Natural Language Navigation Support in Virtual Reality

Jeroen van Luin, Anton Nijholt, Rieks op den Akker

University of Twente
Centre of Telematics and Information Technology
PO Box 217, 7500 AE Enschede, The Netherlands
{vanluin, anijholt, infrieks}@cs.utwente.nl

ABSTRACT

We describe our work on designing a natural language accessible navigation agent for a virtual reality (VR) environment. The agent is part of an agent framework, which means that it can communicate with other agents. Its navigation task consists of guiding the visitors in the environment and to answer questions about this environment (a theatre building). Visitors are invited to explore this building, ask questions and get advice from the navigation agent. A 2D map has been added to the environment so that visitors can make references to the locations and objects on this map, both in natural language and by clicking with the mouse, making it a multimodal system with cross-modality references.

1. INTRODUCTION

We have designed and built a VRML version of a theatre according to the drawings of the architects [3]. In the theatre we have added multiple agents, with the hostess 'Karin' being the main agent. She is standing behind an information desk and knows a lot about the performances that take place and the performers that will perform in the real theatre. Visitors can ask Karin questions in a natural language and Karin will access the performance and performers databases and try to extract and formulate answers. When the virtual world was made accessible to the audience, the need for an other agent emerged to solve problems like: "To whom do we address our questions about the environment itself?" and "To whom do we address our questions about how to continue, where to find other visitors or where to find domain-related information?". We learned from reactions of visitors that they had problems with navigating through the virtual world as well. At this moment we are following different approaches to solve these problems. The approaches are related and can be integrated as all of them are agent-oriented and are oriented towards a common framework of communicating agents. In this paper we will describe how we build navigation intelligence into an agent.

2. NAVIGATION

We define navigation as "the process in which people control their movements using visual clues in the environment and artificial aids such as maps, to reach their target without getting lost" [4]. Virtual worlds have many of the navigation problems that exist in the real world. However, virtual worlds are often considered more difficult to navigate than real worlds. This is due to the fact that most virtual worlds are less detailed, use no or different laws of physics and contain less visible clues that can be used as landmarks during navigation. This causes visitors of virtual environments to fail in getting a good overview of the environment. A problem we discovered while using the virtual theatre is that moving through the world using the VRML browsers mouse or keyboard interface is difficult and unnatural. Especially moving around objects turned out to be quite a struggle. These problems, causing desorientation and failure to find new or known locations will lead to unsatisfied and frustrated visitors who will stop using the environment.

3. AGENT AND MAP BASED NAVIGATION SUPPORT

As mentioned, the problems can be roughly divided into two categories: loss of overview and failure to navigate. Both problems also exist in navigating through the real world and many solutions have been tried to solve them. For instance, to solve the problem of loss of overview, we can use a map. To solve the problem of wayfinding in an unfamiliar area, we can use a

guide. The combination of a map and guide makes it possible to point out objects and locations on the map and refer to them when communicating with the guide.

We have implemented the combination of these two solutions in our virtual theatre to see if they would work in VR as well. The reader is referred to [2] for observations on user preferences for navigation support. Our work is in progress, meaning that the system is there, but no effort has been done to add 'graphic sugar' to the layout and the integration of the different windows that are used.

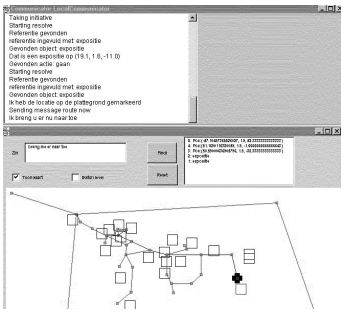

Figure 1: Floor map and agent window

In figure 1 we display the current floormap and the agent window and in figure 2 a view on part of the virtual world. In this view the user is looking at the stairs going to the next floor of the building. The visitor can ask questions, give commands and provide information when prompted by the agent. This is done by typing natural language utterances and by moving the mouse pointer over the map to locations and objects the user is interested in. On the map the user can find the performance halls, the lounges and bars, selling points, information desks and other interesting locations and objects.[1] The current position of the visitor in the virtual environment is marked on the map as well, allowing the visitor to check his position on the map while moving in VR. When using the mouse to point at a position on the map, both the user and the navigation agent can refer to the object or location pointed at.

4. ABSTRACT WORLD

Our visitors are able to interpret the objects and locations they see in the virtual world. However,

[1]Interestingly, one of these objects is a board in the virtual world displaying a map with chair positions in a performance hall. When the visitor clicks on a chair on this VR map, she is teleported to this chair to get a view at the stage.

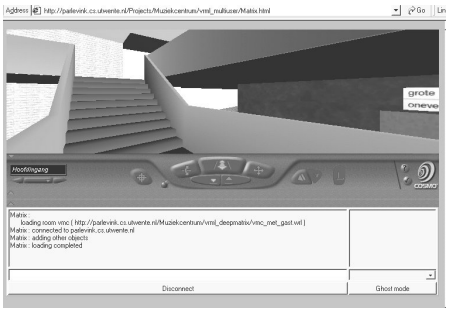

Figure 2: Visitor has been brought to the stairs

our navigation agent lacks such skills. To enable him to deal with the virtual world, we have made an abstract representation of the theatre. The agent has access to three databases. The first database contains information about physical objects that are present in the theatre, such as name, location, aliasses, adjectives, size, etc. The second database contains information about 'imaginary objects', i.e. objects without a physical 'body', such as performances. The third database is a general knowledge base, containing knowledge about relations between objects, such as chairs, tables and cupboards all being furniture. At this moment, the physical objects in the database have been entered by hand. When we have translated our theatre from VRML to Java3D, we will have the database automatically filled with the objects in the virtual world.

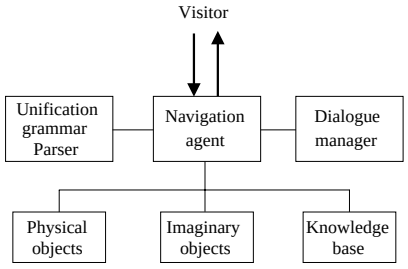

Figure 3: System layout

5. NATURAL LANGUAGE ACCESS

As mentioned, the navigation agent can be accessed using natural language. We have annotated a small corpus of example user utterances that may appear in navigation dialogues. The utterances were chosen based on a set of use cases that were made during the design of the navigation agent. The corpus contains two types of utterances. The first type are full sentences which contain complete questions and commands. Examples of full sentences are: "What is this?", while the visitor points at an object on the map,

or “Is there an entrance for wheel chairs?” or “Bring me to the information desk.”. The second type of utterances are short sentences, that the visitor can use when reacting on a question or remark from the navigation agent. Examples of short sentences are “No, that one.” or “Karin.” We use Treebank software to induce a grammar from the annotated corpus[1]. The grammar has unification rules to compose semantics based on the semantics of underlying non-terminals and the semantics of the terminals, which can be found in a lexicon.

S → *NP VP*
<S sem main> = <NP sem>
<S sem verb> = <VP sem>

NP → *Det N*
<NP sem det> = <Det sem>
<S sem main> = <N sem>

VP → *V*
<VP sem> = <V sem>

Figure 4: Example grammar

Figure 4 shows an example grammar with semantic unification rules for a sentence that consists of a *noun phrase (NP)* followed by a *verb phrase (VP)*, in which a NP consists of a *determiner (Det)* and a *noun (N)*. A VP can only have a single *verb (V)*. The Det, N and V are terminals and their semantics can be found in a (seperate) lexicon file. The next section will explain how this grammar is used by a unification type parser. Using a corpus based grammar has the advantage that new sentences can easily be added to the system by adding them to the corpus and repeating the induction step.

6. FROM UTTERANCE TO ACTION

When the utterance of the user is received by the navigation agent, the utterance is sent to our unification parser Demosthenes. The parser uses the corpus based grammar to find one or more semantic representations of the utterance. For example, if the sentence “The man whistles” is parsed using the grammar in figure 4, the following representation would be generated:

$$\begin{bmatrix} \text{stype:} & \text{decl} \\ \text{main:} & \begin{bmatrix} \text{det:} & \text{the} \\ \text{main:} & \text{man} \end{bmatrix} \\ \text{verb:} & \text{whistles} \end{bmatrix}$$

Figure 5: Semantic representation

where *stype* refers to the sentence type, in this case a declarative sentence. This simple example only has one possible parse, but when more complex sentence-parts like for example *prepositional phrases* are introduced, the number of possible parses will increase rapidly.

If more than one representation has been found, it is possible to use knowledge of the domain to make an educated guess which representation represents the utterance best, but this has not been implemented yet. At this moment, if one or more representations are found, the first representation is returned. If no representations are found, the parser returns a representation of the longest part of the utterance that could be parsed. The chosen representation is sent to the dialogue manager who tries to create an action object based on the information in the representation. Action objects can contain the action to undertake as well as the object(s) and location(s) used in the action. The dialogue manager uses an abstract representation of the virtual world and a knowledge base to match reference words in the utterance to objects and locations in the world. The abstract representation contains the objects and locations from the real world with their specific names, aliases and adjectives. The knowledge base contains general knowledge about objects, like chairs, tables and cupboards all being furniture.

- Receive visitor utterance
- Analyze input
- If analysis contains enough information
  - Create action object
  - Execute action object
- Else
  - Try to get the missing information
  - If unsuccesful, ask a question

Figure 6: Highest level algorithm

Whenever more than one object would fit the description in the utterance, the dialogue manager automatically assumes the one closest to the visitor is meant.

If the information in the representation is enough to create an action object with all necessary fields filled in and all references could be solved, the action object is executed. The object is then stored on the action object stack, which contains all the action objects generated during this particular visit of the user. If the information in the representation is not enough

to correctly determine the visitors wish, three agents start to work together: the navigation agent, the dialogue manager and the CosmoAgent. The latter can 'talk' to the Cosmo VRML-browser using an EAI interface to find out information about the location and surroundings of the visitor and move the visitor through the virtual world.

Together, they try to find out what information is missing and how they can get that information. They first try to use information that is already available, like the history of the dialogue. If that information isn't enough to fill in all the necessary information, the navigation agent will take over the dialogue initiative and the visitor will be asked an intelligent question. The question is intelligent in the sense that the information that *could* be extracted from the utterance is 'remembered' and used in the creation of the question. For example, if the navigation agent did find out the visitor wanted to be brought to a certain place, but couldn't find out whereto, the question asked will not be "What do you want?" but "Where do you want to go to?".

The visitor can respond in two ways. First, the visitor can answer with a new command or question in which the missing information is entered or in which the misspelled information is corrected. However, the visitor can also respond with a short phrase, in which he only gives the missing or misspelled information. The dialogue manager will then enter the new information in the remembered action object.

Not yet implemented is the possibility that the CosmoAgent can find out what is in the eyesight of the visitor. This information can be used to allow and disambiguate references to objects that are visible in the VR world.

7. NAVIGATION DIALOGUES

To illustrate the way our multi-modal navigation assistence works, we give three examples of dialogues. The first example shows the way the agent responds when confronted with an error in the input. The second example shows the way the multi-modality is used, and the third example shows how the visitor can refer to objects mentioned in earlier parts of the dialogue.

1. *Visitor:* How do I get to the information deks?
   Agent: Where did you want to go?
   Visitor: The desk.

2. *Visitor:* Can you bring me to the information desk?
   Agent: I have marked the position on the map. Now I will bring you there.
   Visitor: [Clicks on an object on the map] What is this?
   Agent: That is an exposition.

3. *Visitor:* Where is it?
   Agent: You can find it in the lounge.
   Visitor: Let's go there.
   Agent: I will bring you there.

8. CONCLUSIONS AND FUTURE RESEARCH

The prototype navigation agent which we discussed here is only a first step in navigation support. It is certainly not our final solution in assisting the visitors of our virtual environment. In the next phase of research we need to concentrate on the communication with other agents that are available in the virtual theatre. How can we take care that a visitor's question reaches the appropriate agent? How can we model the history of interaction in such a way that different agents do not only know about their own role in this interaction but also about that of the others? Unlike other environments, our environment allows the investigation of communication between active and passive agents that inform the visitor about the possibilities and the properties of an information-rich virtual environment.

9. REFERENCES

[1] H.W.L. ter Doest, Towards probabilistic unification-based parsing, Neslia Paniculata publishers, Enschede, The Netherlands, 1999

[2] K. Höök, et al. Towards a framework for design and evaluation of navigation in electronic spaces. *Persona Deliverable* for the EC, 1998

[3] A. Nijholt, and J. Hulstijn, Multimodal interactions with agents in virtual worlds. In: *Future Directions for Intelligent Information Systems and Information Science*, Springer Physica-Verlag, 2000.

[4] J. Zwiers, B. van Dijk, A. Nijholt and R. op den Akker, Design issues for navigation and assistance agents in virtual environments. In *TWLT 17 Learning to behave*, pages 119-132, 2000