A demonstration of a natural-language pedestrian routing system

Johan Boye*, Morgan Fredriksson†, Jana Götze*, Jürgen Königsmann†

*KTH, School of Computer Science and Communication, 10044 Stockholm, Sweden
†Liquid Media, Hammarby allé 34, 12061 Stockholm, Sweden

1. Overview

This demonstration highlights some features of a dialogue system for pedestrian routing (Boye et al. 2012) that uses natural-language routing instructions to help pedestrians navigate and explore the city in real-time. The instructions are given as the user is moving: When the user reaches a node p_i in the planned route, he is informed how to reach the next node p_{i+1}. The user can ask for (more) instructions if he is unsure where to go next, and reply to the system's questions (by "yes" or "no"). The system keeps track of the user's position by GPS relayed from the user's smartphone. Obviously, it is increasingly interesting to create and study such applications as more and more people have smartphones equipped with a GPS receiver.

In order to make the system scalable and transferrable across cities, the system uses only open, crowd-sourced geographical data from OpenStreetMap (Haklay 2008). Such data is admittedly more error-prone and not as consistently organised compared to data released from a governmental surveying organization, but on the other hand, national surveying data tends to be expensive, hard to access, and vary in format from nation to nation. Moreover, the possibility to port the system to different cities is important from a research point of view. Different cities have different characteristics (narrow streets or big boulevards, a grid-like structure or a medieval city plan, etc), and these features will influence the success of different system strategies for producing route instructions.

A key problem for routing systems is the generation of natural language references to objects in the city. Such references form the link between the algebraic and geometric model of the domain on the one hand, and the communication with the user on the other. The strategy employed by our system is to, as far as possible, describe the route using *landmarks*, by which we understand distinctive objects in the city environment. Street names in instructions are avoided unless there are no other options available. For instance, the system would prefer the instruction "Walk towards the cafe on the corner" rather than "Walk towards King's street". We base these heuristics on previous research showing that it is predominantly by landmarks that people describe routes to one another (Denis et al. 1999), and that inclusion of landmarks into system-generated instructions for a pedestrian raises the user's confidence in the system (Ross et al. 2004).

For the system to give instructions based on landmarks, it needs an internal model of what the user can and cannot see from any specific point. To this end, we have implemented a *visibility engine*, which is used to calculate whether there is a free line-of-sight between any two points in the city. When the dialogue system wants to issue a routing instruction, it uses the visibility engine to calculate which landmarks in the near vicinity are visible, and then selects from this set the landmark which is assumed to be most salient based on a number of factors.

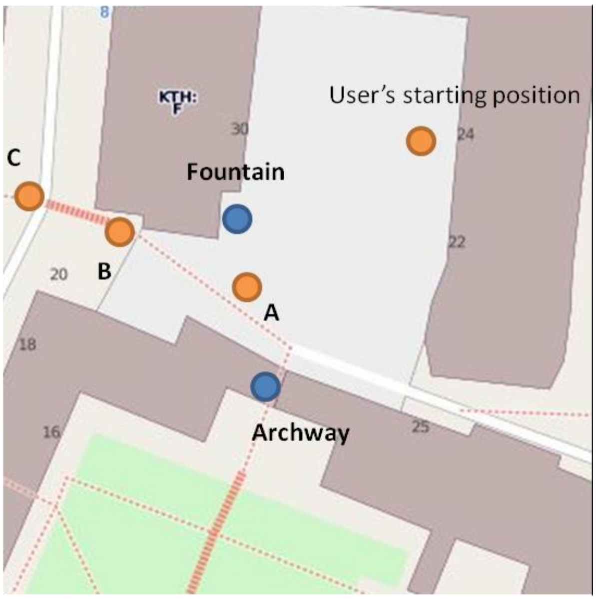

Figure 1: A pedestrian routing scenario

The figure above shows a situation in which the user should be directed to node A. However, there is no good way of describing A, so the system computes the set of visible landmarks (the fountain and the archway, in this case). Thus possible utterances from the system could be "Can you see the fountain?", "Walk left of the fountain", "Can you see the archway?" etc.

The visibility engine not only returns information whether it is possible to see from A to B, but also returns information about objects intersected by the visibility vector AB, such as streets, parks, etc. This allows for better route instructions, as illustrated by Figure 2 below.

Figure 2: Information returned from the visibility engine allows for instructions like “Walk into the park” (left), “Walk out of the park (middle), and “Walk through the park (right)”.

The visibility information, which is continuously updated as the user is moving, forms part of the system’s user model. This user model represents the system’s information about the user’s knowledge of the city. Besides the visibility information it is updated by information of where the user has previously been, what he has previously seen, and what the system previously has described to the user. Such information allows the system to use references like “the intersection where you were previously”, “the cafe I mentioned earlier”, etc.

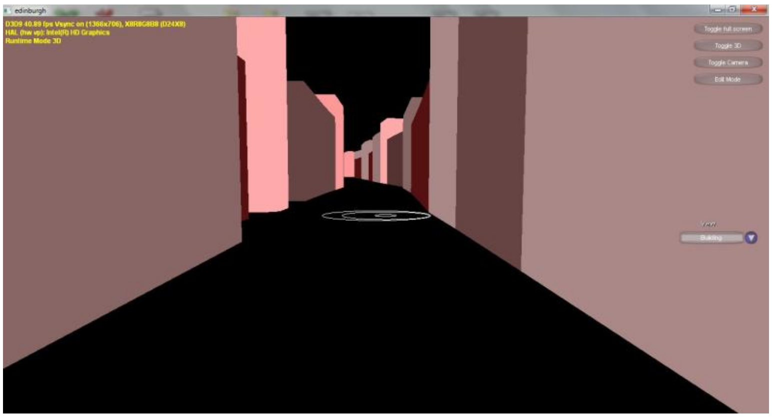

Figure 3: A visualization of the system’s representation of the user’s view.

2. Demo scenario

In order to develop and test the system, we have developed a simulation environment in which we can simulate user movement, either by manual input on a map, or by means of a fully automatic simulated user. We will demo both these functionalities. If possible, we will also allow conference attendees to try the system live in the near vicinity of the conference venue.

References

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

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

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

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