Int J Software Informatics, Vol.2, No.1, July 2008, pp. 33–60 E-mail: ijsi@iscas.ac.cn
International Journal of Software and Informatics ISSN 1673-7288 http://www.ijsi.org
©2008 by Institute of Software, Chinese Academy of Sciences. All rights reserved.
Tel: +86-10-62661040

Modelling Route Instructions for Robust Human-Robot Interaction on Navigation Tasks*

Hui Shi, Bernd Krieg-Brückner

(Universität Bremen and DFKI-Lab Bremen, Germany, {bkb,shi}@informatik.uni-bremen.de)

Abstract In this paper, we demonstrate the use of qualitative spatial modelling as the foundation for the conceptual representation of route instructions, to enable robust human-robot interaction on navigation tasks. Our conceptual model is motivated by empirical studies on route navigation, and combines Qualitative Orientation Calculi for spatial reasoning using directional orientation information and topological maps for structuring route segments and routes. Moreover, we present a formal definition of the conceptual model using the algebraic specification language CASL for syntactic and semantic checking, consistency checking and verification. Finally, we introduce a generic route graph concept and its formalization. The instantiation of the generic route graph at different abstraction levels provides a formal foundation for linking the conceptual model to a global environment map used by an intelligent robot, e.g., a semi-autonomous wheelchair, to carry out human navigation tasks.

Key words: qualitative spatial calculi; route graph; navigation space; human-robot interaction; conceptual model; algebraic specification

Shi H, Krieg-Brückner B. Modelling route instructions for robust human-robot interaction on navigation tasks. *Int J Software Informatics*, **2008, 2(1): 33–60 http://www.ijsi.org/1673-7288/2/33.pdf**

1 Introduction

Route navigation instructions allow one agent to instruct another to navigate to a particular location within their shared environment. While robotic agents in the near future may have access to extremely detailed environmental descriptions through technological application (e.g., GPS, a-priori mappings), the need for an artificial agent to be capable of processing route instructions remains an interesting research question for a number of reasons. Firstly, and arguably most importantly, cooperation with naïve users in natural interaction (e.g., route instructions), rather than forced goal selection through other means (e.g., hierarchical list selection), is expected, especially for personal service robots. Furthermore, disparities between the robot's spatial representation and the ever-changing world are always possible, thus it is necessary to open up the capability of adding route instructions to goals which were not previously known to the robotic agent. Our major research aim here is natural and robust communication between humans and robots on spatial navigation tasks,

---

* This work is support by the "Deutsche Forschungsgemeinschaft" (DFG) through the Collaborative Research Center SFB/TR 8 "Spatial Cognition".

Corresponding author: Hui Shi, shi@informatik.uni-bremen.de

Manuscript received 17 Nov. 2007; revised 19 May 2008; accepted 30 May 2008; published online 30 Jul. 2008.

Copyright by Institute of Software, Chinese Academy of Sciences. All rights reserved

for example, dialogues between a driver and the Bremen semi-autonomous wheelchair Rolland[18,19].

Modelling and interpretation of route instructions have been addressed from a number of different research perspectives. In the spatial cognition and cognitive modelling communities, a great deal of effort has been applied to the analysis of route instructions as a reflection of a speaker's cognitive map (e.g., [55,13,58,16]). While such approaches are motivated by the way humans conceptualize and describe spatial information, they are often somewhat abstracted from the detail required for on-line computational analysis. On the other hand, some authors in the robotics community have attempted to process route instructions as procedural information without reference to any explicit spatial representation[20,25]. In practice however, concrete robotic systems use very detailed spatial representations and computation processes that operate at a finer level of granularity than those models proposed by the cognitive modelling community[44,6,50], but little work has been done to date on unifying such approaches with cognitive spatial representation and language processing efforts.

In Qualitative Spatial Reasoning there are two major research directions: approaches based on spatially extended objects (e.g., regions[34,7,38] or intervals[1]); and reasoning about the directions and distances of point configurations. Several empirical studies show that the critical elements of the space of navigation are landmarks and paths, links and nodes (e.g., [46,8,4,43]), where points and their connections play a more important role, thus they have been extensively used to represent and reason about various human navigation tasks. Two important examples of the point-based approach are the Cardinal Direction calculus[10] using orientation grids, and the Double-Cross calculus[13,58] based on relative orientation information. In the last ten years, several variants and extensions of these two calculi have been reported (cf. [11,21,37,58,27]).

Routes, on the other hand, are a concept commonly used when dealing with human spatial navigation activities (cf. [56,15]). A route from one place to another consists of a sequence of decision points (places), which are identified by salient landmarks. Instead of providing a particular way for making decisions, the route graph concept focuses on the integration of routes into a graph-like structure and on the acquisition of new route knowledge. Thus, modelling human navigation knowledge requires the representation and reasoning of humans' qualitative orientation activities to make decisions, and of their topological knowledge about places, route segments and routes. In this paper, we present a model, which combines qualitative spatial orientation calculi (e.g., the Double-Cross calculus[13,58]) with route graphs (e.g., the *Route Graph*[56,15]).

Quantitative and qualitative levels. Following the main idea of Kuipers' Spatial Semantic Hierarchy (SSH)[17], that navigation space consists of multiple interacting representations, both qualitative and quantitative, besides the qualitative model of humans' spatial navigation knowledge, a global quantitative (or metric) map is useful for the robot to carry out spatial activities (e.g., moving from one place to another, or reorientating itself). However, a quantitative representation with numerous metric data is in general less efficient for planning and reasoning and can be expensive and error-prone to create. Therefore, efficient and robust interaction requires both

Copyright by Institute of Software, Chinese Academy of Sciences. All rights reserved

levels of representation of spatial knowledge, and a well-defined mapping concept. In our wheelchair scenario, a Voronoi graph based topological map with metric information is used[15,42], and a set of fuzzy functions is defined to interpret human route instructions. Krieg-Brückner *et al.* in [15] developed a unified concept for defining route graphs at a variety of abstraction levels, thus providing a formal framework to link the conceptual model to the global quantitative map.

Formalizing the conceptual model of human navigation knowledge is an additional objective of this research. We hypothesize that the spatial concepts in the conceptual model can be defined as a set of axioms that are formalized as algebraic specifications. In this paper, the conceptual model, including relevant types, operations and predicates, will be specified using the formal specification language CASL[2,31,32]. With the help of some software tools, it then becomes possible to check the consistency and verify the properties of the conceptual model.

Structure of the paper. We begin in Section 2 by reviewing empirical studies involving the presentation of route instructions to artificial communicating partners. After introducing the route graph model and the Double-Cross Calculus in Section 3, we present our conceptual model in Section 4, which combines the route graph model and the Double-Cross Calculus to represent and reason about human route instructions. The formalization of the conceptual model in the algebraic specification language CASL is then given in Section 5. In Section 6 we introduce a generic concept of route graphs and its formalization in CASL. Based on the generic route graph concept we then show how to link the conceptual model to a quantitative map. We compare our research with related work in Section 7, and conclude in Section 8.

2 Empirical Studies on Route Navigation

In the last decade, a number of research initiatives on communication between people and intelligent robots using natural language have been reported. Among them are several studies that concern navigation tasks (e.g., [20,4,43]). The Instruction Based Learning (IBL) project[20,4], for example, concentrated on navigation tasks in a miniature outdoor environment, with emphasis on the robot's route learning by human verbal instructions. In this section, we analyse a corpus collected in our research center, which is used in the following sections to investigate the qualitative modelling of route instructions. After introducing the corpus, we will discuss some empirical results in detail.

2.1 Telling Rolland Where to Go

The corpus discussed here was obtained in an empirical study to investigate natural language in communicating with robots about spatial concepts, such as route navigation and spatial relations between objects[47,9,43]. 23 German and 6 English native speakers took part in the study.

The experiment participants were seated on the wheelchair Rolland and asked to move around in a university building that they know well, and explain important places (landmarks) to Rolland. The participants believed that Rolland understood what they said and completed its internal map using the information they gave. In the following experimental phase, the participants were asked to instruct Rolland to navigate to a specific place, where they had been previously, for example, the "Stuga-

www.ijsi.org

Copyright by Institute of Software, Chinese Academy of Sciences. All rights reserved

Raum" (student-union room). In this phase the participants produced *in-advance* route instructions with "Stuga-Raum" as the goal, i.e., the participants gave complete route descriptions before the trip, instead of incremental ones. The information given to the wheelchair in the previous phase was applicable in this phase. At this point, a number of communication problems occurred because of mismatches between the robot's knowledge and the users' route descriptions. Thus, an adequate representation of the route descriptions is necessary to enable an effective interaction between them. The route descriptions produced in the second phase will be used here to discuss the conceptual modelling of human route descriptions. Figure 1 shows a sample route description with a descriptive translation.

1. dann äh, muss ich mich jetzt umdrehen
   (*I must turn around*)
2. aus der Tür fahren
   (*drive out of the door*)
3. äh, dann nach rechts mich drehen
   (*then turn to the right*)
4. dann, - ziemlich lange geradeaus
   (*then straight ahead for a rather long time*)
5. äh, vorbei an den Fahrstühlen
   (*pass by the lifts*)
6. vorbei an den Haupttreppen
   (*pass by the stairs*)
7. ähm, und dann, – sind wir eigentlich schon am Stuga-Raum
   (*and then we should be at the student-union room*)

Figure 1. A sample route description from the Rolland corpus

2.2 Some Empirical Evidence

The corpus introduced above confirms that a route description given by a human to an intelligent robot is similar to the way in which people instruct other people how to navigate (cf. [55,8]), which consists of a sequence of decisions with reorientations and landmarks; for example, "door", "stairs" and "room" are used either as decision points or as reference points for orientation. At a first glance, expressions like "on the left" and "pass by the lifts" describe some relations between a path and a region. However, in route descriptions such regions can usually be treated as points, since their spatial extents have no influence on the route, especially in corridor scenarios (e.g. the Rolland corpus), where most landmarks are geometrically simple and regular. In the Rolland corpus, most landmarks referred to (90% in the Rolland corpus) can be treated as points. The typical expressions referring to a landmark as a region are, for example, "dem Korridor folgen" (*follow the corridor*) or "den Gang hoch" (*up the corridor*), in which the spatial extent of the corridor delimits the described route. In outdoor settings, such as the IBL corpus, more exceptions can be found, such as "keep going forward until you hit the end of the train station", or "then follow it round and you should be right outside Boots".

A route instruction may contain one or more simple ones by reference, since it is natural that people use existing route knowledge when describing a new one, instead of re-describing all known routes in detail. In the IBL project[4], each subject gave six route instructions to the robot, among which three were "short" and three "long". A long route may include a reference to the destination of a short one, thus typically contain such utterances as “once you pass the car park”, in which “the car park” is the destination of a route described before, such that users could avoid re-describing the route to “the car park”. The use of existing route knowledge to describe new routes shows that human route knowledge is well structured, instead of a collection of isolated routes. We discuss the integration of routes into a route graph in Section5.4.

Turns and *moves* are two operations prevalent in route descriptions. Instead of metric angle and distance, people frequently give qualitative information about the angle of a turn (e.g., “turn around”, “turn right” and “slightly left”), or the distance of a travel, for example, “erst mal rechts um bis zum Ausgang” (*first turn right [and drive] until the exit*). Quantitative information in human route descriptions is usually inaccurate and of little help, such as “ungefähr zwanzig bis fünfundzwanzig Meter” (*about twenty to twenty-five meters*). Of 247 utterances in the Rolland corpus, only 17 contain metric information; such quantitative data are sometimes redundant as in “ungefähr 20 bis 30 Sekunden, dann am Ende des Gangs” (*about 20 to 30 seconds, then at the end of the corridor*).

Most route instructions found in the Rolland corpus are *well-structured*, i.e., they contain a temporally ordered sequence of qualitatively described route segments with movements, (re)orientations and spatial references to landmarks. However, we found the occurrences of quantitative instructions like “neunzig Grad nach rechts” (*turn ninety degrees right*), constraint information like “Stugaraum ist da wo wir eben vorbei gefahren sind” (*Student-union room is where we just passed by*), as well as temporal disorder like “dann auf dem Flur nach links fahren bis zur ersten möglichen Abzweigung nach links, also da kommt dann noch dieser Turm...” (*then drive on the corridor to the left until the first possible turn off to the left, I mean, there is also this tower...*). Among the 28 successfully transcribed trails, 21 trails contain a well-structured route instruction. Three route instructions consist exclusively of drive commands, such as “turn left” and “go straight”, without any reference to landmarks, and four have only one task relevant utterance like “go to the student-union room”. In this paper, we focus on well-structured route instructions in which the spatial extents of landmarks do not influence the routes.

3 Orientation Calculi and Route Graphs

Motivated by the empirical studies, we chose Freksa’s qualitative spatial calculus using relative orientation information[12,13,58], i.e., the *Double-Cross* calculus, to represent and to reason about human spatial orientation behavior. Moreover, the *Route Graph* model[15] will be taken to represent the topological structure of the navigation space, such that structural relations between route segments and routes can be inferred.

3.1 Route Graph Model

Route Graphs have been introduced as a general model of environment for navigation by various agents in a variety of scenarios[56,15,22]. They can be used as quantitative maps with sensory input and additional metrical computation models to control the navigation of robotic agents in the environment, or at the cognitive level to abstractly model humans’ topological knowledge while they act in space. Perhaps the most important property of route graphs is that different routes can be integrated

Copyright by Institute of Software, Chinese Academy of Sciences. All rights reserved

into a graph structure, in which the information concerning these routes is composed.

Route graphs are a special class of graphs. A *node* of a route graph, called a *place*, has a particular position and has its own "reference system"; it may, but need not, be rooted in a (more) global reference system, such as a 2-D geometric system. An *edge* of a route graph, called *route segment*, is directed from a source place to a target place, and always has three attributes: an *entry*, a *course* and an *exit*. Exactly what information is associated with these attributes is specially defined for each route graph instantiation. For example, an entry or an exit in a route graph at the quantitative level can be an angle, measured in degrees, with respect to a global 2-D geometric system, the course is then characterized by metrical data for length and width; while an entry/exit at the conceptual level may contain qualitative orientation information (e.g., to the left/right), the course is then the path between two reorientations. For more information about route graphs see [56,15,16].

3.2 Qualitative Spatial Reasoning with Orientations

Freksa put forward the *Double-Cross* calculus for qualitative spatial representation and reasoning using orientation information[12,13,58] for external qualitative spatial knowledge. In this calculus, the concept *orientation frame* is introduced to represent qualitative orientation information. The frame is aligned to the orientation determined by two points in 2-dimensional space, the start point and the end point of a movement. Combining the front/back and the left/right dichotomy, the Double-Cross calculus may distinguish 15 meaningful disjoint orientation relations, as illustrated in Fig.2, where we assign names to the orientation relations.

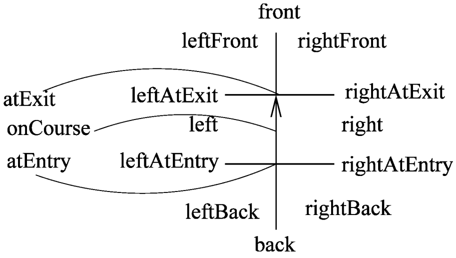


Figure 2. Orientation frame with 15 relations for the Double-Cross calculus

The directed line between two points gives rise to the orientations *front* and *back* on this line, also *leftAtExit*, *rightAtExit*, *leftAtEntry*, and *rightAtEntry*, whose terminology we have borrowed from *Route Graph*s; *left* and *right* correspond to two areas besides the line, similarly *leftFront*, *rightFront*, *leftBack*, and *rightBack*; finally, the special positions *atEntry*, *atExit*, and the property to be on the line, *onCourse*, are introduced.

The composition table for reasoning with orientation relations in the Double-Cross calculus can be found in [13].

4 A Conceptual Model for Route Navigation

Motivated by the empirical evidence (see Section 2.2), we are now going to develop a spatial model, called *conceptual model*, to represent and reason about human route navigation knowledge, which combines the Double-Cross calculus and the Route

Graph. Its structure and operational properties are then formally specified in Section 5.

The *conceptual model* is a triple of types, functions and predicates. The set of types consists of:

- $\mathcal{O}$ for orientations, here we use 15 disjoint orientations defined by the Double-Cross calculus (Fig.2), denoted as o, o', o_1, etc.
- $\mathcal{M}$ for landmarks, denoted as m, m', m_1, etc.
- $\mathcal{L}$ for locations, denoted as l, l', l_1, etc.
- $\mathcal{V}$ for vectors from a source location to a target location, $\mathbf{ab}$ denotes the vector from location a to b.
- $\mathcal{S}$ for route segments, a subset of $\mathcal{V}$, denoted as s, s', s_1, etc. The source and target of a route segment are reachable locations.

The most important functions are *oEntry*, *oExit* and *oCourse*, which take a route segment and define the entry, the exit (as orientations) and the course of the route segment, respectively. An additional function *at* defines the location of a given landmark, thus $at(m) = l$ means that landmark m is at location l.

The elementary relation of the *conceptual model* is *ori*, which defines the orientation of a location with respect to a vector: $ori(\mathbf{bc}, l, rightFront)$, for example, means l is on the right front of $\mathbf{bc}$. The orientation of some location with respect to a vector is possibly undefined; the orientation relation *ori* and the following auxiliary relations are used to check the consistency of route instructions (except for the representation of orientation information in route instructions), thus they are defined by predicates instead of by functions.

Based on *ori*, a set of auxiliary relations can be defined, for example,

on :	$\mathcal{V} \times \mathcal{L}$	a *location* is on an *vector*
left_of :	$\mathcal{V} \times \mathcal{L}$	a *location* is on the left of an *vector*
right_of :	$\mathcal{V} \times \mathcal{L}$	a *location* is on the right of an *vector*

The following axioms assert that these relations can be defined using the elementary relation *ori*.

$$on(\mathbf{ab}, l) \Leftrightarrow ori(\mathbf{ab}, l, onCourse) \tag{1}$$

$$left\_of(\mathbf{ab}, l) \Leftrightarrow ori(\mathbf{ab}, l, left) \tag{2}$$

$$right\_of(\mathbf{ab}, l) \Leftrightarrow ori(\mathbf{ab}, l, right) \tag{3}$$

4.1 Qualitative Route Representations

As discussed in Section 2, operations used in the route descriptions include moves and turns. In addition, users often use spatial relations in their route descriptions. For example, "the main stairs are on the left" just states the spatial relation between the "main stairs" and the current *position* (i.e., the current place and direction); "through the door" includes a movement and a relation: after the movement the door

Copyright by Institute of Software, Chinese Academy of Sciences. All rights reserved

is behind. Thus, we define a set of relations to describe the operations used by people to describe routes.

$$\mathit{via}(\mathbf{ab}, m) \Leftrightarrow \exists l.\ \mathit{at}(m) = l \wedge \mathit{on}(\mathbf{ab}, l) \tag{4}$$

$$\mathit{passOn}(\mathbf{ab}, m, \mathit{left}) \Leftrightarrow \exists l.\ \mathit{at}(m) = l \wedge \mathit{left\_of}(\mathbf{ab}, l) \tag{5}$$

$$\mathit{passOn}(\mathbf{ab}, m, \mathit{right}) \Leftrightarrow \exists l.\ \mathit{at}(m) = l \wedge \mathit{right\_of}(\mathbf{ab}, l) \tag{6}$$

$$\mathit{passBy}(\mathbf{ab}, m) \Leftrightarrow \exists l.\ \mathit{at}(m) = l \wedge (\mathit{passOn}(\mathbf{ab}, l, \mathit{left}) \vee \mathit{passOn}(\mathbf{ab}, l, \mathit{right})) \tag{7}$$

To illustrate, let us consider the sample route description in Fig.1. The resulting qualitative route representation is shown in Fig.3, where all variables (s_0, s_1, s_2 and l) are existentially quantified. The corresponding utterance parts are also given. The sample route description is represented as a route with three route segments. The representation of the first utterance takes the start position as the reference direction for the first route segment s_0. The representation of the fourth utterance "*straight ahead for a rather long time*" is empty, because it does not make any contribution to the qualitative route. Movements with references to landmarks such as "through", "pass by" are all represented as spatial relations.

1	$\mathit{oEntry}(s_0) = \mathit{back}$	*turn around*
2	$\mathit{via}(\mathit{oCourse}(s_0), \text{door})$	*drive out of the door*
3	$\mathit{oEntry}(s_1) = \mathit{right}$	*turn to the right*
4		*staight ahead for a long time*
5	$\mathit{passBy}(\mathit{Course}(s_1), \text{lifts})$	*pass by the lifts*
6	$\mathit{oEntry}(s_2) = \mathit{front}$, $\mathit{passBy}(\mathit{oCourse}(s_2), \text{mainStairs})$	*pass by the stairs*
7	$\mathit{at}(\text{Stuga-Raum}) = l$, $\mathit{ori}(\mathit{oCourse}(s_2), l, \mathit{front})$	*at the student-union room*

Figure 3. The representation of the sample route description

As defined, a route is a sequence of route segments. Suppose $\langle s_0, s_1, \cdots, s_n\rangle$ is a route, then the following constraints should be satisfied, where $0 \leqslant i \leqslant n$ and $0 \leqslant j \leqslant n-1$:

- $\mathit{oEntry}(s_i),\ \mathit{oExit}(s_i) \in \mathcal{O}$
- $\exists \mathbf{l_1 l_2} \in \mathcal{V}.\ \mathit{oCourse}(s_i) = \mathbf{l_1 l_2}$
- $\mathit{oCourse}(s_j) = \mathbf{l_1 l_2} \wedge \mathit{oCourse}(s_{j+1}) = \mathbf{l_3 l_4} \Rightarrow l_2 = l_3$

To integrate two routes into a route graph, one has to come up with a common location in the environment, and all related entries and exits on one route at that location must be recomputed to conform with the other route at that location. Furthermore, the consistency of the spatial relations related to the common location should be proved according to the Double-Cross calculus[13]. In Section 5.4, we will discuss this issue formally.

4.2 Reasoning About Qualitative Route Representations

Now we can reason about qualitative route descriptions against a representation of the environment. A *conceptual environment* might consist of familiar landmarks, locations, routes, and spatial relations. To illustrate, let us return again to the empirical study described in Section 2.1, where a user at first describes the environment to the wheelchair in a joint attention task, before a route description to a destination is provided. Following one description where the user made a number of annotations while driving to the Stuga-Raum (student-union room), the conceptual environment would include the landmarks “the door”, “the lifts” and “the main stairs”, and the following equations and relations:

$$\{\ \ at(\text{door}) = l_1, at(\text{lifts}) = l_2, at(\text{mainStairs}) = l_3, \\ on(\mathbf{ab}, l_1), ori(\mathbf{ab}, c, rightAtExit), ori(\mathbf{bc}, l_3, rightFront), ori(\mathbf{bc}, d, front)\ \}$$

Thus, we can see that “the door”, “the lifts” and “the main stairs” are located at l_1, l_2 and l_3, respectively. l_1 is *on* the route segment **ab**. Moreover, c is *rightAtExit* w.r.t. **ab**, l_3 is *rightFront* w.r.t. **bc**, and d in front of **bc**. But we do not know, for example, the answer to the question “where is the Stuga-Raum”.

Before the conceptual representation of a route description can be integrated with the representation of the environment, its consistency with the environment should be proved, which means that the spatial relations in the route representation should be consistent with those already present in the conceptual environment. Furthermore, the newly introduced conditions may be simplified according to the environment, if possible. Taking the above example, and supposing $oCourse(s_0) = \mathbf{ab}$, $oCourse(s_1) = \mathbf{bc}$ and $oCourse(s_2) = \mathbf{cd}$; then, from the relations $ori(\mathbf{bc}, d, front)$ and $ori(\mathbf{bc}, l_3, rightFront)$ in the environment, we can conclude, according to the Double-Cross calculus, that

$$ori(\mathbf{cd}, l_3, rightFront) \vee ori(\mathbf{cd}, l_3, right) \vee ori(\mathbf{cd}, l_3, rightAtExit)$$

Thus, the main stairs (at the location l_3) can only be passed on the right in this case, and the alternation $ori(\mathbf{cd}, l_3, left)$ in the route representation is therefore inconsistent. After checking the consistency and simplifying the spatial relations, the route $\langle s_0, s_1, s_2\rangle$ will be added to the conceptual environment, together with a new equation $at(\text{Stuga-Raum}) = l$ and the following three new relations: $left\_of(\mathbf{bc}, l_2) \vee right\_of(\mathbf{bc}, l_2)$, $right\_of(\mathbf{cd}, l_3)$, and $ori(\mathbf{cd}, l, front)$. Figure 4 shows the updated conceptual environment.

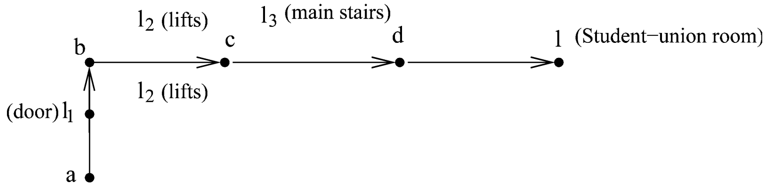


Figure 4. A sample conceptual environment

5 Formalizing the Conceptual Model in CASL

In this section we present an approach to specify the conceptual model using the Common Algebraic Specification Language, CASL[2,31,32]. CASL is supported by tools in the Heterogeneous Tool Set, HETS[29,30], for syntactic and semantic checking, consistency checking, verification, and deduction. Initial ideas for the formalization of human route descriptions using CASL are discussed in [16]. A CASL specification basically includes declarations, to introduce components of signatures, such as sorts to introduce types, operations or predicates, as well as axioms written in first-order logic, to give properties of those structures. The Casl syntactic constructors are explained in separate sections where they are first used, such that the reader without any previous Casl knowledge can understand the intuitive meaning of the specifications. Moreover, all the specifications presented in this section have passed the syntactic, semantic and consistency checking by Hets.

We first introduce relative orientations with eight directions, and their algebraic properties. Then, we define locations, vectors, and orientations between them as a separate theory, and finally use both to define the Double-Cross calculus. With this stepwise approach, each algebraic specification can be considered separately, the properties of each can be generalised beyond the particular application, and potentially each can be used in a variety of other application contexts.

5.1 Ego Orientation Calculus

While the Cardinal Direction calculus (cf. [10,21]) and the Star calculus[37] are based on a global reference frame, the ego orientation calculus uses local reference frame associated to each point. Figure 5 shows an ego orientation distinguishing eight directions: the thick arrow denotes the ego orientation as a basis, and the fine arrow the orientation of some other object, in relation to the ego orientation. Depending on the application, we may want to distinguish 2, 4, and 8 orientations (or more); the more orientations we consider, the finer our statements become, but we generate more computational complexity in this way. In this paper we follow Freksa[13], who claims, based on psychological studies, that eight orientations are cognitively adequate.

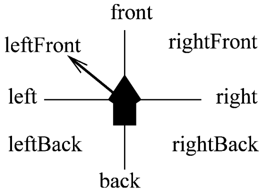


Figure 5. Eight ego orientations

The Casl specification in Fig.6 is structured as follows: First, two orientations, *front* and *back* with respect to the ego orientation, are introduced, and two operations, inverse ($\sim o$) and complement ($\overline{o}$); moreover, plus ($+$) and minus ($-$), denoting addition and subtraction of orientations. $+$ is associative, commutative and has $front$ as a unit element ($a + front = a$), thus ($Orientation, front, +$) constitutes a commutative monoid; the other algebraic properties are given as universally quantified axioms. These properties hold generally for all ego orientation calculi, no matter how fine the orientations are.

Note that a datatype declared as “free” (i.e., **free type**) introduces a sort (e.g., *Orientation2*) with constructors (e.g., *front* and *back*) as alternatives on the right-hand side of “::=”, such that terms with distinct constructors have different values, and the sort is generated by its constructors. With **sort** or **sorts** one or more types can be declared, possibly as subsorts of some other sort (using the symbol “<”). The two underscores “__” indicate a placeholder for an argument of operations, such as “+” or “∼”. A text after “%%” is a comment.

spec EGOORIENTATION =
  free type *Orientation2* ::= *front* | *back*
  sort *Orientation2* < *Orientation*
  ops $\sim\_\_ : \mathit{Orientation} \rightarrow \mathit{Orientation}$; %% inverse / converse
    $\overline{\_\_} : \mathit{Orientation} \rightarrow \mathit{Orientation}$; %% full complement
    $\_\_+\_\_ : \mathit{Orientation} \times \mathit{Orientation} \rightarrow \mathit{Orientation}$,
      assoc, *comm*, *unit front*;
    $\_\_-\_\_ : \mathit{Orientation} \times \mathit{Orientation} \rightarrow \mathit{Orientation}$
  $\forall\ a, b : \mathit{Orientation}$
  • $\overline{\overline{a}} = a$ • $a + \overline{a} = \mathit{front}$
  • $\sim a = \mathit{back} + a$ • $\sim \sim a = a$ • $a - b = a + \overline{b}$
then %implies
  $\forall\ a : \mathit{Orientation}$
  • $\overline{\mathit{front}} = \mathit{front}$ • $\overline{\mathit{back}} = \mathit{back}$ • $\sim \mathit{front} = \mathit{back}$
  • $\mathit{back} + \mathit{back} = \mathit{front}$ • $\overline{a} = \mathit{front} - a$
then
  free type *Orientation4* ::= *sort Orientation2* | *right* | *left*
  sort *Orientation4* < *Orientation*
  • $\mathit{left} = \sim \mathit{right}$ • $\mathit{left} = \overline{\mathit{right}}$
then %implies
  • $\mathit{right} + \mathit{right} = \mathit{back}$ • $\mathit{left} + \mathit{right} = \mathit{front}$
then
  free type *Orientation8* ::= *sort Orientation4* |
    rightFront | *rightBack* | *leftFront* | *leftBack*
  sort *Orientation8* < *Orientation*
  • $\mathit{rightFront} + \mathit{rightFront} = \mathit{right}$
  • $\mathit{leftFront} = \overline{\mathit{rightFront}}$ • $\mathit{leftBack} = \overline{\mathit{rightBack}}$
  • $\mathit{rightFront} = \sim \mathit{leftBack}$ • $\mathit{leftFront} = \sim \mathit{rightBack}$
then %implies
  • $\mathit{rightFront} = \overline{\mathit{leftFront}}$ • $\mathit{rightBack} = \overline{\mathit{leftBack}}$
end

Figure 6. Specification of ego orientation with 2, 4, and 8 orientations

Now the sort *Orientation2* is extended by two other orientations, *left* and *right*, and their properties; subsequently with the orientations *leftFront*, *rightFront*, *leftBack*, and *rightBack*, respectively; thus we arrive at eight orientations. The structuring operator **then** introduces a new section of the specification that extends the previous part. In the case of **then %implies**, the following formulae are not axioms, but meant to be deducible from the axioms; they constitute a proof obligation. For instance, $\sim \mathit{front} = \mathit{back}$ is deducible from $\sim a = \mathit{back} + a$ and *front* is the unit of $+$; $\mathit{left} + \mathit{right} = \mathit{front}$ deducible from $\mathit{left} = \overline{\mathit{right}}$, $a+\overline{a}=\mathit{front}$ and $+$ is commutative; and $\mathit{rightFront} = \overline{\mathit{leftFront}}$ deducible from $\mathit{leftFront}=\overline{\mathit{rightFront}}$ and $\overline{\overline{a}} = a$.

Other authors have introduced a yet finer splitting and qualitative distances (e.g., [37,27]); such additions should be extensions following the pattern introduced here.

5.2 Locations and Vectors

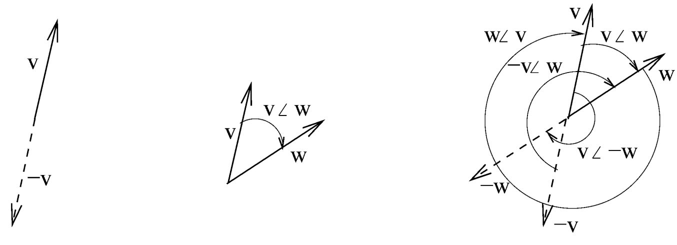
The definition of locations extends the pre-defined specification IDENTITIES (just introducing a sort *Id* for simplicity here) with a new sort *Location* and an operator *id* associating each location with an identifier (the second specification in Fig.7). As an extension of EGOORIENTATION and LOCATIONS, the specification VECTORS introduces the sort *Vector* that denotes directed connections between two locations; note that these vectors do not reside in a global reference frames.

```
spec IDENTITIES =
      sort  Id
end

spec LOCATIONS = IDENTITIES then
      sort  Location
      op    id : Location → Id
end

spec VECTORS = EGOORIENTATION and LOCATIONS then
      sort  Vector
      ops   __⟶ __ : Location × Location →? Vector
            source, target : Vector → Location
      • def x ⟶ y ⇔ ¬ x = y                                  %% definedness
      ops   −__ :  Vector → Vector;  %% inverse
            __∠__ : Vector × Vector → Orientation
      ∀ v, w : Vector
      • v ∠ w = ∼ (v ∠ − w)          • v ∠ − w = − v ∠ w
      • v ∠ w = ‾(w∠v)‾              • − − v = v
then %implies
      ∀ v, w : Vector
      • v ∠ w = − v ∠ − w            • − v ∠ w = ∼ (v ∠ w)
end
```

Figure 7. Specification of locations, vectors, and orientations between them

Each vector has a source and a target location, defined by two functions *source* and *target*. Vectors are constructed by the operation "$\longrightarrow$ ". Note that "$\rightarrow ?$" is a partial constructor operation in $Location \times Location \rightarrow ?\ Vector$; the additional axiom *definedness* states that the vector is defined, only if the operation $\longrightarrow$ is applied to two different locations.

The operation "$-$" inverts the direction of a vector; the operation "$\angle$" yields an orientation between two vectors: $v\angle w$ is called the orientation of w with respect to v. The algebraic properties nicely relate the orientation relation between vectors ($\angle$), vector inversion ($-$), and the orientation operations inverse ($\sim o$) and complement ($\overline{o}$). Figure 8(a) shows a vector and its inverse; Figure 8(b) the orientation of a vector with respect to another one, in the example, w is on the *rightFront* of v; and Fig.8(c) some orientations between vectors that may help to understand the axioms in the specification. Taking the first axiom as an example, we can easily conclude that $v\angle - w$ is an inverse of $v\angle w$, i.e., if we add *back* to $v\angle w$, it equals to $v\angle - w$. Finally, the last two formulae can be derived from the above axioms directly.



(a) (b) (c)

Figure 8. Specification of places, route segments, and place integration: (a) a vector and its inverse; (b) the orientation of a vector with respect to another one; (c) some orientations between two vectors

5.3 The Double-Cross Calculus

We now want to define the Double-Cross calculus in terms of the Ego Orientation calculus. Note that we defined a variant of the Double-Cross calculus. The orientation o of a point c with respect to a vector $\mathbf{ab}$ (i.e., $\mathbf{ab} : c = o$ in the Double-Cross calculus) is denoted as the orientation relation between the two vectors $\mathbf{ab}$ and $\mathbf{bc}$ (i.e., $a \longrightarrow b \angle b \longrightarrow c \rhd o$). Additionally, we can define the orientation between any two vectors if the source or target of one vector equals either the source or the target of another one, specified by the axiom (O0) in Fig.10.

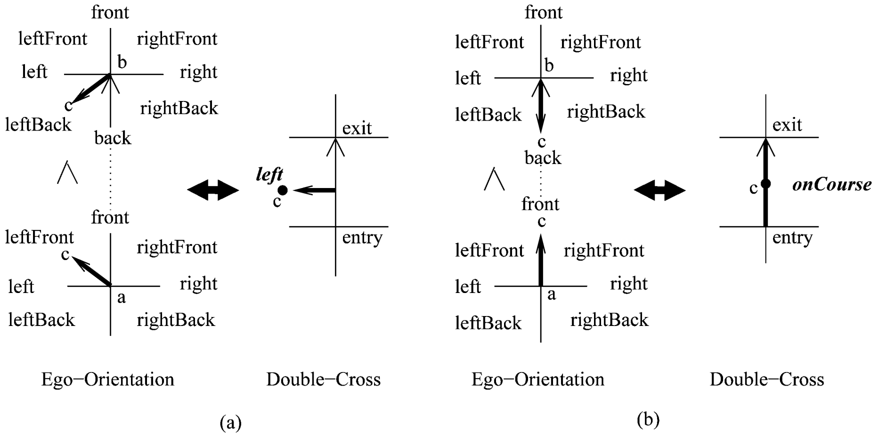


Figure 9. Relation between two Ego and corresponding Double-Cross orientations: (a) the definition of *left* as the conjunction of *leftBack* and *leftFront*; (b) *oCourse* as the conjunction of *back* and *front*

We start with the Double-Cross calculus with only eight orientation relations. Figure 9 shows the principle of defining each relation of the Double-Cross calculus as a conjunction of two relations in the Ego Orientation calculus (cf. Fig.5), a pattern that will repeat in the specification: the basic vector of the Double-Cross calculus on the right-side combines two ego orientation frames on the left-side, corresponding to the orientations at the entry and the exit, by the logical conjunction $\wedge$. Example (a) in Fig.9 shows the definition of *left*; the other 7 relations follow the same pattern, cf. the axiom (O8) in the specification (Fig.10). The pattern is still the same when we extend the Double-Cross with relations at the entry and exit, respectively, yielding thirteen relations. For the predicate *onCourse* we also need two ego orientation frames in the same pattern (Fig.9(b)), cf. the axiom (O13) in the specification (Fig.10).

Now we turn to the remaining relations *atExit* and *atEntry* of the Double-Cross calculus. Their definitions deviate from the standard pattern (see the axiom (O15) in Fig.10). Since the location c is located at *entry* or *exit*, the vectors $entry \longrightarrow c$ and $exit \longrightarrow c$ are undefined.

Note that this extended variant of the Double-Cross calculus enjoys the additional algebraic properties inherited from the ego calculus and relations between vectors. In particular, inversion and complement is defined naturally; also, vectors may be inverted.

5.4 Route Graphs and Integration

In Sections 5.1 and 5.2, we defined several sorts, such as *Orientation*, *Location* and *Vector*, and their operations to specify various orientation calculi. To introduce

*Route Graph*s we are interested in special kinds of locations and vectors to define concepts like places and route segments. A *Place* is a (uniquely identifiable) location and has its own local reference system, as specified in Fig.11. The operation *oOrigin* defines the inherent reference direction as a vector at a place, i.e., the local reference frame (or *origin*) at that place. The origin may also be used to ground a place in a global reference system.

spec DoubleCrossCalculus = Vectors
then
sorts $\mathit{Orientation8} < \mathit{OrientationDCC13}$;
$\mathit{OrientationDCC13} < \mathit{OrientationDCC}$
$\mathit{OrientationDCC} < \mathit{Orientation}$
pred $\_\_\angle\_\_\rhd\_\_ : \mathit{Vector} \times \mathit{Vector} \times \mathit{OrientationDCC}$
$\forall\ a, b, c, d : \mathit{Location};\ o : \mathit{OrientationDCC}$
$\bullet\ a \longrightarrow b \angle c \longrightarrow d \rhd o \Rightarrow a = c \vee b = c \vee a = d \vee b = d$ %% (O0)
$\forall\ \mathit{entry}, \mathit{exit}, c : \mathit{Location};\ u, v, w : \mathit{Vector}$
$\bullet\ v = \mathit{entry} \longrightarrow \mathit{exit} \wedge w = \mathit{exit} \longrightarrow c \wedge u = \mathit{entry} \longrightarrow c \Rightarrow$
$(v \angle w \rhd \mathit{leftFront} \Leftrightarrow v \angle w = \mathit{leftFront} \wedge v \angle u = \mathit{leftFront}) \wedge$
$(v \angle w \rhd \mathit{left} \Leftrightarrow v \angle w = \mathit{leftBack} \wedge v \angle u = \mathit{leftFront}) \wedge$
$(v \angle w \rhd \mathit{leftBack} \Leftrightarrow v \angle w = \mathit{leftBack} \wedge v \angle u = \mathit{leftBack}) \wedge$
$(v \angle w \rhd \mathit{front} \Leftrightarrow v \angle w = \mathit{front} \wedge v \angle u = \mathit{front}) \wedge$
$(v \angle w \rhd \mathit{back} \Leftrightarrow v \angle w = \mathit{back} \wedge v \angle u = \mathit{back}) \wedge$
$(v \angle w \rhd \mathit{rightFront} \Leftrightarrow v \angle w = \mathit{rightFront} \wedge v \angle u = \mathit{rightFront}) \wedge$
$(v \angle w \rhd \mathit{right} \Leftrightarrow v \angle w = \mathit{rightBack} \wedge v \angle u = \mathit{rightFront}) \wedge$
$(v \angle w \rhd \mathit{rightBack} \Leftrightarrow v \angle w = \mathit{rightBack} \wedge v \angle u = \mathit{rightBack})$ %% (O8)
then
free type $\mathit{OrientationDCC13} ::= \mathit{sort}\ \mathit{Orientation8}\ |$
$\mathit{leftAtEntry} \mid \mathit{rightAtEntry} \mid \mathit{leftAtExit} \mid \mathit{rightAtExit} \mid \mathit{onCourse}$
$\bullet\ \mathit{leftAtExit} = \sim \mathit{rightAtEntry}$ $\bullet\ \mathit{leftAtEntry} = \sim \mathit{rightAtExit}$
$\bullet\ \mathit{leftAtExit} = \overline{\mathit{rightAtExit}}$ $\bullet\ \mathit{leftAtEntry} = \overline{\mathit{rightAtEntry}}$
$\bullet\ \mathit{onCourse} = \overline{\mathit{onCourse}}$
$\forall\ \mathit{entry}, \mathit{exit}, c : \mathit{Location};\ u, v, w : \mathit{Vector}$
$\bullet\ v = \mathit{entry} \longrightarrow \mathit{exit} \wedge w = \mathit{exit} \longrightarrow c \wedge u = \mathit{entry} \longrightarrow c \Rightarrow$
$(v \angle w \rhd \mathit{leftAtExit} \Leftrightarrow v \angle w = \mathit{left} \wedge v \angle u = \mathit{leftFront}) \wedge$
$(v \angle w \rhd \mathit{leftAtEntry} \Leftrightarrow v \angle w = \mathit{leftBack} \wedge v \angle u = \mathit{left}) \wedge$
$(v \angle w \rhd \mathit{rightAtExit} \Leftrightarrow v \angle w = \mathit{right} \wedge v \angle u = \mathit{rightFront}) \wedge$
$(v \angle w \rhd \mathit{rightAtEntry} \Leftrightarrow v \angle w = \mathit{rightBack} \wedge v \angle u = \mathit{right}) \wedge$
$(v \angle w \rhd \mathit{onCourse} \Leftrightarrow v \angle w = \mathit{back} \wedge v \angle u = \mathit{front})$ %% (O13)
then
free type $\mathit{OrientationDCC} ::=$
$\mathit{sort}\ \mathit{OrientationDCC13} \mid \mathit{atEntry} \mid \mathit{atExit}$
$\bullet\ \mathit{atExit} = \sim \mathit{atEntry}$ $\bullet\ \mathit{atExit} = \overline{\mathit{atExit}}$ $\bullet\ \mathit{atEntry} = \overline{\mathit{atEntry}}$
$\forall\ \mathit{entry}, \mathit{exit} : \mathit{Location}$
$\bullet\ \mathit{entry} = \mathit{exit} \vee$
$(\mathit{entry} \longrightarrow \mathit{exit} \angle \mathit{entry} \longrightarrow \mathit{exit} \rhd \mathit{atExit} \wedge$
$\mathit{entry} \longrightarrow \mathit{exit} \angle \mathit{exit} \longrightarrow \mathit{entry} \rhd \mathit{atEntry})$ %% (O15)
end

Figure 10. The Double-Cross calculus with 8, 13, and 15 orientations

A route segment of the sort *Segment* is a vector from a source place to a target place, thus it is in fact a subsort of *Vector*, such that the source and target of a segment belong to the sort *Place*. A segment always has an entry, an exit (place) and a course defined by the operators *oEntry*, *oExit* and *oCourse*, respectively. In the context of route descriptions here, an entry or an exit is in fact represented as an orientation with respect to a given origin at this place. The operation *oEntry* returns the orientation of a segment with respect to the origin at its entry (see axiom (S1)), and *oExit* the orientation of the origin at its exit with respect to the segment (see axiom (S2) in Fig.11).

The second part of the specification SEGMENTS contains five axioms describing some properties of the orientations between segments ("∠") and the entries and exits of segments. Figure 12 (a) and (b) show two examples which give the intuitive meanings of the axioms (S3) and (S4), respectively, where the thick arrow shows the origin at the place b. The three other axioms can be derived from the axiom (S4) directly.

```
spec PLACES = DOUBLECROSSCALCULUS then
        sort  Place < Location
        op    oOrigin : Place → Vector
end

spec SEGMENTS = PLACES then
        sort  Segment = {s : Vector • source(s) ∈ Place ∧ target(s) ∈ Place}
        ops   oEntry, oExit : Segment → OrientationDCC;
              oCourse : Segment → Vector;
              __⟶ __ : Place × Place →? Segment
        ∀ x, entry, exit : Place
        • source(oOrigin(x)) = x
        • oEntry(entry ⟶ exit) = oOrigin(entry) ∠ entry ⟶ exit          %% (S1)
        • oExit(entry ⟶ exit) =  entry ⟶ exit ∠ oOrigin(exit)           %% (S2)
then  %% recomputation of entry or exit
        ∀ a, b, c, d : Place
        • a ⟶ b ∠ b ⟶ c = a ⟶ b ∠ b ⟶ d + b ⟶ d ∠ b ⟶ c              %% (S3)
        • a ⟶ b ∠ b ⟶ c = oExit(a ⟶ b) + oEntry(b ⟶ c)                 %% (S4)
        • oExit(a ⟶ b) = front ⇒ a ⟶ b ∠ b ⟶ c = oEntry(b ⟶ c)
          % implied
        • oEntry(b ⟶ c) = front ⇒ a ⟶ b ∠ b ⟶ c = oExit(a ⟶ b)
          % implied
        • a ⟶ b ∠ b ⟶ c = a ⟶ d ∠ d ⟶ c ⇒
          oExit(a ⟶ b) + oEntry(b ⟶ c) = oExit(a ⟶ d) + oEntry(d ⟶ c)
          % implied
then
        ∀ a, b, c, d : Place
          • id(b) = id(d) ∧ def b ⟶ c ∧ def d ⟶ c ⇒
            oEntry(b ⟶ c) = oOrigin(b) ∠ oOrigin(d) + oEntry(d ⟶ c)   %% (S5)
          • id(b) = id(d) ∧ def a ⟶ d ∧ def a ⟶ b ⇒
            oExit(a ⟶ d) =  oExit(a ⟶ b) − oOrigin(d) ∠ oOrigin(b)     %% (S6)
end
```

Figure 11. Specification of places, route segments, and place integration

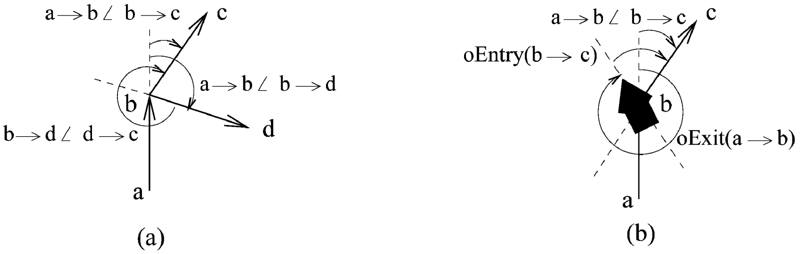


Figure 12. Properties of orientations between segments, entries and exits

The third part of Fig.11 contains two axioms which deal with place and route integration. For instance, while describing the route from a source place (s) to a target place (t), people usually describe the route from the source to some intermediate place, say b, and continue with an expression like "then take the route from b to t", if the route from b to t has been previously described or is already known to them. The entry recomputation rule enables the integration of the existing route from b to t to the route described from the start point to b, thus forming a complete route from s

to t. The integration of different routes into a route graph is often non-trivial and makes it necessary to extend the union of the corresponding sets of nodes and edges of directed graphs (cf. [15]). A significant step to integrate two different routes is to identify their *common places*, i.e., places with the same *Id* in our simplified case. In order to integrate two routes that share a common place, we can take the origin (denoted as a thick arrow) of one route segment at that place, calculate the entry (or exit, resp.) of the other route segment with respect to the chosen origin, and obtain a route graph. Figure 13 (a) shows two routes, in which b and d are identified as a common place, and Fig.13 (b) the route graph by integrating the two routes at place b and d according to the origin of b (i.e., o_b). Note that the integration of two routes with a common place offers new possibilities: if two segments emanate from this place, it becomes a decision point, which subsequent route segment to choose; instead of two routes originally, we now have four potential ones. Thus, it is not surprising that place (and route) integration requires some extra recomputation work. Conversely, each individual route (as a sequence of route segments, a separate little route graph) can be simplified, since each entry (or exit) is already recomputed during place integration.

Now we demonstrate the integration of two routes, for example, $a \longrightarrow b$ is a route segment from one route and $d \longrightarrow c$ a route segment from another route (Figs. 14(a) and (b)), where $\mathit{id}(b) = \mathit{id}(d)$. The entry of $b \longrightarrow c$ with respect to the origin of b is calculated as the sum of the orientation of the origin of d with respect to the origin of b (i.e., $\mathit{oOrigin}(b) \angle \mathit{oOrigin}(d)$) and the *oEntry* of $d \longrightarrow c$, as depicted in Fig. 14(c) and specified by the axiom (S5).

Similarly, Fig.15 shows an example of computing the exit of $a \longrightarrow d$ with respect to the origin of d, specified by the axiom (S6). The exit recomputation rule is useful, if the route from the start place to some intermediate place is known to a user, then he/she only needs to describe the route from the intermediate place to the target place. A complete route can be obtained by integrating them.

5.5 Route Descriptions

Based on the spatial relations *via*, *passOn* and *passBy* defined in equations 4, 5, 6, and 7 in Section 4.1, a CASL specification of route descriptions is presented in Fig. 16. We introduce the predicates $a \longrightarrow b$ *via* c to represent the relation "the location c is on the route segment $a \longrightarrow b$"; $a \longrightarrow b$ *pass* c *on* lr to represent "the location c is in the orientation lr with respect to the route segment $a \longrightarrow b$"; and $a \longrightarrow b$ *passBy* c to represent "the location c is either on the left or on the right-hand side of the route segment $a \longrightarrow b$".

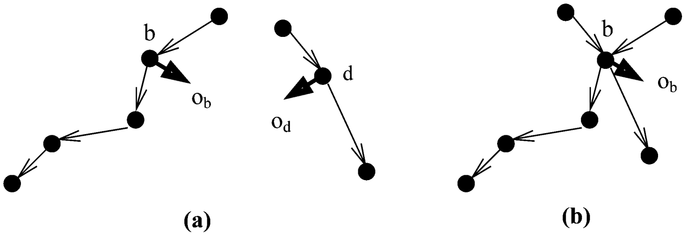


Figure 13. (a) Two routes and (b) their integration at places b and d

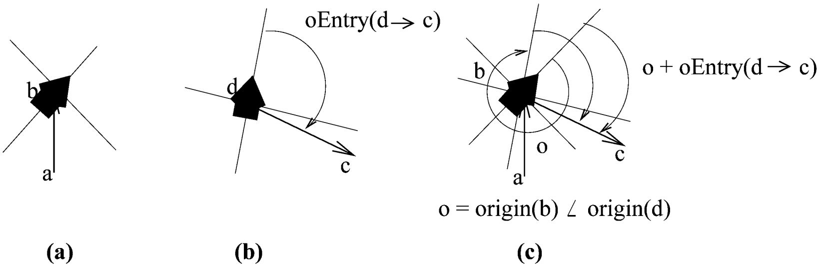


Figure 14. Integration of route segments: (a) the origin of b; (b) the entry of $d \longrightarrow c$; (c) computing the entry of $b \longrightarrow c$ w.r.t. the origin of b

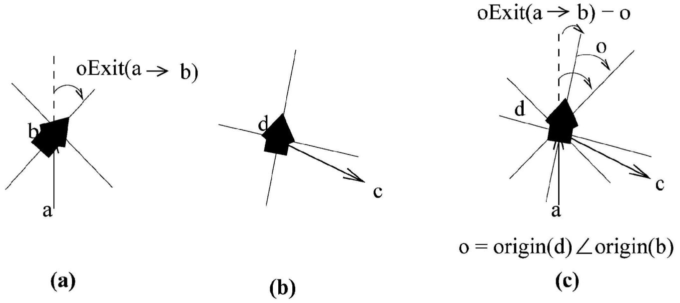


Figure 15. Integration of route segments: (a) the exit of $a \longrightarrow b$; (b) the origin of d; (c) computing the exit of $a \longrightarrow d$ w.r.t. the origin of d

Finally, the sample route instruction in Fig.1, represented in the *conceptual model* in Fig.3, is formalized as a set of axioms at the end of the specification (Fig.16), where *start*, p_1, p_2, p_3 and *door* are constants of the sort *Place*, and *door*, *lifts*, *mainStairs* and *stugaRoom* constants of *Landmark*.

```
spec RoutePredicates = Segments then
     pred __via__ : Segment × Location
     ∀ a, b, c: Place
     • a⟶ b via c ⇔ a⟶ b ∠ b⟶ c ▷ onCourse
then
     preds __passBy__ : Segment × Location;
           __pass__on__ : Segment × Location × OrientationDCC
     ∀ a, b, c: Place; lr: OrientationDCC
     • a⟶ b pass c on lr ⇔
       (∃ d: Location • a⟶ b via d ∧ a⟶ d ∠ d⟶ c ▷ lr)
     • a⟶ b passBy c ⇔
       a⟶ b pass c on left ∨ a⟶ b pass c on right
then
     sort  Landmark < Location
     ops   start, p1, p2, p3 : Place;
           door, lifts, mainStairs, stugaRoom : Landmark
     • oEntry(start⟶ door) = back
     • start⟶ p1 via door
     • oEntry(p1⟶ p2) = right
     • p1⟶ p2 passBy lifts
     • p2⟶ p3 passBy mainStairs
     • p2⟶ p3 ∠ p3⟶ stugaRoom ▷ front
end
```

Figure 16. Route predicates and the sample route

6 Linking the Conceptual Model to a Robot's Navigation Map

While the *conceptual model* provides a cognitively adequate framework for representing and reasoning about human route instructions, a quantitative map with metric information is still necessary, which enables an intelligent mobile robot to do self-localization, detect obstacles and execute the navigation tasks involved in the route instructions. However, to date most intelligent mobile robots use only a quantitative map for environment exploration (e.g., [3,52]), where no human interaction is involved, or they apply mapping functions to interpret high level instructions to a quantitative representation (e.g., [20,48,25]). In this section, we first compare our *conceptual model* with a quantitative model by interpreting route instructions using them. To handle interpretation disparities which may occur while interpreting route instructions in different models, linking them is necessary. Therefore, Section 6.2 introduces a generic route graph concept, which is then used in Section 6.3 to link the two route graph models compared in the first subsection.

6.1 Comparing Two Distinct Interpretations

Voronoi Graphs describe networks of places and connections between those places with maximal clearance to surrounding obstacles, while the conceptual model focuses on qualitative spatial information and abstracted landmarks and locations, enabling qualitative spatial reasoning on route instructions. In order to interpret route instructions against a global environment map as a Voronoi Graph, Mandel *et al.* suggested in [25] a fuzzy function based approach, where a set of fuzzy functions was developed for each primitive route expression, such as "turn", "go through", "pass by", "go until" or "on the left/right of". An algorithm using a search tree is developed to find the most likely target from the initial position of the agent, regarding a given sequence of primitive route expressions. The nodes of a search tree represent fuzzy rated places. The root represents the initial position with the highest possibility 1. Taking the next primitive expression from the sequence, the algorithm computes the following possible places and uses the fuzzy function defined for the selected expression to associate the places with fuzzy values, until all the expressions are considered. Finally, the algorithm evaluates the possible targets and selects the one with the highest fuzzy rate.

Navigation tasks belong to high-level cognitive processes that involve the assessment of environment information, the localization of spatial objects, and spatial relations between these objects. Describing a route to a robot is a special case of navigation, and may be subject to *knowledge-based mistakes* [35] made by humans. In [42], we sketched two distinct approaches, one interpreting human route instructions in the *conceptual model*, the other one using the fuzzy function based approach, introduced above, to interpret route instructions on the Voronoi graph. Comparing the two interpretations in such error situations, we found that they may deliver different results, called *interpretation disparities*. The following are some examples.

Unsolvable spatial information. Among the utterances in the Rolland corpus, a considerable number of them contained references to landmarks. An example is: "Drive past the copier room and the mailbox room". Here, the robot's knowledge of the intended entities, the "copier room" and the "mailbox room", is presupposed.

Assume that the robot has no such knowledge; then the fuzzy function approach will search for all possible places before eventually detecting missing information. However, the conceptual approach will first find out whether the place with the given landmark already exists in the environment. Only if it exists already, it begins to compute the route representation.

Spatial relation mismatches. A user may relate spatial objects incorrectly; take the above example again: the copier room is in fact located subsequent to the mailbox room with respect to the robot's current position. The *conceptual model* might detect this mismatch, if there exist relations in the conceptual environment, from which the correct spatial relation between the copier room and the mailbox room can be inferred. During integration of the new route description, an inconsistent situation will be detected using logical reasoning.

To interpret this instruction, the search algorithm in the fuzzy function approach might first reach the place for the mailbox room without memorising it, and then find the place for the copier room. After finding the place marked with "copier room", it continues to search for a place marked with "mailbox room". A zigzag route would result in such a case.

Orientation mismatches. Moreover, mistakes can also be caused by using incorrect orientation, a special case of deciding spatial relations. Consider again the scenario described in Section 4.1, where the lifts may only be passed on the right. Now suppose that the utterance "pass by the lifts on your left" is given by a user. At the conceptual level, this inconsistency can easily be detected using logical reasoning on spatial relations. In contrast, the search algorithm of the fuzzy interpretation will search all possible places on the left without success.

Consequently, we believe that robust human-robot interaction requires the representation of human route instructions and the robot's representation to be at distinct abstraction levels. Thus the *conceptual model* has to be linked to the quantitative level representation, and each representation will separately contribute to resolve mismatches.

6.2 Generic Route Graph Ontology

As already mentioned in Section 3.1, route graphs have been introduced as a general concept to be used for navigation in a variety of scenarios. In [15], Krieg-Brückner *et al.* developed a generic route graph concept using an ontology to specify route graph terms, interrelations between them and to standardize or mediate between different usages of route graphs. Figure 17 shows its formalization in CASL. GENERICGRAPH is a specification with two sorts as its parameters. Therefore, we can use various kinds of edges and nodes (e.g., local reference directions) and edges (e.g., courses of route segments) to instantiate GENERICGRAPH, and get different graphs. In the generic graph specification, three sorts *Graph*, *Edge* and *Node* are introduced. A graph has nodes and edges, which are specified by the overloaded predicates *has*. Operations *source* and *target* are functions from an edge to its source and target, respectively. Furthermore, a mapping (*info*) between the sort *Edge* or *Node* to the corresponding parameter is declared. The first axiom (G1) states that if a graph contains an edge, then it contains the source and target nodes of the edge.

The second part of the specification defines the infix "node deletion" function

"__/__" on graphs recursively. The third part specifies whether a node is *reachable* from another node in a graph. The predicate *isSequence*, presented in the third part of the specification, characterizes a special class of graphs, called *Path*. A path has a start point from which any other nodes can be reached, and any two different edges of a path have different sources and targets.

```
spec GenericGraph [sort NodeInfo] [sort EdgeInfo] =
        sorts  Graph, Edge, Node
        preds  __has__ : Graph × Node;
               __has__ : Graph × Edge
        ops    source, target : Edge→ Node
               info : Edge → EdgeInfo
               info : Node → NodeInfo
        ∀ g : Graph; e: Edge
        • g has e ⇒ g has source(e) ∧ g has target(e)                  %% (G1)
then
        op     __/__ : Graph × Node → Graph
        ∀ g, g' : Graph; e: Edge; n: Node
        • ¬(g has n) ⇔ g / n = g
        • g has n ∧ g / n = g' ⇒
          ¬ (g' has n) ∧ ¬ ((source(e) = n ∨ target(e) = n) ∧ g' has e)
then
        pred   reachable : Graph × Node × Node
        ∀ g : Graph; e : Edge; l, m, n : Node
        • reachable(g, n, n)
        • g has e ∧ source(e) = m ∧ target(e) = n ⇒ reachable(g, m, n)
        • reachable(g, l, m) ∧ reachable(g, m, n) ⇒ reachable(g, l, n)
then
        pred   isSequence : Graph
        • isSequence(g) ⇔ ∃ n : Node • ∀ m : Node •
          ¬ n = m ∧ g has n ∧ g has m ⇒ reachable(g, n, m)
          ∧
          ∀ e1, e2 : Edge • g has e1 ∧ g has e2 ∧
              (source(e1) = source(e2) ∨ target(e1) = target(e2)) ⇒ e1 = e2
        sort   Path = { g : Graph • isSequence(g) }
end
```

Figure 17. Specification of the generic graph in Casl

Now we move to the specification of generic route graphs, and start with RouteSegment and GenericPlace. Each *Place* has an *origin* of type *RefSystem*; and the sort *RouteSegment* is associated with three operations *course*, *entry* and *exit* (see the first two specifications in Fig. 18).

The generic specification GenericRouteGraph has two parameters referring to the specification GenericPlace and RouteSegment (see the third specification in Fig. 18), which is an instantiation of GenericGraph, where the sorts *RefSystem* and *Course* are provided for the two parameters *NodeInfo* and *EdgeInfo*, respectively. Any specifications that extend GenericPlace or RouteSegment can be used to instantiate the generic route graph specification. The sorts *Graph*, *Node* and *Edge* are renamed to *RouteGraph*, *Place* and *RouteSegment*, respectively. Additionally, the sort *Route* is defined as a special class of paths containing no circle. *RefSystem*, *Place*, *Course*, *Entry* and *Exit* have been left unspecified for the moment.

6.3 Linking the Conceptual Model to the Voronoi Graph based on the Generic Route Graph

The *conceptual model* can be seen as an abstraction of the Voronoi graph used by our wheelchair robot. Relating these two levels of spatial representations requires an abstraction relation, more specifically from a route to a route segment, or a graph to a node, as in Fig. 19. When a set of nodes SN in a lower level graph is abstracted to a single node N in a higher level graph, some obvious conditions must hold, for example, all incoming/outgoing edges at a node in SN must correspond to incoming/outgoing edges at N; for each pair of incoming and outgoing edges at N there must be a corresponding connecting path in the lower level graph. To relate the *conceptual model* to the Voronoi graph formally, we use the generic route graph specification given in Fig.18.

```
spec GenericPlace =
     sort  RefSystem, Place
     op    origin: Place → RefSystem
end

spec RouteSegment =
     sorts RouteSegment, Course, Entry, Exit
     ops   course: RouteSegment → Course;
           entry:  RouteSegment → Entry;
           exit:   RouteSegment → Exit
end

spec GenericRouteGraph [GenericPlace] [RouteSegment] =
     GenericGraph [sort RefSystem] [sort Course]
     with Graph ↦ RouteGraph,
          Node ↦ Place,
          Edge ↦ RouteSegment
then
     sort  Route = { r : Path • ∀ n, m : Place • r has n ∧ r has m ∧
                          reachable(r, n, m) ⇒ ¬ reachable(r, m, n)}
end
```

Figure 18. Specification of the generic route graph in Casl

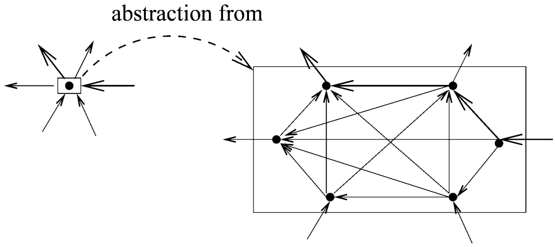


Figure 19. Abstraction a sub-graph to a place

From the specification in Section 6.2 we see that the generic route graph specifies only very general route graph concepts, where *RouteSegment*, *Course*, *Entry*, *Exit*, as well as *RefSystem* are unspecified. Given different definitions of these sorts, the generic route graph can be instantiated to route graphs at distinct abstraction levels. For instance, our *conceptual model* of route instructions is in fact an instance of the generic route graph, if we take PLACES and SEGMENTS in Fig. 11 to instantiate GENERICPLACE and ROUTESEGMENT in the GENERICROUTEGRAPH specification, respectively. The sort *RefSystem* and the operation *origin* of GENERICPLACE are instantiated by *Vectors* and *oOrigin* of PLACES. *RouteSegment* from ROUTESEGMENT is instantiated by *Segment* from SEGMENTS, *Course* by *Vectors*, *Entry* and *Exit* by *OrientationDCC*, and the operations *course*, *entry* and *exit* by *oCourse*, *oEntry* and *oExit*, respectively, as illustrated in Fig.20. The sort *RouteGraph* from GENERICROUTEGRAPH is renamed to *ConceptualModel*.

```
spec ConceptualRouteGraph = GenericRouteGraph
            [Places fit
            RefSystem ↦ Vector, origin ↦ oOrigin]
            [Segments fit
            RouteSegment ↦ Segment,
            Course ↦ Vector, Entry ↦ OrientationDCC, Exit ↦ OrientationDCC,
            course ↦ oCourse, entry ↦ oEntry, exit ↦ oExit]
      with RouteGraph ↦ ConceptualModel
end
```

Figure 20. Specification of the conceptual route graph in CASL

On the other hand, the Voronoi graph can be instantiated in a similar way, as illustrated in Fig.21. Suppose the specification for nodes of a Voronoi graph is VPLACES, in which the data structures for metric *Angle*, *VPlace* with x- and y-coordinates in a global reference system, and the operation *vOrigin*, mapping a point to the global reference direction, are specified. In the specification VSEGMENTS, the sorts *VSegment* and *VEdge* define the edges and their courses (e.g., length and width); and the operations *vCourse*, *vEntry* and *vExit* map an edge to its course, entry and exit (e.g., the angle between the course to the global reference direction). The sort *RouteGraph* is now renamed to *VoronoiGraph*.

```
spec VoronoiGraph = GenericRouteGraph
            [VPlaces fit
            RefSystem ↦ Angle, Place ↦ VPlace, origin ↦ vOrigin]
            [VSegments fit
            RouteSegment ↦ VSegment,
            Course ↦ VEdge, Entry ↦ Angle, Exit ↦ Angle,
            course ↦ vCourse, entry ↦ vEntry, exit ↦ vExit]
      with RouteGraph ↦ VoronoiGraph, Route ↦ vRoute
end
```

Figure 21. Specification of the voronoi route graph in CASL

Finally, to abstract the Voronoi graph to the *conceptual model* is in fact to define a set of abstraction relations *abstractionFrom* between the two graphs, their nodes and their edges, as specified in Fig.22 with CONCEPTUALROUTEGRAPH and VORONOIGRAPH as parameters.

The first axiom (A1) states three elementary properties of the route graph abstraction. If a conceptual route graph is an abstraction of a Voronoi graph, then a place or an edge in the Voronoi graph can only be abstracted to one single place or edge in the *conceptual model*; and if an edge is an abstraction of an edge in the Voronoi graph, then its source and target should be the abstractions of the source and target of the edge in the Voronoi graph, respectively.

The next four axioms are about the correspondence between edges and places at the two levels. If a place or an edge is an abstraction of a place or an edge in the Voronoi graph, then it is in the conceptual route graph (A2 and A3). Moreover, any place or edge in the conceptual graph is an abstraction of some edge or place in the Voronoi graph (A4 and A5).

The last axiom (A6) states that if the source and target of an edge are abstracted to two different places in the conceptual graph, then there is an edge between them in the conceptual graph, thus the incoming/outgoing edges for a set of Voronoi places correspond to the incoming/outgoing edges of their common abstraction place.

```
spec RouteGraphAbstraction[ConceptualRouteGraph] [VoronoiGraph] =
then
          preds __abstractionFrom__ : ConceptualModel ∗ VoronoiGraph;
                __abstractionFrom__ : Place ∗ VPlace;
                __abstractionFrom__ : Segment ∗ VSegment
then
          ∀ p1, p2: Place; vp: VPlace; s1, s2 : Segment; vs: VSegment;
            m: ConceptualModel; vg: VoronoiGraph
            • m has p1 ∧ m has p2 ∧ m has s ∧
              vg has vp ∧ vg has vs ∧ m abstractionFrom vg ∧ ⇒
              (p1 abstractionFrom vp ∧ p2 abstractionFrom vp ⇒ p1 = p2) ∧
              (s1 abstractionFrom vs ∧ s2 abstractionFrom vs ⇒ s1 = s2) ∧
              (s1 abstractionFrom vs ⇒ source(s1) abstractionFrom source(vs) ∧
               target(s1) abstractionFrom target(vs))                          %% (A1)
then
          ∀ p : Place; vs: VSegment; m: ConceptualModel; vg: VoronoiGraph
            • m abstractionFrom vg ∧ vg has vp ∧ p abstractionFrom vp
              ⇒ m has p                                                        %% (A2)
            • m abstractionFrom vg ∧ vg has vs ∧ s abstractionFrom vs
              ⇒ m has s                                                        %% (A3)
          ∀ p: Place; s: Segment; m: ConceptualModel; vg: VoronoiGraph
            • m abstractionFrom vg ∧ m has p
              ⇒ ∃ vp:VPlace • vg has vp ∧ p abstractionFrom vp                 %% (A4)
            • m abstractionFrom vg ∧ m has s
              ⇒ ∃ vs:VSegment • vg has vs ∧ s abstractionFrom vs               %% (A5)
then
          ∀ p1, p2: Place; vs: VSegment; m: ConceptualModel; vg: VoronoiGraph
            • m abstractionFrom vg ∧ vg has vs ∧ p1 abstractionFrom source(vs) ∧
              p2 abstractionFrom target(vs) ∧ ¬ p1 = p2 ⇒
              ∃ s:Segment • s abstractionFrom vs ∧ source(s) = p1 ∧ target(s) = p2  %% (A6)
end
```

Figure 22. Specification of the route graph abstraction in CASL

In the specification ROUTEGRAPHABSTRACTION we do not give any specific definition of the operation *abstractFrom* for segments or places. It is a requirement specification: in any implementation of the route graph abstraction these operations should be implemented in such a way, that all the axioms given in the requirement specification hold.

7 Related Work

Our *conceptual model* has been inspired by Kuipers' work on the Spatial Semantic Hierarchy (SSH), especially, its topological level of spatial representation[17]. The topological map represents the navigation space as a collection of places, paths and regions, which are associated by spatial relations. The *conceptual model* can be seen as an instance of the SSH's topological map. The Route Graph provides an ontological framework to structure places and paths, while the spatial relations between places and paths are defined according to the Double-Cross calculus. Topological relations from the SSH, such as *at*, *on*, *left_of*, have been redefined in our *conceptual model* with the orientation predicate specified in Fig. 10. The instantiation is empirically well motivated, and enables efficient computational reasoning about human route navigation knowledge. On the other hand we use location-based relative orientations to represent relations between places and paths, instead of regions and their connections.

Thus, some topological relations from the SSH like *in* between a place and a region is beyond the expressiveness of our model at present.

There are many different qualitative spatial calculi which are region-, interval- or point-based. We limit ourselves to those calculi specifically concerning the representation and reasoning about the directions and point configurations, which are further classified into the calculi where directions are determined by a predefined reference frame and those using relative orientation. The Star Calculus[37] is a typical example for representing and reasoning about qualitative relations between points in a two dimensional space with a given reference direction, which generalizes several well-known calculi such as those using cardinal directions distinguished by Frank[10,11]. In addition to the Double-Cross calculus, several calculi have been developed recently, mostly as a variant or an extension of the Double-Cross calculus. In [28], Moratz *et al.* proposed calculi on different levels of granularity using so called dipoles (i.e., oriented line segments formed by a pair of two points, a start point and an end point) to deal with intrinsic orientation information by specifying qualitative relations between dipoles. In [27], Moratz and Ragni developed the Ternary Point Configuration calculus, which is derived from the single cross frame with a front/back dichotomy and eight orientation relations, but makes finer distinctions to model qualitative distances. We used the Double-Cross calculus for the following two reasons: First, we aim to provide a conceptual model which is powerful enough and allows to bridge the gap between a human's cognitive spatial knowledge and a robot's quantitative level representation, since humans often use relative orientations in their route instructions (see, e.g., [46,8,54,43]). Second, to be applied in real human-robot interaction, a qualitative spatial calculus should support an efficient reasoning process.

On the other hand, metric or topological maps are widely used in the mobile robotics community (cf. [53,49,3,36,26,18,19]) to solve robotic problems, such as concurrent mapping, localization and exploration. Probabilistic methods for simultaneous localization and mapping (SLAM, cf. [49]), for example, are usually accurate and reliable to do incremental localization in a metrical map with a single frame of reference. Beeson *et al.* proposed in [3] an autonomous place detection approach using an extended Voronoi graph. The symbolic nature of topological representations of Voronoi graphs allows some higher-level computation process, for example, containment and connectivity. However, little attention has so far been paid to the aspects of human-robot interaction. Many researchers in the robotics area gradually recognize the importance of human-robot interaction in personal service robotics (cf. [40,51]). The continuous improvement in robotic autonomy unavoidably changes the focal point of the interaction to a higher level, where integrating quantitative maps used by robots to carry out spatial tasks autonomously, and qualitative spatial models for representation and reasoning about human spatial knowledge is much more important (cf. [8,45,43]).

Human-robot interaction is different from both human-computer interaction and human-human interaction. To date a number of empirical studies on human-robot interaction have been published (cf. [33,14,39,5,45]). Kanda *et al.*, for example, reported in [14] a study, in which a humanoid robot listens to the route instructions given by a human, while Ono *et al.* studied in [33] how to develop a robot that describes routes to people. In their experiments the Wizard of Oz method was used and the application of speech and gesture were investigated. Moreover, there is research on the direct interpretation of human instructions to the robot's behavior using *functional primitives* abstracted from corpora (cf. [25,20,4,48]). In the IBL project[20,4], for example, Lauria *et al.* abstracted a number of functional primitives from the IBL corpus and implemented them as a set of procedures corresponding to the robot's behaviors, which are to some extent similar to the fuzzy functions used to interpret route instructions on the Voronoi graph (see Section 6.1). However, the direct interpretation approach does not provide conceptual representations as grounding for natural language expressions. Thus, it can only be used in behavior-based robots rather than interactive robots. Furthermore, several models have been proposed, which attempt to capture route instructions as spatial models[58,17,23]. Route graphs, referenced in Section 3.1 and 6.2 and detailed in [56,15,41,24], are prominent examples. The Route Graph[56,15] is essentially a graph-based abstract representation of space which can be instantiated to a number of different *levels* including Voronoi-like low-level representations to more abstract representations such as the conceptual user model described in Section 4. While route graphs are very interesting as a method for representing complete navigation spaces, little work has been performed to date on showing how individual route graphs can be composed from verbal descriptions, or on how individual route descriptions can be integrated into route graphs in computational systems. We believe that this paper meets such requests.

8 Conclusion

In this paper, we have presented an approach that combines qualitative spatial calculi using orientation information and route graphs in one conceptual model to represent and reason about human route navigation knowledge. The formal specification of the model in the algebraic specification language Casl, together with some software tools for the language, makes it possible to check the consistency of the model and to prove properties (e.g., [57]). Moreover, the formalization of the generic route graph and its instantiations to both the *conceptual model* and the quantitative level representation, used by our wheelchair robot for localization and navigation, provides a formal foundation for linking them, in order to achieve a robust and efficient human-robot interaction.

We believe that there are two major contributions of this work. First, combining qualitative spatial calculi and topological route graphs for modelling human spatial knowledge is valuable since it allows a more widespread understanding of the practical issues involved in interpreting and reasoning about route instructions in shared controlled robot systems. Second, the formalization of the *conceptual model* using a formal specification language is useful beyond checking consistency and proving properties of the model, since it also serve as a basis for comparing different models and linking representations at distinct abstraction levels.

The focus of the *conceptual model* is on the information about relative orientation. Future work will consider regions to cover descriptions such as "go into the cafeteria" and "along the river". Furthermore, the *conceptual model* is qualitative. There are situations in which quantitative information is used to give route descriptions, for example "*about 20 to 25 meters*" or "*about 30 degrees to the left*". Such quantitative data are sometimes redundant, such as "*about 20 to 30 seconds, then at the end of the* *corridor*"; if the place "the end of the corridor" is uniquely defined in the environment, then the information "about 20 to 30 seconds" is irrelevant for finding the route. However, there are still cases, where quantitative data are indispensable, such as "*about 30 degrees to the left*". The *conceptual model* discussed in this paper may be extended to treat such measurements in a qualitative (cf. [27]) or even quantitative way.

The major concern of this paper was the development of a qualitative model for representing and reasoning about route instructions, as well as its formalization. For motivating our work we studied several existing corpora regarding human-robot interaction on navigation tasks, especially the Rolland corpus. However, it could be an interesting future research to carry our some empirical study to evaluate the model discussed in this paper.

Acknowledgements

We gratefully acknowledge the support of the "Deutsche Forschungsgemeinschaft" (DFG) through the Collaborative Research Center SFB/TR 8 "Spatial Cognition". We also like to thank Dr. Klaus Lüttich and Dr. Christian Maeder for useful discussions on the CASL specifications and helping us to check their consistency using the Heterogeneous Tool Set, HETS, and Dr. Yohei Kurata for his careful proofreading.

References

[1] Allen JF, Maintaining knowledge about temporal intervals. CACM, 1983, 26(11): 832–843.

[2] Astesiano E, Bidoit M, Krieg-Brückner B, Kirchner H, Mosses P, Sannella D, Tarlecki A. CASL-The Common Algebraic Specification Language. Theoretical Computer Science, number 286. Springer-Verlag, 2002.

[3] Beeson P, Jong NK, Kuipers B. Towards automous topological place detection using the extended voronoi graph. IEEE Int. Conf. on Robotics and Automation, 2005.

[4] Bugmann G, Klein E, Lauria S, Kyriacou T. Corpus-based robotics: A route instruction example. In: Proc. of IAS-8. 2004.

[5] Burke JL, Murphy RR, Coovert MD, Riddle DL. Moonlight in miami: Field study of human-robot interaction in the context of an urban search and rescue disaster response training exercise. Human-Computer Interaction, 2004, 19(1&2).

[6] Chronis G, Skubic M. Robot navigation using qualitative landmark states from sketched route maps. In: Proc. of the IEEE 2004 Int'l Conf. on Robotics and Automation. 2004. 1530–1535.

[7] Cohn AG, Bennett B, Gooday J, Gotts NM. Qualitative spatial representation and reasoning with the region connection calculus. Geoinformatics, 1997, 1: 1–44.

[8] Denis M. The description of routes: A cognitive approach to the production of spatial discourse. Cahiers de Psychologie Cognitive, 1997, 16: 409–458.

[9] Fischer K. What Computer Talk Is and Is Not: Human-Computer Conversation as Int'l Communication. Computational Linguistics, volume 17, 2006.

[10] Frank AU. Qualitative spatial reasoning with cardinal directions. In: Proc. of the Seventh Austrian Conference on Artificial Intelligence. Springer-Verlag, 1991.

[11] Frank AU. Qualitative spatial reasoning: Cardinal directions as an example. Journal of GIS, 1996, 10(3): 269–290.

[12] Freksa C. Qualitative spatial reasoning. In: Mark DM, Frank AU, eds. Cognitive and Linguistic Aspects of Geographic Space. Kluwer, 1991.

[13] Freksa C. Using orientation information for qualitative spatial reasoning. Theories and Methods of Spatio-Temporal Reasoning in Geographic Space. LNCS 639, Springer-Verlag, 1992. 162–178.

[14] Kanda T, Kamasima M, Imai M, Ono T, Sakamoto D, Ishiguro K, Anzai Y. A humanoid robot that pretends to listen to route guidance from a human. Autonomous Robots, 2006, 22(1):

87–100.

[15] Krieg-Brückner B, Frese U, Lüttich K, Mandel C, Mossakowski T, Ross RJ. Specification of an ontology for route graphs. In: Freksa C, Knauff M, Krieg-Brückner B, Nebel B, Barkowsky T, eds. Proc. of Spatial Cognition IV, Chiemsee, Germany. LNCS 3343, Springer-Verlag, 2004. 989–995.

[16] Krieg-Brückner B, Shi H. Orientation calculi and route graphs: Towards semantic representations for route descriptions. In: Proc. of GIScience 2006, Münster, Germany. LNCS 4197, Springer-Verlag, 2006.

[17] Kuipers B. The spatial semantic hierarchy. Artificial Intelligence, 2000, 119: 191–233.

[18] Lankenau A, Röfer T. A safe and versatile mobility assistant. IEEE Robotics and Automation Magazine, 2001, 1: 29–37.

[19] Lankenau A, Röfer T. Mobile robot self-localization in large-scale environments. In: Proc. of the IEEE Int'l Conf. on Robotics and Automation. 2002. 1359–1364.

[20] Lauria S, KyriacouT, Bugmann G, Bos J, Klein E. Converting natural language route instructions into robot executable. In: Proc. of the 2002 IEEE Int'l Workshop on Human and Robot Interactive Communication. 2002. 223–228.

[21] Ligozat G. Reasoning about cardinal directions. Journal of Visual Languages and Computing, 1998, 9: 23–44.

[22] Lüttich K, Krieg-Brückner B, Mossakowski T. Tramway networks as route graphs. In: Proc. of Formal Methods for Automation and Safety in Railway and Automotive Systems. 2004. 109–119.

[23] MacMahon M. A framework for unterstanding verbal route instructions. In: Proc. of AAAI Fall Symposium on the Intersection of Cognitive Science and Robotics: From Interfaces to Intelligence. 2004.

[24] Mallot HA, Gillner S. Route navigation without place recognition: What is recognized in recognition-triggered response? Perception, 2000, 29: 43–55.

[25] Mandel C, Frese U, Röfer T. Robot navigation based on the mapping of coarse qualitative route descriptions to route graphs. In: Proc. of the IEEE/RSJ Int'l Conf. on Intelligent Robots & Systems. 2006. 205–210.

[26] Mandel C, Huebner K, Vierhuff T. Towards an autonomous wheelchair: Cognitive aspects in service robotics. In: Proc. of Towards Autonomous Robotic Systems (TAROS 2005). 2005.

[27] Moratz R, Ragni M. Qualitative spatial reasoning about relative point position. Journal of Visual Languages & Computing, 2008, 19(1): 75–98.

[28] Moratz R, Renz J, WolterD. Qualitative spatial reasoning about line segments. In: Proc. of the 14th European Conf. on Artifical Intelligence. IOS Press, 2000.

[29] Mossakowski T. CASL: From semantics to tools. In: Graf S, Schwartzbach M, eds. TACAS 2000. LNCS 1785, Springer-Verlag, 2000. 93–108.

[30] Mossakowski T, Maeder C, Lüttich K. The heterogeneous tool set. In: Grumberg O, Huth M, eds. Proc. of TACAS 2007. LNCS 4424, Springer-Verlag, 2007. 519–522.

[31] Mosses PD. CASL Reference Manual. LNCS 2960, Springer-Verlag, 2004.

[32] Mosses PD, Bidoit M. CASL — the Common Algebraic Specification Language: User Manual. LNCS 2900, Springer-Verlag, 2004.

[33] Ono T, Imai M, IH. A model of embodied communication with gestures between humans and robots. Annual Meeting of the Cognitive Science Society, 2001.

[34] Randell D, Cui Z, Cohn A. A spatial logic based on regions and connection. In: Proc. of 3rd Int'l Conf. on Knowledge Representation and Reasoning. Morgan Kaufmann, 1992. 165–176.

[35] Reason J. Human Error. Cambridge University Press, 1990.

[36] Remolina E, Kuipers B. Towards a general theory of topological maps. Journal of Artifical Intelligence, 2004, 152: 47–104.

[37] Renz J, Mitra D. Qualitative direction calculi with arbitrary granularity. In: Proc. of the PRICAI 2004. LNCS 3157, Springer-Verlag, 2004. 65–74.

[38] Renz J, Nebel B. On the complexity of qualitative spatial reasoning: A maximal tractable fragement of the region connection calculus. Artificial Intelligence, 1999, 108(1-2): 69–123.

[39] Robins B, Dautenhahn K, Dubowshi J. Does appearance matter in the interaction of children

with autism with a humanoid robot. Interaction Studies, 2006, 7(3).

[40] Rogers E, Murphy R. Final report for DARPA/NSF study on human-robot interaction. http://users.csc.calpoly.edu/ erogers/HRI/HRI-report-final.html, 2001.

[41] Schölkopf, B, Mallot HA. View-based cognitive mapping and path planning. Adaptive Bahavior, 1995, 3: 311–348.

[42] Shi H, Mandel C, Ross RJ. Interpreting route instructions as qualitative spatial actions. In: Barkowsky T, Freksa C, Knauff M, Krieg-Br üc kner B, Nebel B, eds. Proc. of the Int'l Conference Spatial Cognition 2006. LNCS, Springer-Verlag, to appear.

[43] Shi H, Tenbrink T. Telling Rolland where to go: HRI dialogues on route navigation. In: Coventry K, Tenbrink T, Bateman J, eds. Spatial Language and Dialogue. Oxford University Press, to appear.

[44] Skubic M, Matasakis P, Forrester B, Chronis G. Extracting navigation states from a hand-drawn map. In: Proc. of the IEEE 2001 Int'l Conf. on Robotics and Automation, 2001.

[45] Stubbs K, Hinds P, Wettergreen D. Autonomy and common ground in human-robot interaction: A field study. Special Issue on Interacting with Autonomy, 2007, 22(2): 42–54.

[46] Talmy L. How language structures space. In: Pick HL, Acredolo LP, eds. Spatial Orientation: Theory, Research and Application. Plenum, NY, 1983.

[47] Tenbrink T. Identifying objects on the basis of spatial contrast: an empirical study. In: Spatial Cognition IV: Reasoning, Action, Interaction. Int'l Conf. Spatial Cognition 2004, Springer, 2005. 124–146.

[48] Theobalt C, Bos J, Chapman T, Espinosa-Romero A. Talking to Godot: Dialogue with a mobile robot. In: Proc. of the 2002 IEEE Int'l Conf. on Intelligent Robots & Systems. 2002. 1338–1343.

[49] Thrun S. Probabilistic algorithms in robotics. AI Magazine, 2000, 21(4): 93–109.

[50] Thrun S. Robotics mapping: A survey. In: Lakemeyer G, Nebel B, eds. Exploring Artificial Intelligence in the New Millenium. Morgan Kaufmann, 2002.

[51] Thrun S. Towards a framework for huamn-robot interaction. Human Computer Interaction,2004, 19(1&2): 9–24.

[52] Thrun , Fox D, Burgard W, Dellaert F. Robust monte carlo localization for mobile robots. Journal of Artifical Intelligence, 2001, 128(1-2).

[53] Thrun S, Gutmann S, Fox D, Burgard W, Kuipers B. Integrating topological and metric maps for mobile robot navigation: A statistical approach. In: AAAI-98. 1998. 989–995.

[54] Tversky B. Structures of mental spaces – how people think about space. Environment and Behavior, 2003, 35(1): 66–80.

[55] Tversky B, Lee P. How space structures language. In Freksa C, Habel C, Wender K, eds. Spatial Cognition: An interdisciplinary Approach to Representation and Proc. of Spatial Knowledge. LNCS 1404, Springer-Verlag, 1998. 157–175.

[56] Werner S, Krieg-Brückner B, Hermann T. Modelling navigational knowledge by route graphs. In: Spatial Cognition II: Integrating Abstract Theories, Empirical Studies, Formal Methods, and Pratical Applications. LNCS 1849, Springer-Verlag, 2000. 259–316.

[57] Wölfl S, Mossakowski T, Schröder L. Qualitative constrain claculi: Heterogeneous verification of composition tables. In: 20th Int'l FLAIRS Conf. AAAI Press, 2007. 665–670.

[58] Zimmermann K, Freksa C. Qualitative spatial reasoning using orientation, distance, and path knowledge. Applied Intelligence, 1996, 6: 49–58.