A Framework for Mobile Intention Recognition in Spatially Structured Environments

Peter KIEFER and Klaus STEIN
{peter.kiefer,klaus.stein}@uni-bamberg.de
Laboratory for Semantic Information Technologies, University of Bamberg

Abstract. Mobile intention recognition is the problem of inferring an agent's intentions from the spatio-temporal behavior she shows. We present a framework for mobile intention recognition that covers all levels, from low-level position data to high-level intentions, with the detailed structure of the motion track as an intermediate level. Our framework is new in explicitly including the structure of space into the intention recognition process at each step. We demonstrate the benefit of adding spatial knowledge by an ex-post interpretation of the behavior tracked in a location-based game.

Keywords. Spatial cognition, plan recognition, mobile computing

'There is a great deal of work in low-level perception, and a great deal in high level recognition - including this thesis. The semantic gap between the output of the low-level processes and the high-level inference engines remains wide, and few have ventured to cross it.' *H. Kautz (1987) [12, outlook (p. 127)]*

1. Introduction

Suppose you see a car showing a strange driving behavior. It might be traveling at a speed far below average, slowing down at green traffic-lights, changing its direction frequently, or stopping at the roadside several times. You probably infer that the driver is foreign and trying to find his way to some destination. This problem of inferring an agent's intentions from her behavior is called *intention recognition problem*[1]. Over many years, plan recognition has mainly considered problems like language and story understanding, or the design of intelligent user interfaces (see [7] for an overview).

Mobile intention recognition problems differ from these 'traditional' use cases, especially because mobile behavior happens in space and time. The location where a certain behavior takes place tells us about possible intentions. The detailed (spatio-temporal) characteristics of the motion track can also help. In the example above, an insecure driving style could be a sign for a foreign, drunk, or a novice driver. An experienced observer (e. g. a police officer) could intuitively distinguish these cases by the motion track de-

[1] Also known as plan recognition problem, [22]

tails. Automating this kind of inference requires several steps to bridge the gap between low-level processing and high level intentions. One recent approach for mobile activity recognition is presented by Liao et al [16]: they use a hierarchical Markov model with the layers gps measurement, location on a street network, transportation mode, trip segment, goal, and novelty. Due to the specific motivation of their paper, the semantics of 'goal' is 'target location' so there is no need for a more complex user intention model.

In this paper, we present a framework for mobile intention recognition that covers all steps from low-level motion track data to high level intention models. It utilizes knowledge about the structure of space at each level of the processing hierarchy. A modular architecture allows to change the methods used for each step, thus making it easily adaptable to different problem domains. The desktop application INTENSIVE, which helps in testing and improving mobile intention recognition algorithms, complements our framework.

In section 2, we explain how space changes the problem of intention recognition, and introduce the location-based game CityPoker. Section 3 proposes our framework, and explains each stage of the framework's processing hierarchy. We show how the intentions of a CityPoker player are recognized using her motion track, and spatial knowledge about the gaming area (real game data, see Fig. 1). It is important to note that the algorithms presented in section 3 for CityPoker are not claimed to be optimal for *any* mobile intention recognition problem. The idea is rather that spatial knowledge can improve intention recognition, independent from the concrete methods used. The intention simulation environment (INTENSIVE) is outlined in section 4. We give related work in section 5, and conclude in section 6.

2. Mobile intention recognition

In mobile computing we often find ourselves in situations with restricted possibilities of human-computer interaction, due to a limitation of the user's cognitive resources [2]. In these situations, a device should automatically select the information service the user currently needs, and either present it directly (information-push), or ease the access to that service (for instance by assigning it to a 'hot button').

Most location-based services directly deduce the information service needed from the user's current position (on-enter/on-leave paradigm). This direct mapping from raw position data to information service does not work satisfactory in many situations. One prominent example is the room-crossing problem [21]: a user wants to cross a room and accidentally enters areas around POIs (e. g. exhibits in a museum room) triggering various information services not needed in that situation (annoying information push). By mapping the user's behavior to intentions, before mapping the intention to an appropriate information-service, we can create an *intention-aware information-service*. In the museum example, such an intention-aware information service distinguishes room-crossing and exhibition-visiting.

Implementing such a service for a mobile device is a challenging task. It is important that our service performs well under real-time conditions. This is especially hard when computing everything on the restricted resources of a mobile device. Server-based architectures, as an alternative, may suffer from latency, data transfer costs, and network unavailability. Everything needs to work incrementally, i. e. before we know any future

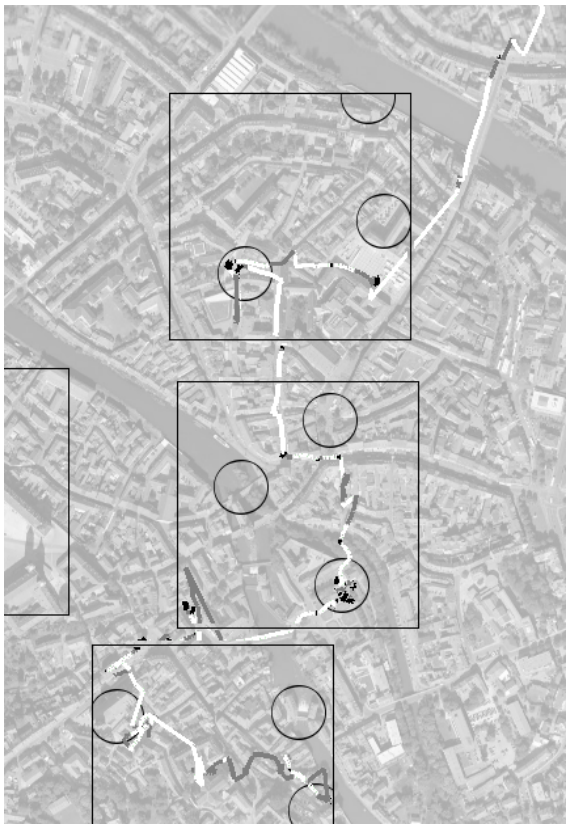

Figure 1. Spatial structure (regions and caches) and motion track of one team in CityPoker

behavior. Another assumption to be made is that we are dealing with *keyhole* intention recognition [7], i. e. the user does not change his behavior to manipulate the intention recognition process.

2.1. Spatial is special

We call an intention recognition problem a *mobile* one, if (at least some of) the behaviors used for interpretation are *spatio-temporal behavior*. This can be any behavior derived from the user's position (with timestamp), or position history. The most basic kind of behavior are simple enter/leave-region behaviors. More complex types of spatio-temporal behavior can be derived by motion track analysis if we analyze a number of sequential motion points (a motion segment). As we will see in section 3, this may include several steps of preprocessing (section 3).

Not only the positional data make mobile intention recognition special, but also the spatial context[2]. We often have knowledge about the spatial structure, e. g. a number of points of interest (POI) with circular or polygonal areas around them [1,9]. Others add a street network to these locations [16], use spatial tessellation [11], or formalize space

[2] As most important form of context; we do not lead discussions about the general notion of context here.

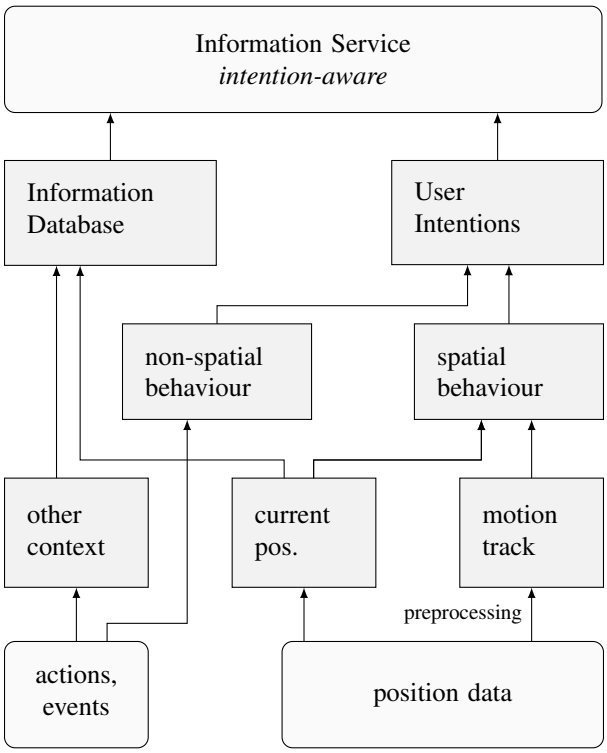

Figure 2. System architecture.

with Spatial Conceptual Maps [19]. A hierarchical structure between POIs can be created by 'assigning to each location a number representing the level of detail that the location belongs to' [6, p. 13]. The next step is to use *partonomies*, i. e. a structure that stores polygons in a tree with part-of relations between regions [24], or even structures allowing overlapping regions. Partonomies are often found in the way how human structure space, e. g. shop $\subset$ shopping_district $\subset$ city_district $\subset$ city $\subset$ county $\subset \ldots$

The basic intuition of our framework is that not every'thing' (every intention, behavior, action, ...) should be interpreted equal in any region. We will see in section 3 how this exactly works, and how we can make the intention recognition algorithm more efficient by using spatial knowledge.

2.2. A location-based game: CityPoker

CityPoker, our example for the following section, is a location-based game played by two adversary teams who try to optimize their poker hand by finding and changing hidden cards. The game is supported by a mobile assistance system (J2ME on smartphones) with GPS localization. It is usually played by bike (for a detailed game description see [20]).

CityPoker imposes the following partonomial structure: the game board (e. g. a city) contains five rectangular regions ($\text{REGION}_1, \ldots, \text{REGION}_5$), each of which contains

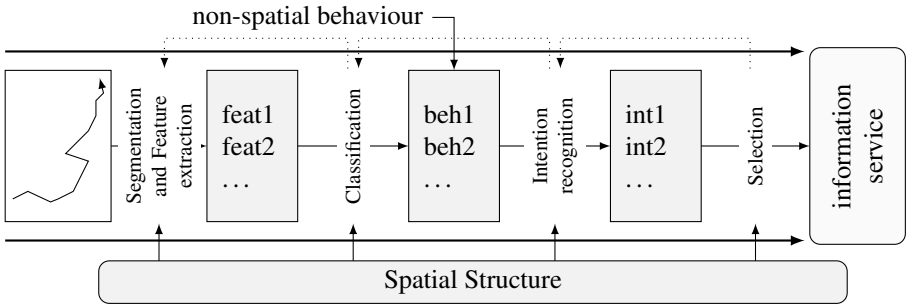

Figure 3. Stream of processing.

three circular regions ($\text{CACHE}_{1,1}, \ldots, \text{CACHE}_{5,3}$). Figure 1 displays only those regions the player entered during the game.[3]

When a player (a team) enters a REGION, the mobile device poses a multiple-choice quiz with three answers. Each answer will lead the player to one of the three caches. Arriving at the cache, the device reveals a perceptual hint where to find the hidden cards (e. g. 'the cards are hidden under a green wooden object'). This kind of 'fuzzy' hint helps the player to locate the exact place of the cards in the circular cache. The wrong answer will lead the player to the wrong cache. After some searching time she may correct her answer and head for another cache.

The information services available are maps of different zoom level (*citymap*, *regionmap*, *cachemap*), *perceptual hint*, *game status*, *quiz*. In case the intention recognition mechanism guesses wrong, all services are also accessible for the user by manual selection.

3. A framework for intention recognition in spatially structured environments

Figure 2 gives an overview on our framework. Positional data are the most important input for mobile intention recognition. From that data we can access the current position and use it in any processing step. Due to the memory limitations of mobile devices, it is not possible to keep a complete motion track history. Instead, motion segments are classified, and kept as sequence of behaviors. Aditionally non-spatial behavior is included: user actions (in CityPoker: reporting the server when the card has been found), and external events (receiving a notification of the other team's actions). Now user intentions are inferred and finally the information service selects the suitable data from the database, possibly parameterized by the current position and other context (e. g. game status). Figure 3 shows the (simplified) processing pipeline. The single steps are described in the following sections using the CityPoker example.

3.1. Segmentation and feature extraction

The first steps in processing motion data are segmentation and feature extraction. Depending on the concrete environment motion data consists of a sequence of sensor data.

[3]The track was recorded during a CityPoker game on Oct. 30, 2006, played by a team of 4 girls of age between 10 and 14.

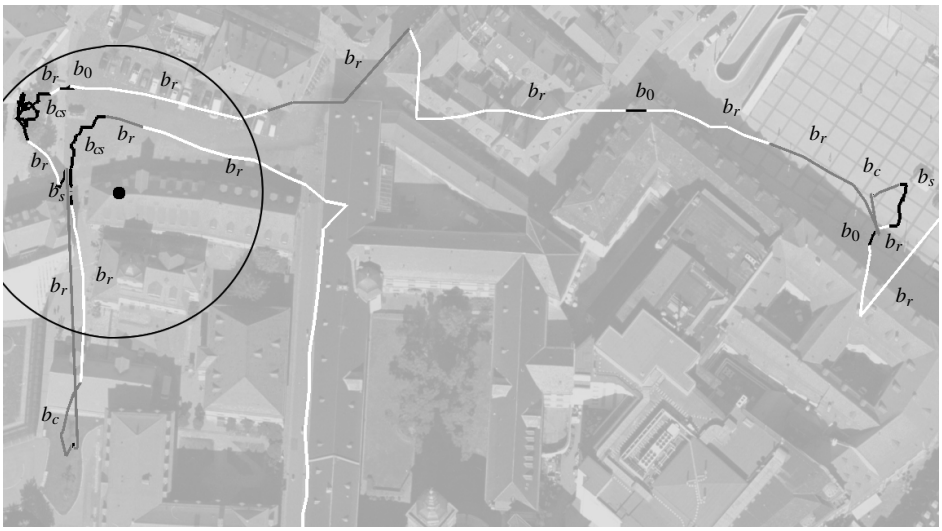

Figure 4. Segmented motion track with classified behavior sequence in a cache region. (The player enters from the right.)

This may include heading, speed, steering angle and so on. In the CityPoker example we get positional information from a GPS device.[4] Now the interesting features are extracted from the motion data and the track is segmented accordingly. The first important feature is the region of the partonomy we are in. The second feature interesting in our example is the current speed. We distinguish low ($< 1\,\mathrm{m/s}$), medium ($\leq 10\,\mathrm{m/s}$) and high speed ($> 10\,\mathrm{m/s}$), abbreviated as ls, ms, and hs in the following.[5] The third feature is the curvature of the motion. To keep things simple we used the average angle of direction change in the last five points for measuring the curvature (low (lc), if $(\alpha_{k-4} + \ldots + \alpha_k)/5 < 45°$, high (hc) otherwise).[6] The last feature is the maximum diameter of the convex hull of the segment (i. e. the largest distance between two segment points).

The features used as well as the thresholds can be chosen dependent on the region entered or left, e. g. while walking down a street other features may be interesting than while being on a market place. Each change in one of the features (here: region, speed, curvature) can trigger the start of a new segment (entering/leaving a region, speed drops from medium to low, curvature gain, etc.). Every time a new segment starts the according feature vector is passed to the next processing step: classification. Other features (here: segment diameter) do not trigger a new segment but are subject to change as the user moves on. In this case the feature vector of the current segment is updated and the change is propagated to the classification.

[4] A sequence of coordinates with timestamps (approx. time between two measures is 2 s). This allows to compute direction and speed. Depending on data quality preprocessing may be necessary to catch sensor errors and smoothing.

[5] The values are specifically choosen for the example domain: $1\,\mathrm{m/s}$ is slow walk (due to data noise the GPS measured speed nearly never drops to 0, even if the user does not move), and $10\,\mathrm{m/s}$ is very fast, at least for a biker in the city.

[6] For more sophisticated motion trach micro structure measures see [23, chapter 12].

Production Rules P		***Grounding***	
HandleRegion	$\rightarrow$ *SelectCache GotoCache GetCard*	REGION	*(1)*
SelectCache	$\rightarrow$ *FindParkingPos SolveQuiz*	REGION	*(2)*
FindParkingPos	$\rightarrow b_r$	REGION	*(3)*
SolveQuiz	$\rightarrow b_0 \mid b_s \mid b_{sc}$	REGION	*(4)*
GotoCache	$\rightarrow$ *ApproachCache GotoCache* $\mid$ *OrientToCache GotoCache* $\mid \omega$	REGION	*(5)*
ApproachCache	$\rightarrow (b_r \mid b_c)^+$	REGION	*(6)*
OrientToCache	$\rightarrow (b_s \mid b_{cs} \mid b_0)^+$	REGION	*(7)*
GetCard	$\rightarrow$ *SearchCard* $b_{\text{ChangeCard}}$	CACHE	*(8)*
SearchCard	$\rightarrow$ *RoamCache SearchCard* $\mid$ *DetailSearch SearchCard* $\mid \omega$	CACHE	*(9)*
RoamCache	$\rightarrow (b_r)^+$	CACHE	*(10)*
DetailSearch	$\rightarrow (b_c \mid b_s \mid b_{cs} \mid b_0)^+$	CACHE	*(11)*

Figure 5. SGIS production rules for CityPoker (a part of the complete grammar).

3.2. Classification

The classification step is mainly a mapping function from the feature vector to a given set of user behaviours. This can be realized in various ways, from decision trees and table lookup to fuzzy or probabilistic approaches. For the CityPoker example we have

$$\begin{array}{lll} \{R_1 \ldots R_n\} \times \{\text{ls}, \text{ms}, \text{hs}\} \times \{\text{lc}, \text{hc}\} \times \mathscr{R}^+ & \rightarrow & \{b_r, b_c, b_{cs}, b_s, b_0\} \\ (\text{region}, \quad \text{speed}, \quad \text{curv.}, \quad \text{diam.}) & \rightarrow & \text{behaviour} \end{array}$$

We distinguish

riding	b_r	$\text{speed} = \text{hs} \vee (\text{speed} = \text{ms} \wedge \text{curv.} = \text{lc})$
curving	b_c	$\text{speed} = \text{ms} \wedge \text{curv.} = \text{hc}$
slowcurv.	b_{cs}	$\text{speed} = \text{ls} \wedge \text{curv.} = \text{hc} \wedge \text{diam.} > 5\,\text{m}$
sauntering	b_s	$\text{speed} = \text{ls} \wedge \text{curv.} = \text{lc} \wedge \text{diam.} > 5\,\text{m}$
standing	b_0	$\text{speed} = \text{ls} \wedge \text{diam.} \leq 5\,\text{m}$

As the gps data is noisy even a stillstanding subject will show a certain speed and curvature, so the diameter of the region covered in the current segment is used to distinguish different behaviours. This also means that the classification of the current segment may change from b_0 to b_s or b_{cs} with the user moving on. Certainly the classification could also use the spatial structure, in our example we leave this to the next step. Figure 4 shows a part of the motion track from Figure 1, annotated with classified behaviors.

In some settings it can be efficient to combine segmentation, feature extraction and classification in one step in the implementation to reduce overhead.

3.3. Intention recognition

The heart of our framework is the intention recognition mechanism itself. We choose *Spatially Grounded Intentional Systems* (SGIS) [20] which are context free grammars (CFG) with intentions as non-terminals, and behaviors as terminals. The applicability of a production rule in an SGIS depends on the region of their respective terminal behaviors. Grammars are, in general, intuitive to understand for the knowledge engineer.

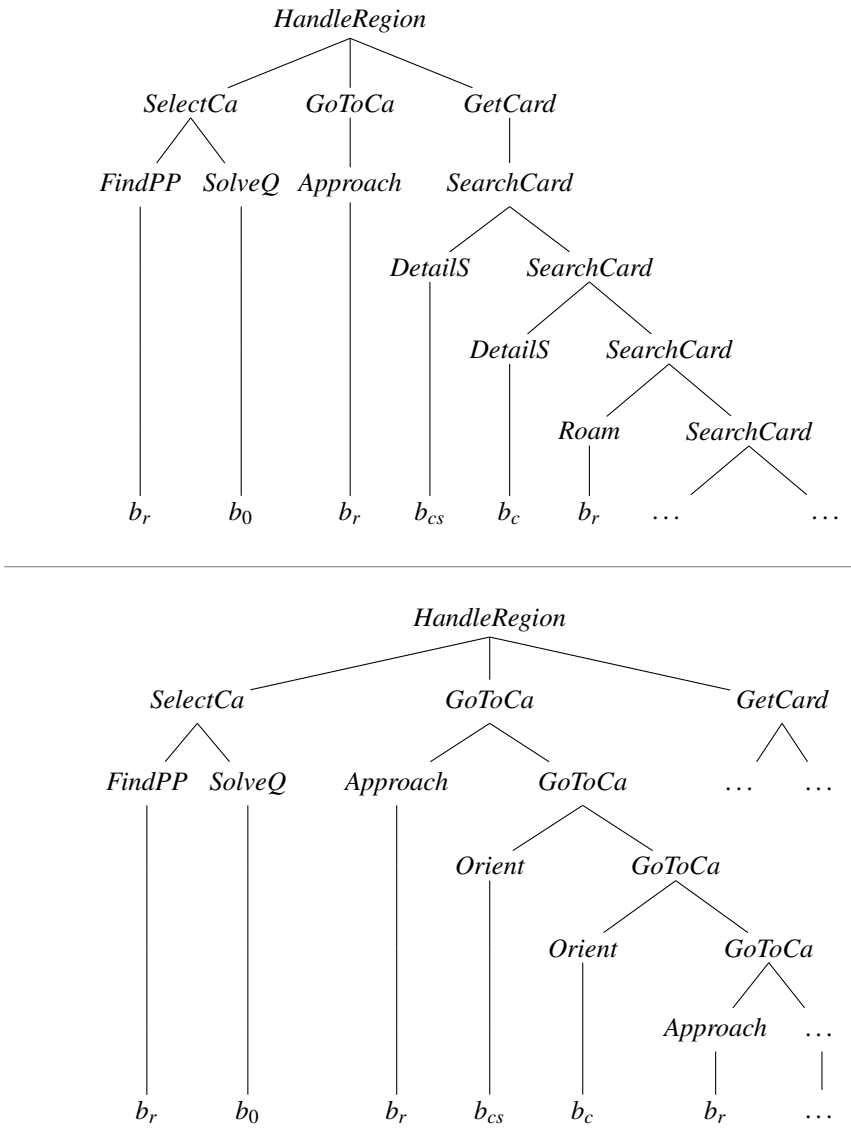

Figure 6. Parsing ambiguity if we had no spatial knowledge.

Figure 5 lists a number of SGIS rules for CityPoker. For reasons of clarity, we selected only rules describing user intentions in a REGION: first, the player finds herself a good place to stand, before she solves the quiz. She navigates towards her selected CACHE by target-oriented moving (*ApproachCache*), interrupted by way-finding (*OrientToCache*). These non-terminals are mapped to behaviors (rules 6, 7).[7] In a similar way, searching the card consists of (target-oriented) roaming through the CACHE, and

[7] These rules are written simplified, and can easily be rewritten as context-free rules.

Production Rules P	***Grounding***
GetCard → *SearchCard AccLeave SearchCard*	REGION*(12)*
AccLeave → *(something)*	REGION*(13)*

Figure 7. Modifying the CityPoker example to catch an accidental leaving behavior.

a (random appearing) detail search. Finally, our only non-spatial behavior ($b_{\text{ChangeCard}}$) occurs when the players has succeeded in finding the card.

If the classification of the last segment (i. e. the last terminal) changes during processing reparsing is necessary. By carefully choosing the production rules one can assure that this will not change the active intention.

Figure 6 shows two possible parse trees for the start of the behavior sequence from Fig. 4. Without spatial knowledge, both trees were possible. We cannot decide at which position in the behavior stream we have to place the cut between *GotoCache* and *GetCard*, thus dealing with high ambiguity. SGIS reduce ambiguity by restraining the applicability of rules to certain regions. Thus, the rule resolving *GetCard* is not applicable if one of its terminals is outside the cache. With that additional information, the top tree is not possible.

The grammar from Fig. 5 is simplified and does not catch all use cases, for instance, if the player accidentally leaves the CACHE during searching (as she actually does in Fig. 4). Also, an unsuccessful search (refine search to another CACHE) has not been formalized.

The context-freeness of SGIS makes them more expressive than finite-state machines but some use cases require more context-sensitivity. Context-sensitive grammars (CSG), on the other hand, are too complex to parse. Researchers have recently argued for the use of mildly context-sensitive grammars (MCSG) for intention recognition [14,10]. MCSGs have been developed in the natural language processing (NLP) community and fall in complexity between CFGs and CSGs. As one of their properties they allow for *cross-dependency relations* between non-terminals in the parse tree. Constraints concerning the applicability of production rules can be formulated for these dependencies. Dependencies with *spatial constraints* are the heart of Spatially Constrained Context-Free Grammars (SCCFG), and Spatially Constrained Tree-Adjoining Grammars (SCTAG), both introduced in [13].

In CityPoker, one such constraint arises if we try to catch the accidental leaving behavior by adding the rules from Fig. 7. To formulate rule (12) correctly, we would need to state that the first *SearchCard* must happen in the same CACHE like the second *SearchCard*. In general, the production rule is applicable if the region R_1 of intention I_1 has a certain spatial relation r (e.g. *identical*, *touches*, *contains*, ...) to another region R_2 where intention I_2 happens. This kind of dependency cannot be formulated with SGIS, but with SCCFGs. If our use case requires to cross such dependencies, SCTAGs are the best choice.

3.4. Information service selection

The task here is to find an information service for the *active intention*. That is, the intention closest to the current terminal behavior. For instance, in Fig. 6 at the second behavior, in both trees the intention *SolveQ* is the active one. Also, by climbing up the tree

to the more high-level intentions, we can easily create an ordered queue of information services by selecting one service for each intention up to the root node.

Finally, finding the information service can be solved very simple by a mapping mechanism, like *SolveQ* → *quiz*. We include space into the selection mechanism by parameterizing the information service. Thus, the information service *quiz* will not just show any quiz, but the one relevant in the current REGION.

3.5. Backpropagation

The processing as shown so far uses a strictly linear pipeline (Fig. 3). For many scenarios this is most efficient regarding memory and time, but there are settings where information from higher steps can help on lower steps. For example, the intention recognition reveals which intention the user currently has (or may have). By propagating this information back to the classification step it can be used to choose the most efficient classification algorithm (e. g. if *SearchCard* applies a more sophisticated and therefore more expensive classification is needed while otherwise a simple table lookup is sufficient).

As the classification step may give any arbitrary sequence of behaviours, the intention recognition production rules need to have some kind of "catch all", i. e. at any time there must be at least some rule applicable for any behaviour. One way to solve this is to tell the classification step that the last behaviour is not applicable and should be reclassified. This will certainly not work if classification is done with a deterministic table lookup algorithm (as there is exactly one behaviour matching) but works fine with probabilistic approaches, where different behaviours apply with different probability.

4. INTENSIVE: a simulation and testing environment

Our framework is flexible in four ways: (1) it can be used in different use cases by choosing appropriate intentions, behaviors, and information services. (2) It allows to change the algorithm chosen on each layer. (3) It allows to parameterize each algorithm. (4) It allows to change the geographic model, i.e. to port the use case to a new area.

It is part of this research to identify the best configuration of algorithms and parameters ((2) and (3)) for use cases and spatial models of different complexity ((1) and (4)). While some decisions on the configuration can be made analytically (e. g. regarding the complexity of the formal grammar), others need to be made empirically using real motion track data. Optimally, we would like to test any possible configuration outdoors with real users, and see if the information services proposed are perceived as helpful. This is certainly not feasible due to the high effort associated with extensive outdoor testing.

The INTention SImulation enVironmEnt (INTENSIVE) addresses this problem. With the desktop application INTENSIVE we can easily configure an intention recognition mechanism (design time) and test it by playing back previously recorded motion track data (runtime). At design time, the INTENSIVE user specifies a *geographic model*, an *intention processing model*, and a *motion track log file* which contains a sequence of latitude/longitude pairs with timestamp. At runtime, the system reads the log file, processes the gps data according to the intention processing model, and outputs the behaviors, intentions, and information services that occur.

An INTENSIVE intention processing model consists of a number of *algorithm blocks* which do the computations. Algorithm blocks have input and output ports over

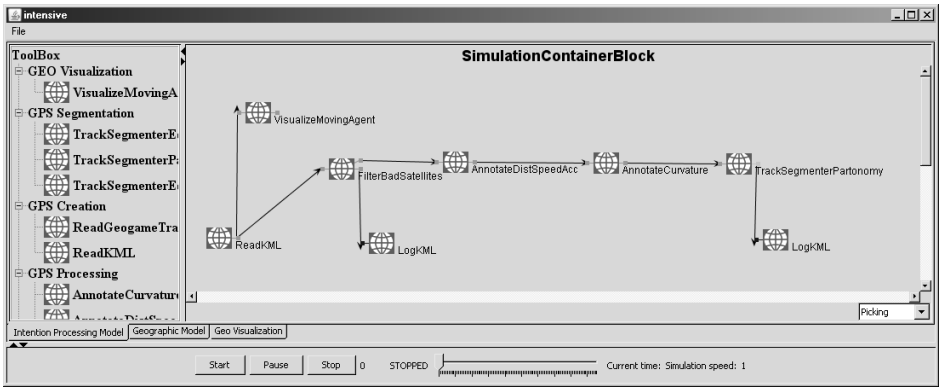

Figure 8. Part of an intention processing model in INTENSIVE (preprocessing, feature extraction, and segmentation).

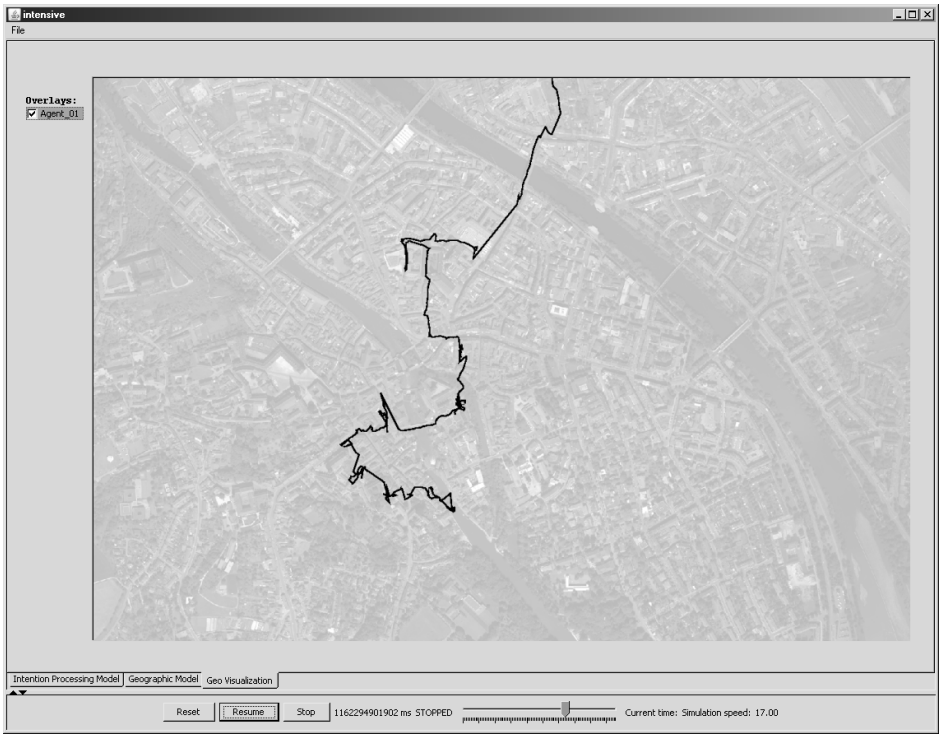

Figure 9. Monitoring the agent in the geo visualization screen.

which they exchange information with other algorithm blocks. The user adds, connects, and parameterizes algorithm blocks easily using the graphical user interface (Fig. 8). INTENSIVE comes with a number of predefined algorithm blocks containing a set of important algorithms for each processing step. The plugin architecture of intensive allows developers to implement new algorithms and add them to the system. Geographic models are imported from KML-files. During runtime, the user can watch his agent[8] moving in the *geo visualization screen* (Fig. 8). To speed up testing, it is possible to adjust the speed of the runtime. By going through a cycle of manual model adjustment and testing, the user hopefully reaches an intention processing model that perfectly handles the use case.

At time of writing the INTENSIVE tool is still under development. The current version of INTENSIVE targets at researchers and experienced application developers. The idea is to try out configurations on the desktop and only implement the best ones for a mobile device. Automatic code generation which converts the intention processing model to code for the target platform will be part of future work.

5. Related Work

Besides Liao et al [16] (see section 1), Bui chooses another probabilistic network based approach with several layers [5]. In both papers, computations are done on a server. Probabilistic grammars were discussed in the areas of pattern recognition, machine vision, and action recognition [3,17]. An overview on grammars in machine vision is given in [8]. Probabilistic state dependent grammars [18] constrain the applicability of a rule dependent on a general state variable. The generality of this state variable leads to an explosion in symbol space if trying to apply a parsing algorithm, so that an inference mechanism is chosen which translates the grammar into a Dynamic Bayes Network (DBN). Geo-referenced DBN are proposed by [4] to fuse sensory data and cope with the problem of inaccurate data.

Learning the locations relevant for a mobile user, instead of modeling them by hand, is done in [1,16]. This can be reasonable in domains where the spatial structure is not fixed. In CityPoker, we could learn the size of the circular caches. Also the steps of our processing hierarchy could replace modeling by learning, e.g. the classification. We could also try to detect the types of behavior interesting for a domain automatically with methods from the spatio-temporal data mining community [15].

6. Conclusion and outlook

We have explained the characteristics of mobile intention recognition, and presented an appropriate framework for including space in the intention recognition process. For the intention recognition mechanism itself we chose a grammatical formalism, enhanced by spatial knowledge. The analysis of a real CityPoker motion track has exemplified how the spatio-temporal details of a trajectory can be exploited.

[8] In general, INTENSIVE is also open to simulate team intentions by using several motion track logs. Log files may also contain non-spatial behavior like interaction with the device.

In our future research, we will further explore the use of MCSGs for mobile intention recognition. Different MCSG formalisms (e.g. Tree Adjoining Grammars, Head Driven Phrase Structure Grammars) need to be considered for different use cases and tested in INTENSIVE. INTENSIVE itself will be extended to cope with maps and motion tracks from different sources and different code generation backends for different mobile platforms.

An implementation on a mobile device will help to explore the feasibility of parsing MCSGs on restricted resources. Interpreting behavior that happens in a spatial structures with overlapping regions is another interesting issue.

References

[1] Daniel Ashbrook and Thad Starner. Using gps to learn significant locations and predict movement across multiple users. *Personal and Ubiquitous Computing*, 7(5):275–286, 2003.

[2] J. Baus, A. Krueger, and W. Wahlster. A resource-adaptive mobile navigation system. In *Proc. 7th International Conference on Intelligent User Interfaces*, pages 15–22, San Francisco, USA, 2002. ACM Press.

[3] A.F. Bobick and Y.A. Ivanov. Action recognition using probabilistic parsing. In *Proc. of the Conference on Computer Vision and Pattern Recognition*, pages 196–202, 1998.

[4] Boris Brandherm and Tim Schwartz. Geo referenced dynamic Bayesian networks for user positioning on mobile systems. In Thomas Strang and Claudia Linnhoff-Popien, editors, *Proceedings of the International Workshop on Location- and Context-Awareness (LoCA), LNCS 3479*, pages 223–234, Munich, Germany, 2005. Springer-Verlag Berlin Heidelberg.

[5] Hung H. Bui. A general model for online probabilistic plan recognition. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2003.

[6] Hung Hai Bui, Svetha Venkatesh, and Geoff A. W. West. Tracking and surveillance in wide-area spatial environments using the abstract hidden markov model. *IJPRAI*, 15(1):177–195, 2001.

[7] Sandra Carberry. Techniques for plan recognition. *User Modeling and User-Adapted Interaction*, 11(1-2):31–48, 2001.

[8] Gaurav Chanda and Frank Dellaert. Grammatical methods in computer vision: An overview. Technical Report GIT-GVU-04-29, College of Computing, Georgia Institute of Technology, Atlanta, GA, USA, November 2004. ftp://ftp.cc.gatech.edu/pub/gvu/tr/2004/04-29.pdf.

[9] H.M. Dee and D.C. Hogg. Detecting inexplicable behaviour. In *Proceedings of the British Machine Vision Conference*, pages 477–486. The British Machine Vision Association, 2004.

[10] Christopher W. Geib and Mark Steedman. On natural language processing and plan recognition. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1612–1617, 2007.

[11] Björn Gottfried and Jörn Witte. Representing spatial activities by spatially contextualised motion patterns. In LNAI 4434 G. Lakemeyer et al., editor, *RoboCup 2007, International Symposium,*, pages 329–336. Springer, 2007.

[12] Henry A. Kautz. *A Formal Theory of Plan Recognition*. PhD thesis, University of Rochester, Rochester, NY, 1987.

[13] Peter Kiefer. Spatially constrained grammars for mobile intention recognition. In Christian Freksa et al., editor, *Spatial Cognition VI*, Lecture Notes in Computer Science. Springer, 2008. accepted.

[14] Peter Kiefer and Christoph Schlieder. Exploring context-sensitivity in spatial intention recognition. In *1st Workshop on Behavior Monitoring and Interpretation, KI 2007*, 2007.

[15] Patrick Laube, Marc van Krefeld, and Stephan Imfeld. Finding remo - detecting relative motion patterns in geospatial lifelines. In *Developments in Spatial Data Handling, Proceedings of the 11th International Symposium on Spatial Data Handling*, pages 201–215, 2004.

[16] Lin Liao, Donald J. Patterson, Dieter Fox, and Henry Kautz. Learning and inferring transportation routines. *Artificial Intelligence*, 171(5-6):311–331, 2007.

[17] Dimitrios Lymberopoulos, Abhijit S. Ogale, Andreas Savvides, and Yiannis Aloimonos. A sensory grammar for inferring behaviors in sensor networks. In *IPSN '06: Proceedings of the fifth international*

conference on Information processing in sensor networks, pages 251–259, New York, NY, USA, 2006. ACM Press.

[18] David V. Pynadath and Michael P. Wellman. Probabilistic state-dependent grammars for plan recognition. In *Proceedings of the 16th Annual Conference on Uncertainty in Artificial Intelligence*, pages 507–514, 2000.

[19] Nancy Samaan and Ahmed Karmouch. A mobility prediction architecture based on contextual knowledge and spatial conceptual maps. *IEEE Transactions on Mobile Computing*, 4(6):537–551, 2005.

[20] Christoph Schlieder. Representing the meaning of spatial behavior by spatially grounded intentional systems. In *GeoSpatial Semantics, First International Conference*, volume 3799 of *Lecture Notes in Computer Science*, pages 30–44. Springer, 2005.

[21] Christoph Schlieder and Anke Werner. Interpretation of intentional behavior in spatial partonomies. In *Spatial Cognition III, Routes and Navigation, Human Memory and Learning, Spatial Representation and Spatial Learning*, volume 2685 of *Lecture Notes in Computer Science*, pages 401–414. Springer, 2003.

[22] C.F. Schmidt, N.S. Sridharan, and J.L. Goodson. The plan recognition problem: An intersection of psychology and artificial intelligence. *Artificial Intelligence*, 11(1-2):45–83, 1978.

[23] Klaus Stein. *Qualitative Repräsentation und Generalisierung von Bewegungsverläufen*. PhD thesis, Institut für Informatik der Technischen Universität München, 2003.

[24] Klaus Stein and Christoph Schlieder. Recognition of intentional behavior in spatial partonomies. In *ECAI 2004 Worskhop 15: Spatial and Temporal Reasoning (16th European Conference on Artificial Intelligence)*, 2005.