mjm@cs.umu.se
Phone +46 70 597 2585 - Fax +46 90 786 6126

Copies of reports and other material can also be accessed via the project's administration homepage, `http://www.spacebook-project.eu`

Contents

Executive summary

This deliverable describes user studies performed in order to shed more light on the cognitive modeling aspects of Spacebook, as well as describes implemented software, namely reference resolution used by simulated pedestrians, and an integrated AI planning and execution environment for cognitive modeling and dialogue management.

1 Introduction

This deliverable describes user studies performed in order to shed more light on the cognitive modeling aspects of Spacebook, as well as describes implemented software, namely reference resolution used by simulated pedestrians, and an integrated AI planning and execution environment for cognitive modeling and dialogue management.

The Spacebook system uses data from a geographic database to construct a route from the user's starting position to his stated goal, and then give the instructions as the user is moving: When the user reaches a node p_i in the planned route, the system informs the user how he should go to the next node p_{i+1}. Obviously, it is vital that each instruction is unambiguous and understandable, lest the user takes a wrong turn.

One option is to use left-right-go-straight instructions together with street names, but this strategy has its drawbacks: Left-right instructions can be ill-timed or otherwise misleading due to GPS errors and system latency, and street names do not always provide the necessary information if, for instance, the user is unacquainted with the neighbourhood.

It is therefore preferable for wayfinding systems to base their instructions on *landmarks*, by which we understand distinctive objects in the city environment. It is well-established that it is predominantly by landmarks that people describe routes to one another (see e.g. [Denis et al, 1999]) and it has been shown that the inclusion of landmarks into system-generated instructions for a pedestrian raises the user's confidence in the system, compared to only left-right instructions [Ross et al., 2004].

Thus, in order to give good navigation instructions, the system needs to have a representation of which direction the user is currently heading, the direction he *should be* going, the set of landmarks currently visible to the user, the landmarks the user currently sees (not the same thing), the landmarks that have been mentioned in previous instructions, and possibly the places and landmarks that the user has visited on previous occasions. All these things make up the system's *cognitive model* of the user.

A *simulation* of a user needs to have a representation of the same aspects. The difference is that a simulated user, to be realistic, cannot be omniscient concerning the location of all places and landmarks. A simulated pedestrian needs to maintain a representation the buildings, roads, etc. in the immediate vicinity of the current position (in order to generate movement, and to understand relative references like "left", "right", "straight ahead", etc.), and similarly a representation of landmarks in a (simulated) field of vision (to be able to understand references like "Starbucks"), but a complete knowledge of the entire city is neither necessary nor desired. The restricted geographic knowledge of the simulated user should mimic that of a real pedestrian.

On the other hand, the Spacebook system has complete geographic knowledge but an incomplete cognitive model of the user. The system can only approximate the set of landmarks visible to the user, since the user's position cannot be determined with 100% accuracy due to errors in the GPS readings. Moreover, even though a landmark is visible for the user, it is not certain that the user is aware of it. In order to successfully give instructions, the system therefore has to estimate whether it is likely that the user will understand a direct reference to the landmark ("Go towards X."), or whether it is first necessary to call the user's attention to it ("Can you see X?"). It is of course possible that the user answers the latter question in the negative, after which the system has to find another landmark on which to base the next instruction. This process is a kind of *grounding* [Traum, 1999], and will be further discussed in the next section.

The report is organised as follows: Section 2 describes a system implemented and a study preformed to investigate the spatial grounding strategy mentioned above. Section 3 describes two studies performed

to evaluate a method of personalising salience computations, to find the most appropriate landmarks to use in navigation instructions. Section 4 discusses spatial reference resolution. Section 5 describes the integrated AI planning and execution method.

2 Spatial grounding for city exploration

This section describes an implemented dialogue system for helping a user explore the city of Stockholm. The system can either guide the user to a location of his choice ("I want to go to Odengatan"), or to specific spots chosen by the system, like a statue or an interesting architectural detail on a particular building. The latter setting in particular is interesting as it allows us to investigate various methods for producing referring spatial expressions, in order to help the user find quite small objects in a complex city environment.

In general, the city exploration problem addressed here is challenging since it involves the interpretation and generation of utterances within a rapidly changing spatial context under uncertainty.

2.1 Background

Many researchers within cognitive psychology have investigated how people give route instructions to one another (see e.g. [Denis et al, 1999]), and what the elements of a good route description are (see e.g. [Lovelace et al, 1999], [Tom and Denis, 2003]). It is however not clear how these results transfer to computational models of route description generation. One finding is that a big portion of such dialogues are devoted to grounding; making sure that the dialogue partner actually sees and understands what is being referred to. Grounding is a well-studied phenomenon also in dialogue systems (see e.g. [Traum, 1999], [Skantze, 2007]).

The implemented systems for guiding pedestrians have mostly been based on spoken output from the system, with little or no possibility for the user to provide information ([Malaka and Zipf, 2000, Krug et al, 2003, Jöst et al, 2005, Bartie and Mackaness, 2006, Zipf and Jöst, 2005]). Spoken dialogue systems in spatial domains have mostly focused on non-dynamic contexts where the user can ask questions about a static map (e.g. [Cai et al, 2003, Wang et al, 2008]), on virtual environments such as computer games (e.g. [Boye and Gustafson, 2005, Boye et al, 2006, Skantze et al, 2006, Striegnitz et al, 2011]), in indoor environments ([Cuayáhuitl and Dethlefs, 2011]), or on natural-language interfaces to robots ([Lemon et al, 2001, MacMahon et al, 2006, Johansson et al, 2011]). Few if any researchers have so far addressed the topic of spoken natural-language dialogue with a user in a real, dynamic city environment.

2.2 Uncertainty and Grounding

A recurring problem for any pedestrian routing system is to describe to the user how to get from his current position to the next node in the planned route. This has to be done reliably even though the user's position, speed and direction are uncertain due to possible errors in GPS readings. Giving simple instructions like "turn left here" is therefore a risky strategy; such instructions might be nonsensical for the user if he is not quite where the system believes him to be. Furthermore, the interpretation of left and right is not always clear, for instance in parks and open squares, or when the user is standing still without the system knowing which way he is facing. Therefore, before giving directions, it is often preferable that

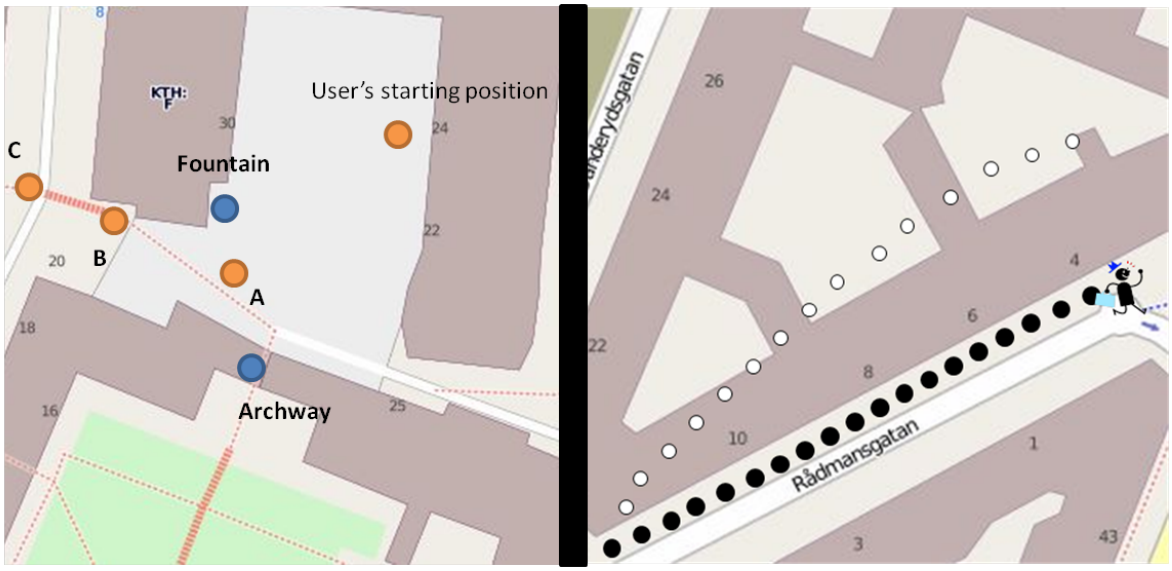

Figure 1: Generating route instructions.

the system first grounds the user's current position and orientation by means of reference landmarks in the near vicinity.

Consider for instance the situation depicted in Figure 1. Here the system seeks to describe the route given by the route planner, first to node A, then (when the user reaches A) to B, then down a flight of stairs to node C, then turn left, etc. Before giving instructions, our system first calculates if there is a clear line of sight from the user's assumed position to a number of reference landmarks. It then selects the most salient landmark, seeks to make the user aware of it, and describes the route relative to it. Here is a sample dialogue:

1. **System:** There is a fountain about 35 metres from here. Can you see it?
2. **User:** Yes.
3. **S:** Good! Please walk to the left of the fountain.
   (user walks)
4. **S:** Please turn right and walk to the top of the stairs.
5. **U:** What?
6. **S:** There is a flight of stairs leading down about 25 metres from here. Can you see it?

In utterance 1, since there is no good way of describing node A, the system cannot ask directly about it. Instead, the system calculates that there are two describable landmarks visible from the user's presumed position; a fountain and an archway, of which the fountain is considered most salient. When the user confirms (utterance 2), the system gives the next instruction with a reference to the fountain. If the user had answered in the negative, the system would have proceeded to ask about another visible landmark. If all possibilities are exhausted, the user is asked to simply start walking, so the system can adjust his course if needed.

Determining salience and producing good referential expressions is a difficult problem in general. Salience measures used by our system include rarity (rare objects such as fountains are more salient than entrances to buildings), distance, uniqueness, and familiarity (objects that have been mentioned before in the dialogue are considered more salient, and are easily described, e.g. "the fountain that you passed before").

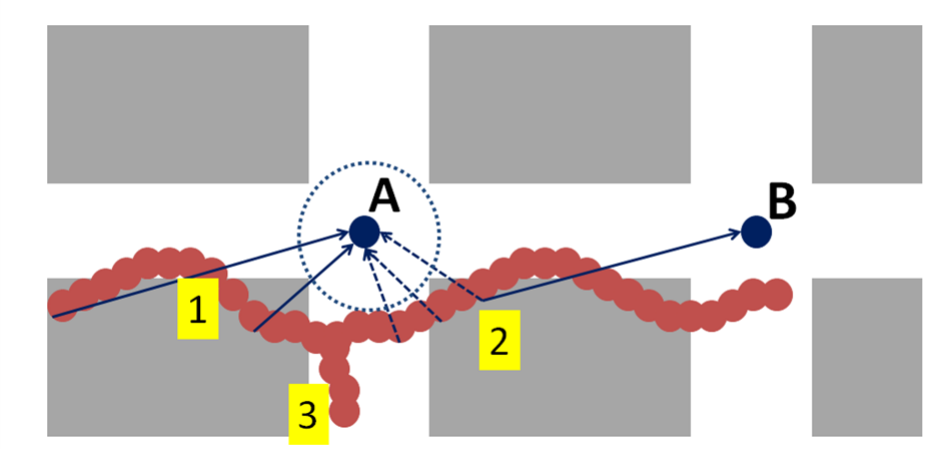

Figure 2: The user appears to "miss" the expected next node due to GPS drift.

2.3 Uncertainty and Replanning

The system knows the user's position by means of the GPS receiver in the user's Android device. When GPS readings indicate that the user is within 20 metres of the next node in the planned route, the system issues the next instruction. Furthermore, the system can also use the GPS readings to estimate whether the user has misunderstood the latest instruction and is going off in the wrong direction. In the latter situation, the system will replan the route.

Unfortunately, the so-called "canyon effect" ([Borriello et al, 2005]) can introduce inaccuracies into GPS readings, and these errors can be quite substantial. Figure 2 shows a typical situation, in which the user is walking along the street (from left to right in the picture), and where the GPS readings (in red) are incorrect a large part of the time. These inaccuracies are a problem for two reasons.

Firstly, the user can appear to "miss" the 20-metre circle around the next node, and appear never to come sufficiently close. The result will be that no instruction is produced by the system at that node. Secondly, the user can appear to walk in the wrong direction when in fact he is not. Consider the situation 2 depicted in Figure 2. The user has passed the next node A, but GPS errors have prevented the system from registering this. At 2, the user is getting further and further away from A, and since the system is still considering A to be the next node, it appears as if the user is going the wrong way. Clearly, it would be very misleading and confusing for the user if the system would say "Please turn around" at this point.

The method we have adopted to address these problems is illustrated above. As long as the distance to the next node A is decreasing (situation 1), everything is fine. If the distance to the next node starts increasing (situation 2), the system checks the distance to the next-next node B as well. In situation 2, the distance to the next-next node B is decreasing while the distance to the expected next node A is increasing. If this pattern persists for 10 seconds, the system assumes that the user has passed the expected node A and is continuing in the correct direction.

Another possibility is when the distance is increasing both to the expected next node A and to the expected next-next node B (situation 3). If this pattern persists for 10 seconds, the system assumes that the user is walking in the wrong direction, and will issue replanning.

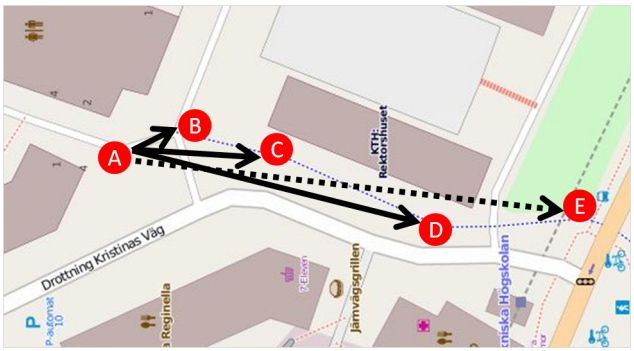

Figure 3: Weeding out nodes in a route plan using visibility information.

2.4 Using Visibility Information

The system repeatedly performs visibility calculations to find out whether there is a free line-of-sight between two given points. Such visibility calculations are currently used for three purposes: Firstly, as mentioned in Section 2.2, they are used to find candidates for referring expressions (the objects of which have to be visible from the user's assumed position). Secondly, they are used to produce better route plans. The system currently gets its data from OpenStreetMap ([Haklay, 2008]), and street objects in OpenStreetMap may contain many nodes very close to each other (in particular in roundabouts or curved streets). Consequently route plans can become very long. By iteratively weeding out any node visible from the preceding node, route plans become more suitable for narration. This process is depicted in Figure 3.

Here the produced route plan starts with the nodes A-E, in that order. The system checks visibility from A to B, from A to C, etc., and finds that D is the last visible node from A. It therefore concludes that B and C can be removed from the plan. The system then continues to check visibility from D to E, etc., until all unnecessary nodes are removed from the plan. However, no segment is allowed to be longer than 60 metres.

A third use of the visibility information is to produce better route instructions. The visibility calculations also return information on the streets, parks, etc., that were intersected by the visibility vector. This allows for instructions like "Now cross X street".

2.5 User Experiment

In order to evaluate the strategies described in Sections 2.2 to 2.4, we performed a user test with 8 subjects on 4 scenarios each. In each scenario, the user was guided to a specific spot in the city and asked to write down some inconspicuous detail (like the serial number on an electricity wiring box).

As a rough estimate of the success of the implemented strategies, we note that 7 users managed to complete all 4 scenarios (1 user only completed 1 scenario due to technical problems), and that, on average, the system had to replan 1.6 times per completed scenario.

User #	Scenarios completed	# Instructions	Duration (mins)	# Re-plannings
1	4	49	28.2	3
2	4	59	34.6	10
3	4	77	40.4	5
4	4	68	28.3	10
5	1	6	2.1	0
6	4	60	27.7	4
7	4	82	35.6	14
8	4	48	24.9	0

Table 1: User experiment

2.6 Discussion

A closer analysis of the results reveals that almost all users went wrong in one particular situation in which the system gave a relative instruction ("Continue straight ahead"). The system compared the direction of the preceding segment in the plan with that of the next segment, and found them to be similar. However, the system didn't have an explicit representation of the user's orientation, and failed to take into account that the user had stopped and turned 90 degrees in order to note the opening hours of a specific shop (in fulfillment of the first task). Because of this, almost all users misunderstood the system's reference "straight ahead'.

It's very difficult to estimate the user's orientation with any degree of accuracy. However, the user's speed might be a good predictor: When the user is standing still, or is moving very slowly, the spatial grounding strategy is preferable to the use of relative instructions like "walk straight".

3 Deriving Salience Models from Human Route Instructions

As discussed in the beginning of this report, it is preferable for a pedestrian navigation system like Spacebook to base its instructions on *landmarks*. However, even on this basic premise, there are a number of options to consider. At each decision point, there are a number of possible landmarks to choose from, and which one(s) to use in a specific route instruction is a difficult problem. In the literature, it is generally assumed that the candidate landmarks can be assigned a *salience* measure, by which they can be compared, and the most salient features are also the most suitable to use in route descriptions. Many researchers have proposed schemes for computing salience from a variety of factors (see e.g. [Duckham et al, 2010, Nothegger et al, 2004, Raubal and Winter, 2002, Xia et al., 2011]). These schemes are typically based on features like size, visibility and proximity of landmarks, and are intended to be valid for all users.

In this section, we investigate to what extent salience computations can be data-driven, that is, (semi-)automatically estimated from human route descriptions. Our aim is to create empirically motivated *personalized* salience models. Ultimately, we want to integrate such personalized salience models into the SpaceBook system. Two hypotheses underlie our work: Firstly, that salience is *user-dependent*. Secondly, if a user is asked to give a routing instruction in a specific situation, he would do so using the landmarks he himself thinks are most salient.

The second hypothesis suggests a kind of tuning mechanism for SpaceBook: Before being guided by the system, the user first walks around and describes the way he is going by means of landmarks. The system interprets the user's descriptions and uses them to derive a personalized salience model, which can later be used when guiding the same user in other parts of the city.

Two studies were performed, in which we let users walk a specified path in the city of Stockholm and ask them to describe it. From these descriptions we can obtain a model for computing salience scores of landmarks that describe their choices for each route segment. In the first study (Section 3.3.1), users describe a route posterior to having walked it on a two-dimensional map. In the second study (Section 3.3.2), users describe a way as they are walking it.

3.1 Deriving Salience Models

For the learning of salience models, we use the Large Margin Algorithm, introduced in [Fiechter and Rogers, 2000]. This algorithm has been used for various preference learning tasks, e.g. children's preferences in computer gaming [Yannakakis et al., 2009].

Each landmark can be described as a vector of numerical features, $\mathbf{x} = (x_1, \ldots, x_n)$ specifying costs along n dimensions. The dimensions might represent scalar attributes such as distance, or categorical attributes (e.g. 1 if the landmark is a restaurant, 0 if it is not). The salience $s(\mathbf{x})$ is a linear combination $\mathbf{w} \cdot \mathbf{x}$, where $\mathbf{w} = (w_1, \ldots, w_n)$ is the salience model that specifies the relative importance of the different features for the user. Naturally we do not assume that the user knows the values of his salience model, or indeed even that such a model exists. Instead we automatically infer the model as follows:

Whenever a person uses a landmark A in a description, he is preferring A over a number of other candidates that *could have been* used in the description but were not. That is to say that A has a lower cost according to the person's personal salience model than has any other candidate B, i.e. $\mathbf{w} \cdot (\mathbf{x_B} - \mathbf{x_A}) > 0$, where $\mathbf{x_A}$ and $\mathbf{x_B}$ are the vectors representing A, and B, respectively. Each route description from the user involving a landmark thus generates a number of inequalities, all in the form $\mathbf{w} \cdot (\mathbf{x_{B_i}} - \mathbf{x_{A_i}}) > 0$, for $1 \leq i \leq m$. Our goal is to find appropriate values for the weights in $\mathbf{w}$ that satisfy all these inequalities. This can be done by solving the following linear optimization problem, e.g. with the Simplex method [Papadimitrou and Steiglitz, 1982]:

$$\begin{array}{lll} \text{minimize} & \sum_{j=1}^{n} w_j & \\ \text{subject to} & \mathbf{w} \cdot (\mathbf{x_{B_i}} - \mathbf{x_{A_i}}) \geq 1, & 1 \leq i \leq m \\ & w_j \geq 0, & 1 \leq j \leq n \end{array}$$

This formulation of the problem assumes that a person is always consistent in his preferences. For the case he is not, we use a slightly extended version of the basic Large Margin Algorithm, in which we introduce one slack variable per inequality, and add a penalty c on those variables (see [Fiechter and Rogers, 2000] for details).

$$\begin{array}{lll} \text{minimize} & \sum_{j=1}^{n} w_j + c \cdot \sum_{i=1}^{m} \xi_i & \\ \text{where} & \mathbf{w} \cdot (\mathbf{x_{B_i}} - \mathbf{x_{A_i}}) + \xi_i \geq 1, & i = 1 \ldots m \\ & w_j \geq 0, & j = 1 \ldots n \\ & \xi_i \geq 0, & i = 1 \ldots m \end{array}$$

For the present problem, we found that $c = 0.1$ was an appropriate value.

3.2 Problem Encoding

3.2.1 OpenStreetMap

For geographic data, we are relying on the OpenStreetMap (OSM) geographic database [Haklay, 2008]. OSM is a freely available crowd-sourced database used in different areas of research, e.g. in robot navigation [Hentschel and Wagner, 2010], in indoor navigation [Goetz, 2012], and in pedestrian navigation [Rehrl et al., 2011].

It has two basic data structures:[1] *nodes* and *ways*. Nodes can represent entities in their own right, e.g. intersections, bus stops, or house entrances, but they can also act as the building blocks of *ways* (sequences of nodes). Ways are used to represent street segments, buildings, or areas. In what follows, we will avoid the polymorphic term "way", and rather talk about buildings, streets, etc.

OpenStreetMap data is categorized according to an extensive scheme of tags[2] that specifies, for example, how an entity can be tagged as a shop, how names are added, or how to indicate speed limits on different parts of a road. Since the data is crowd-sourced on a voluntary basis, it tends to contain inconsistencies in the way tags are applied. Furthermore, the large number of tags results in a separation of entities that ontologically belong together, e.g. different segments of the same street are separate entities in OSM, because they have different speed limits, or because a bus line is using part of the street.

In our first study (cf. Section 3.3.1), we are using only nodes from OSM. If the user refers to a street, we compare whether a predicted landmark, i.e. node, is part of that street. Buildings are excluded from this study.

In the second study (cf. Section 3.3.2), we are including streets and buildings and also slightly modify the database in two ways: First, we combine street segments that have the same name into one. This modification results in a reduction in the number of street segments by 656. Second, we modify the naming conventions of addresses for nodes. Many nodes are assigned their address, i.e. a streetname and a housenumber, as a "name". However, a node, e.g. a shop, can have both a name and an address. We therefore modify the database to consistently assign these tags.

3.2.2 Example

Figure 4 shows a situation in one of our experiments where the subject, standing at "A", chooses to describe the way using two landmarks, indicated by the wider lines "L1" and "L2": *"I continue in a southwesterly direction down the steps towards the arch"*. The position of the succeeding instruction is indicated by "B". The stairs and the archway are indicated by "L1" and "L2", respectively.

Every landmark belongs to the *context* of its closest road node. When describing the path from A (the *starting position* of the segment) to B (the *goal position* of the segment), the positions are mapped to their closest nodes and all landmarks in the contexts of these two nodes are possible referents. We will refer to this set of landmarks as the *candidate set* for A and B. In the figure, this set is visualized as square-shaped icons (for nodes), wide lines (for roads, paths, etc.), or striped shapes (for buildings).

[1] We are disregarding OSM *relations* for the time being.

[2] `http://wiki.openstreetmap.org/wiki/Map_Features`

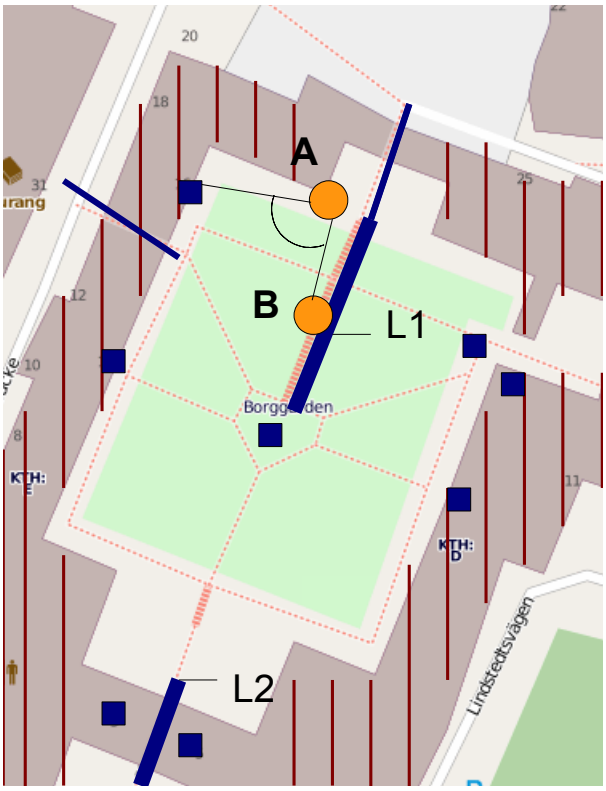

Figure 4: An example segment for the utterance: "I continue in a southwesterly direction down *the steps* [L1] towards *the arch* [L2]" A and B indicate the start and the goal position respectively.

3.2.3 Features

The method described in Section 3.1 requires every landmark L to which the user can refer to be modelled as a vector of features. In this experiment, we use a vector of 12 features that are computable from our geographic database. These features form an initial set of structural landmark features [Sorrows et al., 1999] and we are planning to further explore which other features are important for computing salience. The features used here are the following:

Distance between the user's position A and the landmark L.

Distance between the landmark L and the goal node B.

- In the case where the landmark is a road or building, distances are computed as the minimum of the distances to each of the nodes that make up the road or building.

Angle between the lines AL and AB.

- In the case where the landmark is a building, the angle is computed as the average of the angles when using each of the nodes in the building to compute the line AL.

Name Categorial attribute having the value 1 if the landmark has a name (e.g. "7-Eleven"), or belongs to something that has a name, e.g. a node on X street, and 0 otherwise.

Type These 8 features represent the type of the landmark according to whether they belong into the categories *road network*, i.e. the landmark node is part of a street, *building*, *eating & pleasure*, e.g. a restaurant or a theater, *shops*, *entrances*, i.e. a specific house number on a street, *areas*, e.g. a park or a construction site, *structures*, e.g. a statue or a fountain, or *other*. Each landmark is of at least one type, which is indicated by the value 1 in the corresponding slot. All categories are OSM categories.

In the example in Figure 4, the stairs that the user refers to (the wider line close to the goal point B), is represented by the vector $(2.0, 0, 92, 0, 1, 0, 0, 0, 0, 0, 0, 0)$. The first two positions contain the distances (the 2-logarithm of the actual distance in metres, rounded to the nearest integer). The third position represents the angle (in degrees). The succeeding slot indicates that the landmark does not have a name. The values in the final 8 slots indicate that the landmark is a kind of road, but no other type. All features are normalized within the segment in which they appear. Distances and angles are then relative between the landmarks that appear in the same segment.

3.3 Data Collection

3.3.1 Study 1 – Describing Offline

6 engineering students (4 male, 2 female, average age 25) were asked to describe a route to someone unfamiliar with the area, imagining that they were talking to this person on the phone. The subjects reported to be fluent in English. The subjects had just walked the same route themselves and should

therefore remember it well. To further help them recall their trajectory, they were also shown their route on a map on the screen by a moving mouse cursor (i.e. without using speech), and they could also look at the map while they described the route.

Prior to describing the route, the subjects had walked them themselves, following instructions given by our prototype system. This means that their own instructions might be influenced by what they just heard. However, the system's instructions only partly used landmarks and otherwise relied on relative instructions such as "turn left". This strategy sometimes resulted in ambiguous or wrong instructions, and the subjects were asked to "improve upon the system's behavior". The route they had walked was approximately a kilometer.

3.3.2 Study 2 – Describing Online

For this study, we asked 5 subjects (4 male, 1 female, average age 29) to walk a specific route and describe their path in a way that would make it possible for someone to follow them. We thereby put participants into the environment in which we would later like to guide them. Instead of reading information from the 2-dimensional map, our subjects can now see the environment in the same way as later users of SpaceBook experience it.

The experiment was set up as a Wizard-of-Oz situation in which the subjects were asked to describe to a spoken dialog system with the task of making it understand. They were told that the system, like them, had a 3-dimensional and 1st-person view of the environment. The subjects were not instructed to interact with the system in any special language but were advised to try out what they thought was suitable and that the system would ask them if it needed clarification, in which case they should stop until the situation was clarified. In this way, the experimenter was able to interfere in situations where an instruction was evidently ambiguous. Otherwise, the experimenter took as little initiative as possible in order to avoid influencing them in their choice of landmarks.

The subjects reported to be fluent in English. Two of them reported to be only slightly familiar with the area, three reported to be familiar. All were able to complete the task.

The subjects were equipped with an Android mobile phone (Motorola Razr) that ran an application which allowed us to record their GPS coordinates and speech signal cf. [Boye et al, 2012], [Hill et al, 2012]. It also allowed to send messages from the experimenter to the subject via text-to-speech (TTS). The experimenter sat in a laboratory and used an interface which allowed him to see the subject's position on a map and type messages that were sent to the subject via TTS.

Speech signal and GPS coordinates were automatically logged and time-stamped, thereby allowing to match speech transcriptions with the subject's GPS coordinates.

The route that the subjects were asked to walk was a round tour that started and ended outside the doors of our laboratory. The route was approximately two kilometers long and was given to the subjects on a silent map which is shown in Figure 5. The map had street and other names removed, as well as common symbols, e.g. for churches or bus stops.

One of the subjects deviated slightly from this given route, all others followed the path. Subjects could choose in which direction to start the tour, two chose one direction and three the other. The subjects took on average 28 minutes and 25 seconds to complete the round tour.

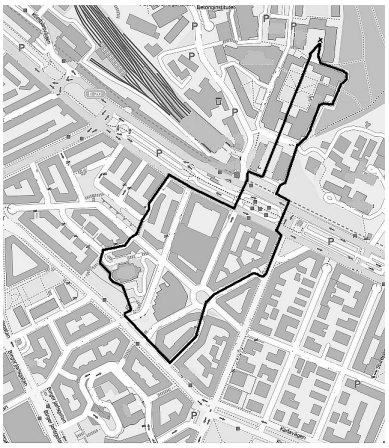

Figure 5: The map that the subjects were asked to follow.

3.3.3 Analysis

In both studies, the subjects' speech was recorded and segmented according to route segments before transcription. In Study 1, all utterances were instructions ("walk towards the church"), while in Study 2, there was a large number of purely descriptive utterances ("and then you can see a church to your left"). We tagged these two types accordingly, looking only at lexical cues, i.e. the choice of verb, and restrict our analysis here to those segments that are instructive, i.e. those segments that specify a connection between a starting point A and a goal point B. Route segments were also annotated with their start and goal nodes, which were either inferred from the subjects' instructions (Study 1), or by mapping the subjects' GPS coordinates to the closest nodes in the database (Study 2).

Each of these segments is then annotated with all landmarks from our geographic database that the subject referred to. In the example in Figure 4, the GPS coordinates indicate where the instruction was given and where the next instruction followed. In this example, the subject referred to two objects, "the steps" and "the arch". Both these objects are OSM *ways* and indicated by lines in the figure.

For each subject, we thus have a number of annotated segments, each consisting of a start node, a goal node, and at least one landmark that the subject referred to (his *preferred* landmark(s) in this segment). Segments where the subject didn't refer to anything at all were excluded from this experiment. The candidate set for the segment (i.e. the landmarks the user *could have* referred to) was automatically computed from the OSM database and contains on average 22 landmarks in Study 1 and 30 landmarks in Study 2 (cf. the differences in the databases between the two studies, Section 3.2.1).

The preferred landmarks might or might not be part of the candidate set. There are two possible reasons for a preferred landmark not to be part of the candidate set: Either the user referred to something that is not in the database at all, or he referred to something that is farther away, and doesn't belong to the context of neither A nor B. In both cases, we removed the reference.

An *instance*, of the salience model learning problem, then, is a candidate set together with one or several preferred landmarks, at least one of which is part of the candidate set. The set of all instances for a particular user was split into a training set and a test set. The training set, typically one third of the segments, was used to derive a salience model $\mathbf{w}$ according to the method presented in Section 3.1. In

Study 2, this was repeated several times and averages were computed (cf. the larger number of segments for each subject, Table 2). To evaluate **w**, the salience of each member of each instance of the test set was computed. A *successful* instance is one in which one of the preferred landmarks had the best salience according to **w**. The number of successful instances in the test set is an indicator of how well the learned salience model actually reflects the preferences of the user.

3.4 Results

For evaluation, we used the induced weights to compute costs on test sets and counted in how many cases the best option was a landmark used by the subject.

The results from both studies are presented in Table 2. The table shows the total number of available segments for each subject (SEGMENTS) and how many of these segments were used for testing (TESTS). In Study 2, we used n-fold cross-validation with differing size and number of folds and average over the results. The size of the test sets was usually two thirds of the segments. The overall success rate is 69% in Study 1, and 23% in Study 2.

In Study 1, for all individual salience models, at least half of the test instances are successful. In one case, the model even returns all the instances as successful. To get an insight into how well the models perform on those landmarks that did not receive the lowest cost but were used by the subject, we also compute the measure RANK. For this measure, we compute the percentage of landmarks receiving costs that were equal or higher than the preferred landmark's cost (recall that the lower the cost, the more salient the landmark). The number of landmarks that can be referred to differs depending on the particular route segment and this measure reflects how high the salience model ranked a landmark in comparison to all available landmarks. For example, subject 1's model has two successful test instances, and in the other two ranks the preferred landmark as 3 of 14 in one instance, and as 5 of 39 in the other.

Note that the ratio of training vs. testing segments differs between the subjects. Initially, the training set contains two thirds of the route segments. For some subjects, the training size had to be reduced, because our algorithm is limited in the number and size of route segments it can process.

In Study 2, additionally to the RANK measure, we also report in which position the preferred landmark was ranked on average (POSITION). For example, subject 2's model was successful in 20% of the test instances. On average, 82% of the landmarks in each candidate set received a lower salience measure and the preferred landmark was ranked between positions 4 and 5.

3.5 Discussion

The method manages to mimic the user's salience preferences in 69% of the cases in Study 1 and in 23% of the cases in Study 2. For Study 2, this means that from a list of (on average) 30 possible landmarks, the learned model selects the same landmark as the user almost 1 time out of 4. Although much better than chance, this figure is significantly worse than in Study 1. There are several possible explanations for this discrepancy. The route walked by the users in the present study was longer and more complex, and the sets of candidate landmarks were bigger (on average 30 landmarks compared to 22 landmarks in the previous study). But more importantly the users had access to more geographical detail when giving a route instruction at the place where it is actually supposed to be interpreted, rather than from a remote location.

However, we believe that the figures can be improved if the model could be trained on more examples. Our

Table 2: Evaluation of the derived salience models. *indicates averages

	SUBJECT ID	SEGMENTS	TESTS	SUCCESSFUL	RANK	POSITION
Study 1	1	13	4	2 (0.50)	0.93	2.3
	2	16	5	3 (0.60)	0.94	1.8
	3	9	3	2 (0.67)	0.94	2
	4	9	3	2 (0.67)	0.94	1.7
	5	16	10	7 (0.70)	0.95	1.9
	6	12	4	4 (1.00)	1.00	1
Average				**20 (0.69)**	0.95	1.8
Study 2	1	20		0.22*	0.52*	11.4*
	2	10		0.20*	0.82*	4.5*
	3	14		0.22*	0.72*	8.9*
	4	23		0.23*	0.86*	5.2*
	5	29		0.28*	0.89*	3.1*
Average				**0.23***	0.76*	6.6*

current algorithm is limited in the number and size of route segments it can process. Future work includes using the revised simplex method [Maros, 2003] in order to cope with larger training sets. Furthermore, it is very likely that a closer analysis reveals that the candidate sets can be restricted a priori, because there are certain kinds of landmarks that few users ever refer to. Removing these landmarks from the candidate sets would also decrease the size of the problem.

A noteworthy finding was that the value of the "has-name" component of all induced salience models was close to zero for all users. This means that all the users in our study tended to prefer landmarks that have names, rather than landmarks that do not.

We also plan to analyse in detail whether the individual preference models all have something in common (i.e. whether there are general properties of salience models that always hold). The results of such an analysis might allow us to restrict our candidate sets, thereby making it possible to build the models from more examples. Furthermore, we aim to investigate which other features, apart from the ones we are considering in this article, are important for the salience computation problem.

4 Spatial reference resolution

A key problem in the Spacebook project is the generation and interpretation of natural language references to objects in the city. Such references form the link between the algebraic and geometric model of the domain on the one hand, and the communication with the user on the other. For navigational systems such as Spacebook, the generation problem is more important, whereas for a simulated user it is crucial to correctly interpret instructions like "Turn left onto Cowgate", "Walk towards Camera Obscura", or questions like "Can you see the statue" in order to be able to follow instructions given by the system. This section briefly describes our approach to this problem.

At any point in time, a simulated pedestrian (henceforth called *S*) has a specified position, direction, and speed. It has a limited awareness of its geographical environment, insofar that it will not walk into

buildings, but rather move on streets and pavements, across squares, etc. [3]. To be able to do this, *S* retrieves data from the OpenStreetMap geographic database. However, as *S* is supposed to be a simulation of a pedestrian with limited knowledge about the city, *S* does not have total access to the database, and cannot directly find the point in the city corresponding to a given referential expression (like "Camera Obscura"). Rather is simulates vision by continuously calculating line-of-sight information from its current position to the landmarks in the near vicinity.

4.1 Spatial contexts and visibility

A visibility engine, implemented by Liquid Media, can calculate whether or not there is a free line-of-sight between any pair of points. *S* repeatedly calls this engine on the set of points in its vicinity in order to construct of a model of its visual environment.

Figure 6: Recursive spatial partitioning of an area in the city in smaller and smaller quadrants.

The current portion of the Edinburgh map used by *S*, roughly a square 2500 metres along the side, contains 25410 OpenStreetMap nodes. A linear search through the whole list of nodes in order to calculate the distance to each node would therefore be too time-consuming. The nodes are therefore organised in a so-called *quad-tree* (see e.g. [Pritchard, 2001]) which allows for logarithmic lookup times. The top vertex of the quadtree represents the whole area. The area is then divided in four equally sized quadrants, each of which is represented by a child vertex. Each of the four quadrants is then divided into sub-quadrants, and the same process continues recursively a fixed number of steps, or until a sub-quadrant contains no

[3] Movement generation will be further described in Deliverable 5.3.1.

Figure 7: Visible points (blue, marked by "V") and non-visible points (red, marked by "N") in the lower left subquadrant of the area shown in Figure 6. The user's position is indicated with a "U" (green dot).

nodes at all. Due to the hierarchical data representation, it is also straightforward and quick to find all nodes within 10 metres, 20 metres, 40 metres etc. from a given point. Therefore, if an object is not found in the closest vicinity of *S*'s current position, the search area can easily be extended.

4.2 Goals

The simulated pedestrian maintains a queue of goal nodes to be visited. For instance, instructions like "Go towards the bus stop" followed by "...and then continue to Hill Place" would put two nodes in the goal queue; the node corresponding to the bus stop, and the node (presumably) corresponding to the intersection of Hill Place and the street *S* is currently moving along. The first node in this queue will be referred to as the *next goal node*. If no instruction has been given to *S*, it will find a next goal node itself, by continuing in (roughly) the same direction as it is currently moving. With a small probability, it may also decide to change direction.

4.3 Representation of utterances

A parser translate natural-language utterances into expressions in a meaning representation language (MRL), which is then further processed to extract the question or instruction that the utterance expresses. For instance, an instruction like "Turn left at the junction and walk towards Starbucks on East Crosscause-

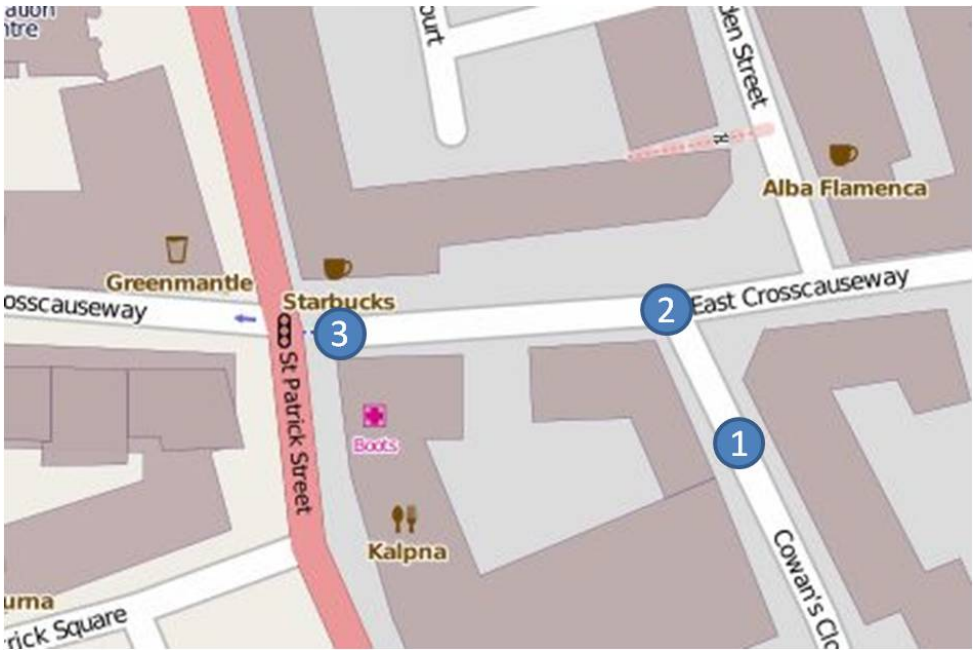

Figure 8: The geographical context of the utterance: "Turn left at the junction and walk towards Starbucks on East Crosscauseway". The indicated point 1 is the user's position, 2 is the turning point, and 3 the aim point, respectively.

way", would be represented by:

$$\begin{aligned}
&\texttt{dialogAct}(\texttt{inform},\texttt{X}),\\
&\texttt{X}:\texttt{turn}(\texttt{left},\texttt{A},\texttt{B},\texttt{C}),\\
&\texttt{isA}(\texttt{A},\texttt{junction}),\\
&\texttt{isA}(\texttt{B},\texttt{cafe}),\\
&\texttt{isNamed}(\texttt{B},\texttt{starbucks}),\\
&\texttt{isA}(\texttt{C},\texttt{street}),\\
&\texttt{isNamed}(\texttt{C},'\texttt{eastcrosscauseway}')
\end{aligned}$$

In this expression, the variables A, B and C are implicitly lambda-bound, and the purpose of the spatial reference resolution mechanism is to find the identifiers of the nodes that the speaker most likely referred to.

The key semantic predicate for instructions is

$$\texttt{turn}(\texttt{Direction},\texttt{TurningPoint},\texttt{AimPoint},\texttt{Street})$$

Different instructions may constrain the possible values of one or more of the four arguments of the predicate. The utterance above ("Turn left at the junction and walk towards Starbucks on Nicolson Street") constrains all four, whereas "Turn left" only constrains the first, and "Go towards Starbucks" only the third.

In order to find concrete nodes to fill in the T*urningPoint* argument, the set of nodes visible from the user's position, and the set of nodes visible from the next goal node (if this node is defined) is calculated, and a node matching the description is sought among these nodes. The A*imPoint* and S*treet* are not required

to be in view, so the whole set of nearby nodes is searched. (A possible improvement would here be to search for the A*imPoint* once *S* has reached the T*urningPoint*).

The resolved utterance then becomes:

$$\begin{aligned}&\texttt{dialogAct}(\texttt{inform},\texttt{X}),\\&\texttt{X}:\texttt{turn}(\texttt{left},\texttt{21135018},\texttt{B},\texttt{23614881}),\\&\texttt{isA}(\texttt{21135018},\texttt{junction}),\\&\texttt{isA}(\texttt{2156953057},\texttt{cafe}),\\&\texttt{isNamed}(\texttt{2156953057},\texttt{starbucks}),\\&\texttt{isA}(\texttt{23614881},\texttt{street}),\\&\texttt{isNamed}(\texttt{23614881},'\texttt{eastcrosscauseway}')\end{aligned}$$

where the lambda-bound variables of the unresolved expression have been substituted with identifiers of OpenStreetMap nodes and ways.

For the interpretation of "left" and "right", let P be the previously visited node, and T the turning point. To interpret relative references like "left" or "right", the bearing of the vector PT is compared with the vector TN, for each node N visible from T. Differences in bearing of $\pm$ 25 degrees are considered to be "straight ahead", from 25 to 120 degrees is "right", -25 to -120 is "left", and the remaining values are considered to be the opposite direction of the current one.

5 AI planning for pedestrian navigation instructions

5.1 Introduction

Automated Planning is the field of Artificial Intelligence devoted to select the next action to apply given a model of the environment where the decision making takes process.

Planning differs from other approaches to decision making by tackling a model-based approach for selecting the action to do next. This gives to the tool, called *planner*, many advantages that fit the needs of the Spacebook project. In fact, using a model to describe a pedestrian navigation problem is a very flexible tool as , once encoded in what is called a planning domain, an unique domain-independent solver can handle several different problems relative to the task of interactively guiding a pedestrian user.

By using a model-based approach, the behaviour of the system is not coded before hand; rather the planner automatically generates the best sequence of actions to satisfy a given goal. The system is then able to instantanously adapt to the distinct tasks of pedestrian navigation in different areas, as the problem to be solved is automatically generated from the available geographical data. The same occurs for the user's model: once the different behaviours of the pedestrian are captured in the actions available in the problem, the planner selects the most adequate action for the given situation and user via a search algorithm; such an action is then translated to a natural-language utterance for the user's benefits. There is no need to learn a policy either, as the best action to apply – given the model – is derived by the planner at run-time.

The interaction with the user is here integrated into the planning task, and constitutes an important aspect of it. Interactivity allows to build dialogues based on the pedestrian's response, and gives the possibility of replanning using a different instance of the planning model when the task changes or some unexpected event occurs.

5.2 The models for automated planning

The *classical* model for planning is a common restriction of the more general problem of selecting actions to reach a desired objective. Actions are described in terms of pre-conditions and post-conditions applicable in the current situation. Here, the actions are assumed to be deterministic and the information about the environment, complete. The model is completed then with information about the initial state, and the goal state. Classical planning can thus be cast as a path finding problem in a graph whose nodes are the states, and whose edges are the transitions that are possible, described in terms of actions. A solution plan is then a path from the node in the graph representing the initial state to a goal node representing a goal state of the problem. A plan is then a linearly ordered finite sequence of actions.

An approach that has been proved to be effective to find solutions for a planning problem relies on the use of heuristic search. Heuristic search uses heuristic functions to evaluate the cost-to-go from a node to a goal, or to be more general, to provide a ranking of a set of nodes based on their relative desirability [Bonet and Geffner, 2001]. This estimation of the distance in the search space is then used by the search algorithm to drive the state space search, preferring to visit nodes considered more promising from their heuristic value.

Classical planning suffers of many limitations, due to the strong assumptions of determinism and full observability. For planning in real world domains, a *contingent planning* model is used instead, that extends the classical model with uncertainty in the initial situation, and sensing actions. Sensing actions generally have non-deterministic outcomes, one for each possible observed state. This enriched model is the one that we use here.

5.3 Dialogue as a Planning Model

As we discussed previously in Section 5.1, techniques borrowed from automated planning allow for data-driven system development and automatic optimisation, and even on-line adjustment to user behaviour.

A dialogue can be viewed as a two agents planning problem: actions are used to interchange information or knowledge about the environment, in order to reach a final situation where both agents achieve their own goal. So sentence generation problems can be represented as planning problems under incomplete information, and with some sensing available.

A multi-agent planning problem, and in particular a two-agent problem, can be cast as a single agent planning problem with incomplete information and sensing [Weerdt et al., 2005], as agents act independently, and no coordination issues arise, meaning that the agents are not competitive nor their plans are conflicting. When communication is possible between such agents, and requests from one of them are executed by the other one, we face the multiple agents model commonly known as *Master–Slave*. Certain narrative models are similar to a master–slave planning problem: a single centralised agent (the planner that composes the story as a plan) distributes the actions in the plan to the different "slave" agents, the characters that are actors of the actions in the story. This model has been successfully compiled as a planning problem, and therefore solved by an off the shelf planner [Haslum, 2012].

In practice, single-agent planning problems that feature incomplete information and sensing are solved using classical planners and replanning. Based on that idea, we developed a software module for modelling system utterances, based on the K-replanner from Blai Bonet [Bonet and Geffner, 2011], and that makes use of lookahead for action selection, and heuristics to evaluate the nodes during the search.

The heuristics used by the planner address two orthogonal aspects of planning with incomplete informa-

tion: the distance of the current situation from the goal, and the disambiguation of the uncertainty bound to the current situation. These combined heuristics provide the search with a sense of direction that has already been shown to be effective for planning under uncertainty [Albore et al., 2011], and they deal with the effective aspects of generating system utterances for the tasks of guiding a pedestrian.

In a navigation scenario, system utterances are actually generated to achieve two main goals: 1) to guide the pedestrian toward his objective using appropriate route-giving instructions, and 2) to reduce uncertainty about the pedestrian's knowledge and position (e.g. when GPS is off). The output of the planning module is a plan, which includes requests (of information, of performing an action) to the pedestrian, while the input are gathered from different sensors, which can be active like the pedestrian's answers, or passive like GPS coordinates.

To illustrate the way system utterances are derived by a planning task, we use an example domain, very close to the example shown in Section 2.2.

5.3.1 Example

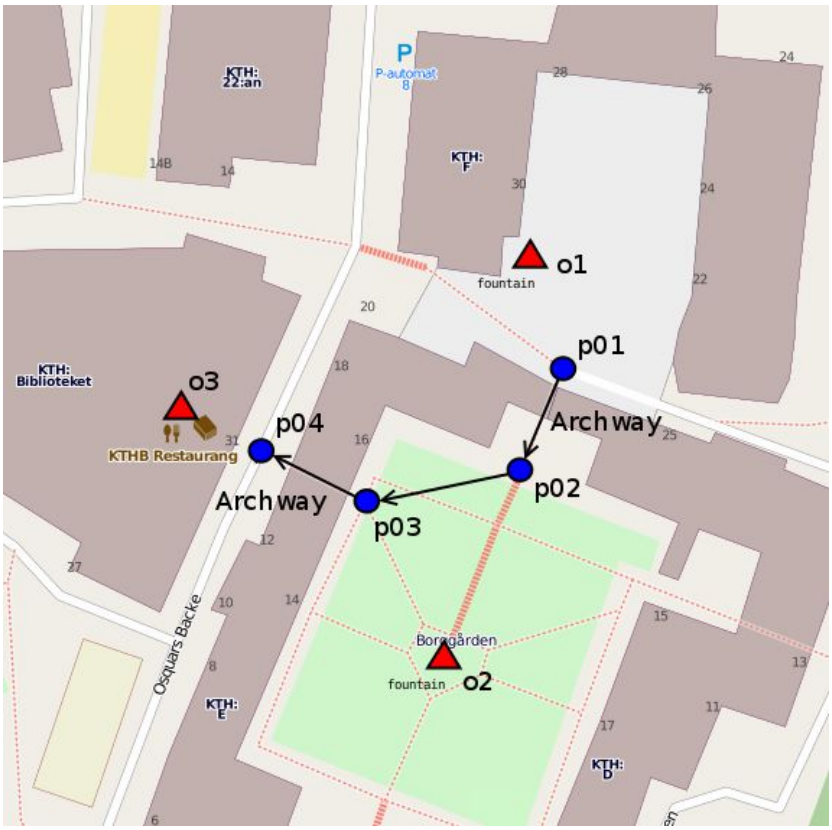

Figure 9: The planning task. The route is drawn by arrows, the position nodes are blue circles, and the describable objects are red triangles.

The task is to generate a system utterance that describes how to get from the initial state to the next nodes in a planned route, in order to eventually reach the given goal. The route (cf. Figure 9) is encoded in as a sequence of nodes, each one reachable from the previous one by applying a referential action. The route

goes from node "p01" (initial situation) to node "p04" (goal). The referential actions are actions in the model that use of different knowledge about landmarks, which are in turn automatically extracted from the geographic data. The cognitive model of the user includes a certain amount of uncertainty: it is not certain what he sees from his current position, nor which objects on the route are familiar or known to him.

So the plan will include sensing actions, in order to reveal the user's knowledge about the landmarks, as this knowledge is necessary to apply an action that is referring to it. Below, an example of the encoding that shows the planning action of moving from p01 to p03 using the fountain o2 as a reference landmark, as it is written in PDDL, the language commonly used to encode planning problems:

```
(:action instruct-goto-object-p01-p03-o2
   :precondition (and (at p01) (visible p03 o2) (sees o2) )
   :effect  (and (not (at p01)) (at p03) )
)
```

The preconditions of the action require the pedestrian to be at node p01, and that the fountain o2 is possibly visible from the destination node p03, which allows to use the fountain in an utterance describing how to reach p03 from p01. The effect is that the pedestrian is in p03, and not at p01 (he moved). The predicate "sees o2" encodes the knowledge that the pedestrian is seeing the fountain o2. This information is gathered from a direct question asked to the pedestrian, i.e. a sensing action:

```
(:observation ask-can-see-p01-o2
   :precondition (and (at p01) (visible p01 o2))
   :observe (sees o2)
)
```

Here, the precondition states that, in order to ask to the pedestrian if he sees the fountain o2, he has to be in node p01, from which the fountain o2 is visible; the latter information is extracted from the geographical data. The observed variable of the problem is "sees o2", which can be true or false after performing this sensing action. A possible execution of a plan is the following:

1. `ask-can-see-p01-o2`
   (the pedestrian answers affirmatively)
2. `instruct-goto-object-p01-p03-o2`

The previous plan is then translated in the following system utterances:

1. **System:** There is a fountain about 35 metres from here. Can you see it?
2. **User:** Yes.
3. **S:** Good! Please walk to the right of the fountain.

The plan expresses actions that might be executed over an extended time duration. The next-to-apply action is provided to the pedestrian and, only once this action has been fully executed, the next action is sent, even if the planner already synthesised a full plan to the goal. In case new information is available or an unexpected event happens, the planner leaves execution mode and constructs a new plan for the new situation that has arisen.

To guide a pedestrian along a route given by the route planner, referential actions encode the different instructions about how to go from a node to the next one. In order to do so, the model represents objects that can be describable (or not) and visible (or not) along the route, extracted from the saliency model, and including visibility information about the landmarks. As we just saw, the planner uses the knowledge about the landmarks as preconditions for the actions.

Sensing actions, or observations, allow to uncover the truth value of an hidden variable of the problem. In the specific case of a pedestrian navigation domain, the observation is a question directly addressed to the user, in order to gather the information needed for plan generation (e.g. which landmark is visible). The result is an interleaving of planning and execution phases, where the planning generates the next-to-apply sequence of instructions, and execution transmits the instructions one by one to the pedestrian. The execution is interrupted when a new question is asked, the goal is reached, or a replanning episode is triggered. In these cases, the current state is updated with the new facts, and a new plan is generated.

5.4 Discussion

The success of the model-based approach has permitted to develop an interactive platform that automatically generates system utterances from a given pedestrian domain. The planning domains are tailored on the navigation tasks, and the produced plans can be used with flexibility, as it is always possible to find a different system utterance by replanning from a given situation, and from dynamically produced data. The planning model also embeds a partial cognitive model of the user, based on the accessibility of the user's knowledge and memory about passed observations. The cognitive model is implicit in the planning domain, encoded as different route-giving instructions, depending on the referential objects available in a given situation. Referential actions, used to instruct how to move between two nodes, can then depend on the name of the object (e.g. "Go to the fountain"), on the nearest landmark (e.g. "Go to at the right of the fountain"), or just on an instruction (e.g. "Go straight ahead"). The actions are chosen using the knowledge available through observations, but it is not excluded that they could depend on a more refined model of the pedestrian (e.g. a local has more chances to know the fountain than a tourist). This direction seems promising, as it allow to dynamically select the most adequate action for each situation and user. The development of future planning domains will then explore further this aspect of the problem.

References

[Albore et al., 2011] A. Albore, M. Ramirez, and H. Geffner. (2011) Effective heuristics and belief tracking for planning with incomplete information. In *Proc. of Int. Conf. Automated Planning & Scheduling (ICAPS-11)*, pages 2–9, Freiburg, Germany, 2011.

[Bartie and Mackaness, 2006] Bartie, P. and Mackaness, W (2006). Development of a speech-based augmented reality system to support exploration of cityscape. Transactions in GIS, 10(1):6386.

[Bonet and Geffner, 2001] B. Bonet and H. Geffner. (2001) Planning as heuristic search. *Artificial Intelligence*, 129(1–2):5–33.

[Bonet and Geffner, 2011] B. Bonet and H. Geffner. (2011) Planning under partial observability by classical replanning: Theory and experiments. In *Proc. of Int. Joint Conf. on Artificial Intelligence (IJCAI-11), Barcelona, Spain*.

[Borriello et al, 2005] Borriello, G., Chalmers, M., LaMarca, A. and Nixon, P. (2005) Delivering real-world ubiquitous location systems. Communications of the ACM, vol 8, issue 3, pp. 36-41.

[Boye and Gustafson, 2005] Boye, J. and Gustafson, J. (2005) How to do dialogue in a fairy-tale world. Proceedings of the 6th SIGDial workshop on discourse and dialogue, Lisbon, Portugal.

[Boye et al, 2006] Boye, J., Gustafson, J. and Wirn, M. (2006) Robust spoken language understanding in a computer game. Journal of Speech Communication, 48, pp. 335-353.

[Boye et al, 2012] Boye, J., Fredriksson, M., Götze, J., Gustafson, J. and Königsmann, J. (2012) Walk this way: Spatial grounding for city exploration. *Proc. IWSDS workshop*, Paris, France. Extended version to appear in *Towards a Natural Interaction with Robots, Knowbots and Smartphones*, Springer-Verlag, 2013.

[Cai et al, 2003] Cai, G. Wang, H. and MacEachren, A. (2003) Communicating Vague Spatial Concepts in Human-GIS Interactions: A Collaborative Dialogue Approach. In Spatial Information Theory: Foundations of geographic information science, LNCS Volume 2825/2003, pp 287-300.

[Cuayáhuitl and Dethlefs, 2011] Cuayáhuitl, H. and Dethlefs, N. (2011) Spatially-Aware Dialogue Control Using Hierarchical Reinforcement Learning. In ACM Trans. on Speech and Language Processing (Special Issue on Machine Learning for Adaptive Spoken Dialogue Systems), vol. 7, no. 3, pp. 5:1-5:26

[Denis et al, 1999] Denis, M., Pazzaglia, F., Cornoldi, C. and Bertolo, L. (1999) Spatial discourse and navigation: an analysis of route directions in the city of Venice. *Applied cognitive psychology*, vol 13, no 2.

[Duckham et al, 2010] Duckham, M., Winter, S. and Robinson, M. (2010) Including landmarks in routing instructions. *Journal of location based services*, vol. 4, no. 1, pp. 28–52.

[Fiechter and Rogers, 2000] Fiechter, C-N. and Rogers, S. (2000) Learning subjective functions with large margins. *Proc. 17th ICML*, pp. 287–294.

[Goetz, 2012] Goetz, M. (2012) Using Crowdsourced Indoor Geodata for the Creation of a Three-Dimensional Indoor Routing Web Application. *Future Internet* 4(2), pp. 575–591.

[Google, 2012] Google Inc. (2013) Google Maps for mobile. http://www.google.com/mobile/maps

[Google, 2013] Google Inc. (2013) Google Maps Navigation. http://www.google.com/mobile/navigation

[Götze and Boye, 2013] Götze, J. and Boye, J. (2013) Deriving salience models from human route instructions. *Proc. CoSLI workshop on Computational models of spatial language interpretation and generation*, Potsdam, Germany.

[Haklay, 2008] Haklay, M. (2008) OpenStreetMap: User-generated street maps. *Pervasive computing IEEE*, vol. 7, issue 4, pp. 12–18.

[Haslum, 2012] Haslum, P. (2012) Narrative Planning: Compilations to Classical Planning. *Journal of Artificial Intelligence Research*, Volume 44, pages 383-395.

[Hentschel and Wagner, 2010] Hentschel, M. and Wagner, B. (2010) Autonomous robot navigation based on OpenStreetMap geodata. *Proc. 13th ITSC*, pp. 1645–1650.

[Hill et al, 2012] Hill, R., Götze, J. and Webber, B. *Final data release, Wizard-of-Oz (WoZ) experiments*. Deliverable 6.1.2, Spacebook project. http://www.spacebook-project.eu/pubs/D6.1.2.pdf

[Janarthanam et al, 2012] Janarthanam, S., Lemon, O. Liu, X., Bartie, P., Mackaness, W., Dalmas T. and Götze, J. (2012), Integrating location, visibility, and Question-Answering in a spoken dialogue system for pedestrian city exploration. *Proc. of SEMDIAL 2012*, Paris, France.

[Johansson et al, 2011] Johansson, M, Skantze, G, and Gustafson, J. (2011) Understanding Route Directions in Human-Robot Dialogue. Proc SemDial 2011: Proceedings of the 15th Workshop on the Semantics and Pragmatics of Dialogue, pages 1927.

[Jöst et al, 2005] Jöst, M., Häussler, J., Merdes, M. and Malaka, R. (2005) Multimodal interaction for pedestrians: an evaluation study. In IUI 05: Proceedings of the 10th international conference on Intelligent user interfaces, pp. 5966.

[Krug et al, 2003] Krug, K., Mountain, D. and Phan, D. (2003) Webpark: Location-based services for mobile users in protected areas. GeoInformatics, pp. 2629.

[Lemon et al, 2001] Lemon, O., Bracy, A., Gruenstein, A., and Peters, S.(2001) A Multi-Modal Dialogue System for Human-Robot Conversation , In Proc. NAACL.

[Looije et al, 2007] Looije, R., te Brake, G. and Neerincx, M. (2007) Usability engineering for mobile maps. In Proceedings of Mobility'07, 4th int. conference on mobile technology, applications, and systems.

[Lovelace et al, 1999] Lovelace, K., Hegarty, M. and Montello, D. (1999) Elements of good route descriptions in familiar and unfamiliar environments. Spatial Information Theory. Cognitive and computational foundations of geographic information science, LNCS, 1661/1991, Springer-Verlag.

[MacMahon et al, 2006] MacMahon, M., Stankiewicz, B. and Kuijpers, B. (2006) Walk the Talk: Connecting Language, Knowledge, Action in Route Instructions. National Conf on Artificial Intelligence (AAAI-06).

[Malaka and Zipf, 2000] Malaka, R. and Zipf, A. (2000) Deep map - challenging IT research in the framework of a tourist information system. Information and communication technologies in tourism, Springer.

[Maros, 2003] Maros, István (2003) *Computational techniques of the simplex method.* International Series in Operations Research & Management Science 61. Boston, MA: Kluwer Academic Publishers. pp. xx+325. ISBN 1-4020-7332-1. MR 1960274.

[Nothegger et al, 2004] Nothegger, C., Winter, S. and Raubal, M. (2004) Selection of salient features for route directions. *Spatial cognition and computation*, 4(2), pp. 113–136.

[Papadimitrou and Steiglitz, 1982] Papadimitrou, C. and Steiglitz, K. (1982) *Combinatorial optimization: Algorithms and complexity*, Prentice-Hall.

[Pritchard, 2001] Pritchard, M. (2001) Direct access quadtree lookup. In Game programming gems 2, Charles River Media publishers, pp. 394401.

[Raubal and Winter, 2002] Raubal, M. and Winter, S. (2002) Enriching Wayfinding Instructions with Local Landmarks. *Proc. GIScience '02 Proceedings of the Second International Conference on Geographic Information Science*, pp. 243-259. Springer-Verlag London, UK.

[Rehrl et al., 2011] Rehrl, K., Häusler, E. and Leitinger, S. (2010) GPS-based Voice Guidance as Navigation Support for Pedestrians, Alpine Skiers and Alpine Tourers. *Proc. workshop on multimodal location based techniques for extreme navigation*, 8th International Conference on Pervasive Computing, Helsinki, Finland, 2010.

[Ross et al., 2004] Ross, T., May, A. and Thompson, S. (2004) The use of landmarks in pedestrian navigation instructions and the effects of context. *Proc. Mobile Human-Computer Interaction - MobileHCI 2004*, Lecture Notes in Computer Science Volume 3160, 2004, pp 300-304.

[Skantze, 2007] Skantze, G. (2007). Making grounding decisions: Data-driven estimation of dialogue costs and confidence thresholds. In Proceedings of SigDial (pp. 206-210). Antwerp, Belgium.

[Skantze et al, 2006] Skantze, G., Edlund, J., and Carlson, R. (2006). Talking with Higgins: Research challenges in a spoken dialogue system. Perception and Interactive Technologies, pp. 193-196. Springer

[Sorrows et al., 1999] Sorrows, M.E. and Hirtle, S.C. (1999) The nature of landmarks for real and electronic spaces. *Spatial information theory: Cognitive and computational foundations of geographic information science*, vol. 1661 LNCS, pp. 37–50.

[Striegnitz et al, 2011] Striegnitz, K., Denis, A., Gargett, A. Garoufi, K., Koller, A. and Theune, M. (2011) Report on the Second Challenge on Generating Instructions in Virtual Environments (GIVE-2.5). In Proceedings of the Generation Challenges Session at the 13th European Workshop on Natural Language Generation (ENLG).

[Tom and Denis, 2003] Tom, A. and Denis, M. (2003) Referring to landmark or street information in route directions: What difference does it make?. Spatial Information Theory: Foundations of geographic science. LNCS 2825/2003, pp. 362-374. Springer-Verlag.

[Traum, 1999] Traum, D. (1999) Computational Models of Grounding in Collaborative Systems. AAAI Technical Report FS-99-03

[Wang et al, 2008] Wang, H., Cai, G. and MacEachren, A. (2008) GeoDialogue: A Software Agent Enabling Collaborative Dialogues between a User and a Conversational GIS. In Tools with Artificial Intelligence, 2008. ICTAI '08. 20th IEEE.

[Weerdt et al., 2005] M. De Weerdt, A. Ter Mors, and C. Witteveen.(2005) Multi-agent planning: An introduction to planning and coordination. Technical report, In: Handouts of the European Agent Summer, 2005.

[Xia et al., 2011] Xia, J., Richter, K-F., Winter, S. and Arnold, L. (2011) A survey to understand the role of landmarks for GPS navigation. *Proc. PATREC research forum*.

[Yannakakis et al., 2009] Yannakakis, G, Maragoudakis, M. and Hallam, J. (2009) Preference learning for cognitive modeling: A case study on entertainment preferences. *IEEE transactions on systems, man, and cybernetics – Part A: Systems and humans*, vol. 39, no. 6, pp. 1165–1175.

[Zipf and Jöst, 2005] A Zipf and M. Jöst. (2005) Implementing adaptive mobile GI services based on ontologies – examples for pedestrian navigation support. Computers, Environment and Urban Systems, Special Issue on LBS and UbiGIS., 2005.