

# The Virtual Guide: A Direction Giving Embodied Conversational Agent

Mariët Theune, Dennis Hofst, Marco van Kessel

Human Media Interaction, University of Twente, Enschede, The Netherlands

{m.theune | d.h.w.hofst}@ewi.utwente.nl

## Abstract

We present the Virtual Guide, an embodied conversational agent that can give directions in a 3D virtual environment. We discuss how dialogue management, language generation and the generation of appropriate gestures are carried out in our system.

**Index Terms:** dialogue management, language generation, embodied conversational agents

## 1. Introduction

We have developed an embodied Virtual Guide<sup>1</sup> that can give route directions in the Virtual Music Centre (VMC), a 3D virtual environment replicating the music theatre in our home town. When navigating through the VMC, the user can approach the Virtual Guide to ask for directions. Currently the Virtual Guide is located at the reception desk of the VMC (see Figure 1), but she can be situated anywhere in the building.

The first part of the interaction between the Virtual Guide and the user consists of a natural language dialogue in which the **dialogue management** module tries to find out the user's intended destination. This may involve subdialogues, in which either the Guide or the user asks the other for clarification, and the resolution of anaphoric expressions (e.g., *How do I get there?*).<sup>2</sup> When the user's destination has been established, the **language generation** component of the Virtual Guide generates a natural language route description consisting of a sequence of segments that are mostly expressed as "point+direction" combinations [1], i.e., a turn direction combined with a description of the location where this turn is to be made, specified in terms of a landmark. For example, *You go left at the information sign*. Currently the route description is given in the form of a monologue that cannot be interrupted. Finally, the **gesture generation** component extends the generated text with tags associating the words in the route description with appropriate gestures. The marked-up text is sent to the animation planner, which actually generates the required animations in synchronization with text-to-speech output, resulting in a multimodal route description. Below, the three main components of the Virtual Guide are described in more detail.

## 2. Dialogue management

The Virtual Guide allows for multimodal dialogues, using text and speech as well as nonverbal input and output modalities. For instance, the user can use speech combined with mouse input by pointing at a 2D map of the VMC and asking *What is this?* In its turn, the Virtual Guide can produce speech accompanied with gestures and indicate locations and routes on the map. Here, we briefly discuss how the dialogue management

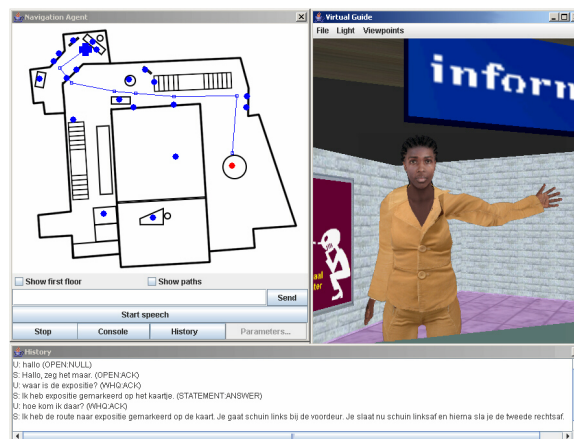


Figure 1: The Virtual Guide.

module processes and responds to the user's utterances. More details can be found in [2].

### 2.1. Dialogue Acts

The first step of the dialogue manager is to assign a dialogue act to each user utterance. A dialogue act consists of a forward tag and a backward tag, based on the DAMSL (Dialog Act Markup in Several Layers) scheme [3]. A forward tag is in direct relation with the user's utterance and concerns the future of the dialogue, whereas the backward tag says how the utterance relates to the last utterance in the current subdialogue. In addition to these tags, a dialogue act contains a deep parse of the utterance annotated with information obtained from the fusion and anaphora resolution steps. The following is an example dialogue act for the utterance *What is this?*

forward:	WHQ								
backward:	NULL								
parse:	<table border="1"> <tr> <td>type:</td> <td>WHQ</td> </tr> <tr> <td>subject:</td> <td>this</td> </tr> <tr> <td>object:</td> <td>what</td> </tr> <tr> <td>verb:</td> <td>to be</td> </tr> </table>	type:	WHQ	subject:	this	object:	what	verb:	to be
type:	WHQ								
subject:	this								
object:	what								
verb:	to be								

It contains the forward tag WHQ, denoting a wh-question, and the backward tag NULL, which indicates this is the first utterance in the dialogue and thus has no previous dialogue act to relate to.

All user utterances are fully parsed using a Dutch unification grammar, resulting in zero or more hypotheses. A fusion module searches for deictic expressions and merges them with

<sup>1</sup> Accessible online via <http://wwwhome.cs.utwente.nl/~hofst/dialogue>.

<sup>2</sup> The actual language of the Virtual Guide is Dutch, but for ease of reading all examples are given in English.

any co-occurring pointing gestures. It also binds non-anaphoric noun phrases to objects in the VMC. The enhanced parses are sent to the dialogue act classifier, which uses rules and the dialogue history to generate one or more dialogue acts for each parse. First it analyzes the parses and generates possible pairs of forward and backward tags. For example, the utterance *Could you take me to the great hall?* – at the surface a yes/no question – results in the forward tag of a request. The next step is to select the most likely pair using the dialogue history. For every possible forward tag, the dialogue act classifier holds an ordered list of preferred backward tags that can follow it. For example, after a WHQ forward tag (a question), the preferred backward tag is ANSWER.

## 2.2. Reference resolution

After the fusion module has bound non-anaphoric expressions to objects in the VMC, the reference resolution does this for anaphoric ones and completes the analysis of the user input. It uses a modified version of Lappin and Leass's reference resolution algorithm [4] that assigns weights to references, based on a set of salience factors such as recency. The following example illustrates various ways of referring to objects:

U1: (pointing at the coffee bar) *Is this the cloak room?*  
 S1: *No, that's the coffee bar. The cloak room is over there.*  
 U2: *Could you take me there?*

The user's first verbal reference, *this*, is a deictic expression that is resolved by the fusion module, which links it to the coffee bar that the user is pointing at. It invokes the first referent in the dialogue model. The fusion module also resolves the user's next reference, *the cloak room*, by matching the noun with the properties of the objects in the virtual environment. In its response, the system refers twice to the coffee bar (*that, the coffee bar*) and once to the cloak room. These references are also added to the dialogue model. Finally the user's anaphoric reference *there* has two possible antecedents: the coffee bar and cloak room. Of these two, the latter has a higher salience value because it was most recently mentioned, leading the system to the conclusion that *there* refers to the cloak room.

## 2.3. Subdialogues

When the dialogue manager receives a dialogue act of the user, it tries to unify the parse with a so-called 'action template', and then creates the appropriate action to be performed. The action may require an argument (e.g., a location to be shown on the map), which is taken from the user's dialogue act. The new action is put on top of the action stack storing the system's planned actions. When it is the system's turn, it takes an action from the stack and executes it. If the action involves the utterance of a dialogue act, this dialogue act is added to the dialogue history, and if it includes an object reference, the dialogue model is updated.

In addition to the action stack, the dialogue manager maintains a subdialogue stack (the stack of "questions under discussion") that keeps track of the current dialogue structure. Either the system or the user can take the initiative to start a new subdialogue by asking for clarification instead of directly answering a question. Consider the following dialogue:

U1: *How do I get to the cloak room?*  
 S1: *Which cloak room are you looking for?*  
 U2: *Are there more cloak rooms?*  
 S2: *Yes, there's one upstairs and one downstairs.*

U3: *I meant the cloak room downstairs.*

S3: *You go left at the coffee bar, then ...*

The first user utterance is a dialogue act with forward tag WHQ and backward tag NULL. The fusion agent binds the noun phrase *the cloak room* with the two cloak room objects that are present in the building. As an appropriate response to this dialogue act, the dialogue manager finds a TellRoute action telling the user the way to the argument object. However, since there is no single object available, the system starts a subdialogue by asking which of the two cloak rooms is meant. This new subdialogue is added to the subdialogue stack. The current subdialogue (containing S1) is on top of the stack and the main dialogue (containing U1) is at the bottom. The originally intended TellRoute action is put on the action stack, with a flag saying that it is not yet executable.

Then the user starts yet another subdialogue by asking if there are more cloak rooms. This is a dialogue act with forward tag YNQ and backward tag HOLD. The noun phrase *cloak rooms* is again bound to both cloak rooms. In response, the system explains about the cloak rooms: a dialogue act with forward tag STATEMENT and backward tag ACCEPT. Meanwhile, the TellRoute action is still on the action stack.

Finally the user chooses the downstairs cloak room. The dialogue act classifier decides that the user's utterance is a dialogue act with forward tag STATEMENT and backward tag ANSWER, which refers to S1, the last dialogue act in the enclosing subdialogue. This means that the user utterance ends the current subdialogue, so the subdialogue U2-S2 is taken off the stack and on top of the stack is now the subdialogue S1-U3. The fusion agent can bind the noun phrase *the cloak room downstairs* to the correct object in the VMC, which is then filled in as the missing object parameter in the action currently on top of the action stack: the TellRoute action. With this parameter substitution the action is made executable, and the Virtual Guide starts explaining the route to the user.

## 3. Language generation

The Virtual Guide uses multiple output modalities in a route description: the route is projected on the map of the VMC (see Figure 1) and also described in words and gestures. Here we discuss the generation of the verbal part of the route description.

### 3.1. Turns and landmarks

The input to the language generation component consists of the shortest path from the starting point to the destination, specified as a list of *markers*: 3D coordinates in the VMC. The path is computed based on a network of predefined paths in the virtual environment. An example path, consisting of the list (*a, b, d, p, f, m*), is shown in Figure 2. Two connected markers form a segment, and the first step of the language generation algorithm is to calculate the angle between each pair of subsequent segments. Based on these angles, a turn direction is determined for each marker (straight ahead, sharp left, left, etc.) and added to the path. Multiple subsequent markers associated with the direction 'straight ahead' are filtered out. For the example path, this would happen to markers *b* and *d*, whereas markers *p* and *f* will be associated with a sharp right and left turn respectively.

The next step is to describe the locations where the turns are to be made in terms of landmarks, i.e., salient objects or other reference points. In buildings, typical landmarks include stairs, hallways and signs. The selection of potential landmarks by the Virtual Guide is done using a 'cylinder' vision. The path

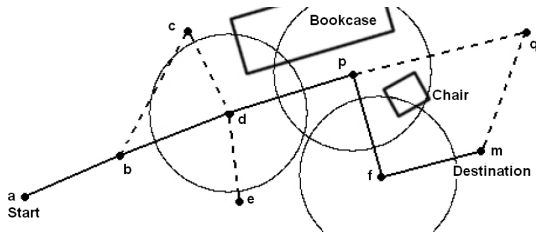


Figure 2: An example path.

marker is used as an origin from which a cylinder-shaped area is drawn that simulates the user's viewpoint as he or she is standing at the location of the marker. All objects located within this cylinder can potentially be used as a landmark. The generator then has to decide which of these objects is best suited for use in the route description. This is done by reducing the set of potential landmarks to one based on their values for properties such as size, movability (immovable objects are more reliable landmarks than movable objects), colour and shape. The algorithm goes through this list of properties one by one, each time reducing the set of potential landmarks to those objects that (a) have a preferred value for the current property or (b) have an optimally distinguishing value for that property, i.e., a value that distinguishes them from the highest number of alternative landmarks. If, after going through all the properties, the set has not been reduced to one landmark, the algorithm simply chooses the landmark that is closest to the turn location.

We will explain the working of the landmark selection algorithm using an example. Assume that the following objects are potential landmarks:

- Object 1: big, movable, brown, round
- Object 2: big, movable, white, round
- Object 3: big, movable, brown, round
- Object 4: big, movable, brown, square
- Object 5: medium, movable, white, square
- Object 6: big, movable, green, square

The algorithm first considers the size of the objects. Since bigger objects are preferred over smaller ones, Object 5 is ruled out in this first round. For the next property, movability, no objects are ruled out since they all have the same value for that property. The next property is colour. This has no *a priori* preferred value, so here the value is chosen that rules out the highest number of potential landmarks. This is the colour green, which rules out all objects except Object 6, which is consequently selected.

Returning to our example in Figure 2, we see that here the bookcase and the chair are potential landmarks for marker *p*, as they are the only objects within its cylinder. In this case the bookcase would be used as a landmark because of its size compared to the size of the chair. Marker *f* get the chair as its landmark, even though the chair is also near marker *p*. (Currently the route generator does not check if objects are closer to other markers than the one that needs to be described.) The information about the chosen landmarks is added to the path specification, which then looks as follows: (*a*, <*p*, sharp right, (bookcase, large)>, <*f*, sharp left, (chair, green)>, *m*).

### 3.2. Surface Realisation

The generation of the actual route description is done using a surface realiser based on Exemplars [5]. Simply speaking,

exemplars are predefined sentence templates associated with a condition that defines their applicability in terms of tests on the input. They are organized in a specialization hierarchy, where more specialized exemplars can augment or override the more general ones they specialize. For example, some of the exemplars used for the Virtual Guide only require a direction to be known, while others also need a landmark to refer to. The last sentence of a route description is created with the use of a specific set of exemplars that emphasize the destination has been reached. In order to achieve some variation within the generated route descriptions, at each level of the specialization hierarchy a number of equivalent exemplars is available, from which a random choice is made. For example, *Turn <direction> at <landmark>* and *At <landmark>, go <direction>*.

The generation of the output text proceeds in two stages. A first version of the route description is generated using a collection of standard sentence structures, as described above. In a second round, this initial description is revised by randomly aggregating some sentences and adding cue phrases such as *then* and *after that* to them in order to make the generated text more varied and coherent.

## 4. Gesture generation

To generate appropriate gestures to accompany the verbal route description, the generated text is extended with tags associating the words in the route description with different types of gestures. The marked-up text is sent to the animation planner, which actually generates the required animations in synchronization with text-to-speech output.

### 4.1. Choosing gestures

The way gestures are added to the route description is similar to that of the BEAT system [6], in the sense that it first creates a collection of gesture suggestions based on information about keywords in the sentence and then selects the most appropriate gestures from this set to accompany the verbal route description. For example, references to objects can be accompanied by (1) a pointing gesture to the absolute location of the object ('objective viewpoint'), (2) a pointing gesture to the location of the object relative to the position of a person who is walking the route ('subjective viewpoint'), (3) an iconic gesture, reflecting the shape of the object, and (4) a simple up-and-down 'beat' gesture, which has no inherent meaning but only adds emphasis to what is said. After the full route description has been generated, a selection from all possible gestures is made, based on a weighted randomization. The weights are currently determined by hand; a more realistic weighted system might be determined empirically based on the results of video analysis.

Currently the gesture weights are set in favour of pointing gestures made from an objective viewpoint. This was done based on the results of a small experiment, where 32 participants judged three movies in which the Virtual Guide gave the same route description, each time with different gestures. In the first movie, the Guide made gestures only from a subjective viewpoint, and in the second movie, only from an objective viewpoint. In the third, the Guide made no gestures at all. Of the participants, 68% said that the movie with the objective viewpoint gestures was the best, and 82% said that the movie without gestures was the worst. The latter finding clearly shows the added value of using gestures in an embodied agent.

A possible explanation for the users' preference for objective viewpoint gestures is that the Virtual Guide makes objec-

tive gestures with her own body as a reference. This means that from the user's perspective the Guide's gestures are mirrored; i.e., when the Guide points left, this is actually to the user's right and *vice versa*. In real direction giving, speakers also tend to gesture from their own perspective, but the mirroring effect is often diminished because speaker and hearer turn so that they largely share the same perspective. In [7] we investigated if this would also be a feasible strategy for the Virtual Guide. We presented users with route descriptions from different orientations, and found that in spite of the mirrored gestures, having the speaker face the hearer was equally effective as, and more natural than, adapting the speaker's orientation to that of the hearer. Therefore we decided to have the Virtual Guide face the hearer as well. However, since this experiment was performed with a human direction giver, further experimentation is still required, this time using the Virtual Guide.

#### 4.2. Animation planning

Finally, the gestures are animated and synchronized with the speech output by a modified version of the animation planner developed by [8]. The input for the planner is a route presentation script, specified in a multimodal mark-up language that is illustrated here using a sentence from the script used for our test movies (see Section 4.1). Shown here is a fragment containing an objective viewpoint gesture:

```
<Channel name="Verbal">
<Verbal>Once upstairs, the door to the balcony is to your
<SyncPoint id="P4"/>right</Verbal>
</Channel>
<Channel name="Gesture">
<Deictic stroke="P4" location="(-1.0, 5.9, -30.0)"/>
</Channel>
```

This (somewhat simplified) example shows that one mark-up channel is reserved for verbal utterances and one for gestures. In the verbal channel, synchronisation points are created that are used as starting points for the gestures in the gesture channel. Utterances specified in the verbal channel are sent to a speech synthesizer which not only pronounces the text but also returns an estimation of the durations of the phonemes in the utterance. This information is used to synchronize the gestures with their associated words.

As illustrated by the example above, the pointing gestures that the Virtual Guide makes from an objective viewpoint are generated dynamically, using the location of the target object as input parameter. Iconic gestures, however, are generated using canned animations. An example is a horizontal tube-like gesture that can be used in references to corridors and tunnels. For a more sophisticated approach, see the work by [9] who describe the dynamic planning of novel iconic gestures by NUMACK, an embodied conversational agent that functions as a virtual guide for the Northwestern University campus.

### 5. Discussion

The Virtual Guide has been implemented and is fully functional, but it still has its flaws. For example, the dialogues that can be carried out with the Guide are still somewhat limited. To make the dialogue manager more robust, its grammar and lexicon are currently being extended with more rules and more synonyms for relevant concepts, so that (for example) the user cannot only ask for the *toilet* but also for the *ladies' room*.

When generating the route description, the objects selected

as potential landmarks tend to be things like furniture and paintings. However, landmarks in route descriptions produced by human speakers more often correspond to structural parts of a building such as hallways and corridors, which are currently not available to our landmark selection algorithm. In addition, route generation should be better integrated with dialogue management to allow for interruptions during the description, for example to ask clarification questions.

With respect to gesture generation, the current design where the verbal description is generated first and gestures are added later prevents the generation of utterances such as *Go left there*, where the verbal reference is dependent on an accompanying gesture for its interpretation. To achieve such complementary word and gesture combinations, again a more integrated approach is necessary. Ideally, language and gesture generation should be fully intertwined.

### 6. Acknowledgements

We thank R. op den Akker for creating the grammar used in the Virtual Guide, M. Bouman and R. Korthuis for their work on the language generation component, and M. de Jong for extending the dialogue manager. This work was carried out within the NWO project ANGELICA (grant no. 632.001.301).

### 7. References

- [1] R. Dale, S. Geldof, and J. Probst, "Using natural language generation in automatic route description," *Journal of Research and Practice in Information Technology*, vol. 37, no. 1, pp. 89–105, 2005.
- [2] D. Hofs, R. op den Akker, and A. Nijholt, "A generic architecture and dialogue model for multimodal interaction," in *Proceedings of the 1st Nordic Symposium on Multimodal Communication*, Copenhagen, Denmark, 2003, pp. 79–91.
- [3] M. Core and J. Allen, "Coding dialogs with the DAMSL annotation scheme," in *Working Notes of the AAAI Fall Symposium on Communicative Action in Humans and Machines*, Boston, MA, 1997.
- [4] S. Lappin and H. Leass, "An algorithm for pronominal anaphora resolution," *Computational Linguistics*, vol. 20, no. 4, pp. 535–561, 1994.
- [5] M. White and T. Caldwell, "EXEMPLARS: A practical, extensible framework for dynamic text generation," in *Proceedings of the Ninth International Workshop on Natural Language Generation*, 1998, pp. 266–275.
- [6] J. Cassell, H. Vilhjálmsón, and T. Bickmore, "BEAT: the Behavior Expression Animation Toolkit," in *Proceedings of SIGGRAPH '01*, Los Angeles, CA, 2001.
- [7] M. Evers, M. Theune, and J. Karreman, "Which way to turn? Guide orientation in virtual way finding," in *Proceedings of the ACL Workshop on Embodied Language Processing*, Prague, Czech Republic, 2007.
- [8] H. van Welbergen, A. Nijholt, D. Reidsma, and J. Zwiens, "Presenting in virtual worlds: Towards an architecture for a 3D presenter explaining 2D-presented information," *IEEE Intelligent Systems*, vol. 21, no. 5, pp. 47–53, 2006.
- [9] J. Cassell, S. Kopp, P. Tepper, K. Ferriman, and K. Striegnitz, "Trading spaces: How humans and humanoids use speech and gesture to give directions," in *Conversational Informatics*, T. Nishida, Ed., in press.