Landmarks in Navigation Instructions for a Virtual Environment

Kristina Striegnitz
Union College
Schenectady, NY, USA
striegnk@union.edu

Filip Majda
Czech Technical University
Prag, Czech Republic
filip@majda.cz

Abstract

This paper describes a system which automatically generates instructions to guide a user through a virtual environment. The directions use landmarks to indicate important points along the route. The system participated in the NLG research challenge on “Giving Directions in Virtual Environment” and the results from that challenge are discussed.

1 Introduction

When humans give walking directions they use landmarks for a variety of purposes. Most importantly they are used to indicate where to turn, to confirm that the recipient is still on the right track, or to locate other landmarks (Denis, 1997). In urban settings, route descriptions with landmarks have also been shown to be more effective than route descriptions without landmarks (Michon and Denis, 2001; Tom and Denis, 2003).

In this paper, we describe an NLG system that uses a landmark based direction giving strategy in an online environment. The system participated in the GIVE-1 challenge (Giving Directions in Virtual Environments, (Koller et al., 2009)), which is a research challenge for the natural language generation (NLG) community. The task of this challenge is to develop an NLG system that can guide a user through a virtual environment to solve a treasure hunt.

Our system switches between the landmark based strategy (e.g., *go to the green button next to the lamp*) and a path based strategy (e.g., *go 5 steps forward*) depending on the user’s behavior and features of the surrounding environment. It was one of the best two in GIVE-1 in terms of task success (Byron et al., 2009). It also performed robustly across the three (quite different) evaluation worlds, and was not sensitive to the user’s English proficiency, gender, computer expertise, or weekly hours of video game play.

In the next section, we will introduce the landmark based strategy and the path based strategy by means of an example and describe the factors that determine which strategy is used. We will then explain the architecture of our system in more detail, in Section 3. Section 4 discusses the results of the GIVE-1 evaluation. Finally, we conclude in Section 5.

2 Example Session

To illustrate the use of landmark based instructions and path based instructions in our system, we are now going to go through a short example interaction between our system and a user. Figure 1 shows the system instructions and explains the user’s reactions; Figure 2 shows what the user sees at some key points in the interaction.

By default, our system uses landmark based instructions. If the object that needs to be manipulated next is visible to the player, the system describes the object and directs the player to move toward it. The action that the player needs to perform on this object can be aggregated to the navigation instruction. For example, in the scene shown in Figure 2(a) our system produces utterance S1 of Figure 1. When instructions are aggregated as in utterance S1, parts are repeated as they become effective; e.g., S2 utterance.

If the object that needs to be manipulated next is not visible, there are two options. First, another object along the way may be used as a landmark as in *Go to the chair, and then turn right.* If such an object is not available either, the system uses a path based instruction like utterance S5. As soon as the target object becomes visible, the system switches back to landmark based instructions like in S6.

Path based instructions are also used if the user seems to have problems following the landmark

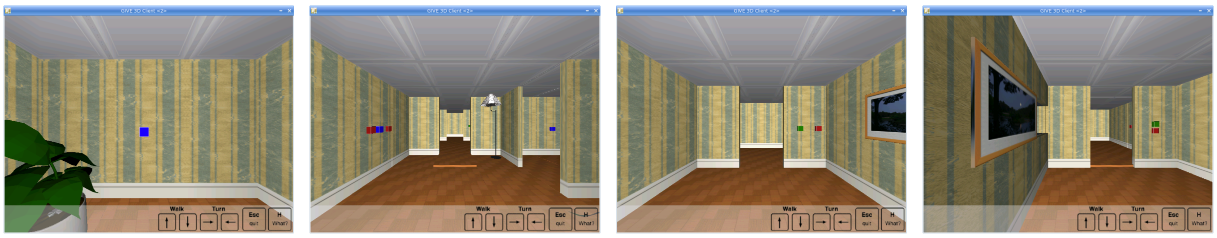

Figure 2: Player views of a GIVE game world.

[U's view: Figure 2(a)]
S1: Go to the blue button, and then press it. *[U moves forward. When he is directly in front of the button, S says]*
S2: Ok. Now press the blue button. *[U presses the button.]*
S3: Excellent. Turn right. *[U turns.]*
S4: Turn right again. *[U turns. U's view: Figure 2(b)]*
S5: Move forward 4 steps and then turn right. *[U moves forward. As soon as he has enters the next room (U's view: Figure 2(c)), S says]*
S6: Go to the red button and then press it. *[U goes to the button. He approaches the button from the left. When he is in front of the button, S says]*
S7: Ok. Now turn left. *[U turns.]*
S8: Press the red button. *[U presses the button.]*
S9: Excellent. Turn right. *[U turns.]*
S10: Turn right again. *[U turns. U's view: Figure 2(d)]*
S11: Go past the picture to the trophy. *[U presses 'H'.]*
S12: Move foward 2 steps, and then turn left. *[U moves forward and turns.]*
S13: Take the trophy.

Figure 1: Example interaction

based instructions, i.e., if the user presses 'H' or if he does not make progress toward the target object.

3 Architecture

We now describe how the strategy illustrated in the previous section is implemented.

The GIVE infrastructure (Koller et al., 2009) provides an abstract class `NlgSystem`, that all GIVE NLG systems have to implement and which manages the communication with the 3D client. It consists of a number of callback methods used to inform the NLG system of user actions and one method for sending a string to be displayed to the user by the client. Our system's contributions are triggered as reactions to user actions or because there has not been a user action for a certain amount of time. Based on the last user action and the currently visible portion of the game world, our system decides whether to use the landmark based or the path based instruction mode (Subsection 3.1), then it plans what to say in the next utterance (Subsection 3.2), performs some aggregation (Subsection 3.3), generates descriptions for the objects mentioned in the instruction (Subsection 3.4), and finally realizes it as a string that gets sent to the client (Subsection 3.5).

3.1 Control

Instructions are sent in reaction to user actions; if users do not do anything for a certain amount of time, we send a message reminding them that they can use the 'H' button to ask for help.

By default, the system is in *landmark mode*. As soon as the **user enters** the evaluation world, the system requests a plan from an off-the-shelf planner, SGPLAN 5 (Hsu et al., 2006), and then triggers content planning.

When the **user manipulates an object**, the system checks whether this is the next object manipulation task that the user was supposed to do. If so, the completed steps get deleted from the plan, *landmark mode* is activated (if the system isn't already in *landmark mode*) and content planning is called. Otherwise, the system performs the following **error handling** steps: It sends the message "No, wait. That was wrong. Let me make a new plan for you." to the client and requests a new plan from the planner. It then goes into *path mode* and starts content planning.

When the **user moves forward** and ends up at the current target landmark or at the location where the next object needs to be manipulated, all plan steps up to the next object manipulation step get deleted from the plan. The system then calculates which turns are necessary (if any) in order to face the object and these turns are added to the plan. Then, the system goes into *landmark mode* and content planning is started.

In other cases where the **users moves forward or turns**, the system's behavior differs depending on the mode it is in. If the system is in *path mode*, it checks whether the movement agrees with the next plan step. If so, that step gets deleted from the plan. If this step is part of a sequence of similar steps (e.g., several steps moving forward) and not the last step in that sequence, the system does not send any output to the client. Otherwise, it starts content planning. If the user's action does not agree with the next plan step, the system performs the same error handling steps described above.

If the system is in *landmark mode*, user moves cannot be directly compared to the plan since the user may choose a different path to the target location than the planner. Instead it monitors that the user is making progress toward the target. As long as the user is making progress, the system does nothing. If the user stops making progress, the system goes into *path mode* and performs the error handling steps described above.

3.2 Content Planning

Content planning determines what to say next. The output is a sequence of simple messages. There are four main types of message: *turn*, *move*, *press*, *take*. The *press* and *take* message take an object as parameter; the *turn* message takes a direction (left or right) and optionally an object (where the turn should happen) as parameters; the *move* message takes a direction (forward or backward) and optionally an object (the end point of the movement) and a list of objects (objects that are passed a long the way) as parameters.

If the system is in *path mode*, the system selects all plan steps up to (and including) the first turn or object manipulation action and translates them into messages. This sequence of messages is passed on to aggregation.

In *landmark mode*, the system finds the next object manipulation action in the plan. If the object referred to in that action is visible to the player, a message instructing the user to go to that object is created. This message may also include a list of objects that are passed along the way. If the object that needs to be manipulated is not visible, the system starts looking backward through the plan to find another visible object along the way that could serve as a landmark. If no such object is found, the system behaves as it does in path mode.

3.3 Aggregation

Our system only uses a very simple aggregation mechanism. It takes the sequence of messages produced by content planning and replaces repetitions of the same message with one messages combining them: for example, five times "move forward" becomes "move forward five steps".

3.4 Referring Expression Generation

Referring expression generation is done using the algorithm and implementation described in (Areces et al., 2008). The only change we made is to add a step at the end which allows us to force the use of certain kinds of properties even if they do not contribute to distinguishing the object from its distractors (for example, we force the use of color if a color predicate is available).

The input for this algorithm is a set of individuals and a specification of their (unary) properties and relations to other individuals. We use all currently visible objects for this set. To be able to generate descriptions of groups of objects, we introduce additional individuals representing groups. In addition to the properties and relations retrieved from the discrete game world representation, we add some location information describing the position of objects with respect to the user and with respect to each other.

3.5 Realization

We use a simple template based realization mechanism.

4 Evaluation

In this section we present some statistical analyses of our system's performance (Section 4.1), describe what causes for fatal errors we can identify in the game logs (Section 4.2), and discuss these results (Section 4.3).

4.1 Statistical Results

In 73% of all games played with our system the user successfully gets the trophy. This task success rate is relatively stable across the different evaluation worlds and different users (see Figures 3 and 4). We did not find significant differences in task success rate that depended on the evaluation world, the user's gender, English skills, computer expertise[1], or how many hours per week the user

[1]We had so few participants at the lowest levels of computer expertise that we excluded those levels from the analy-

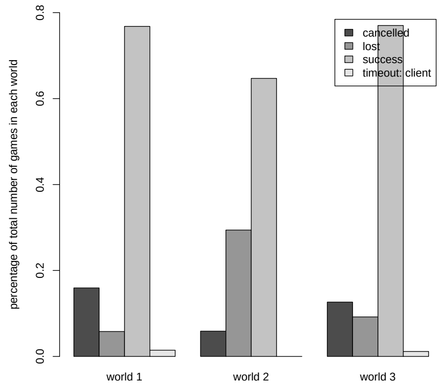

Figure 3: Task success rate in the different evaluation worlds.

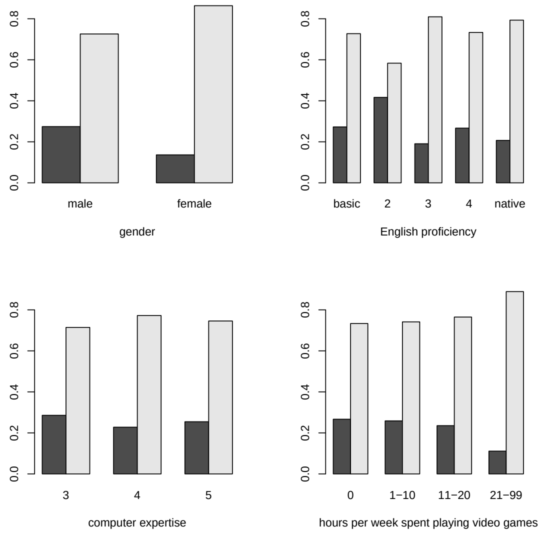

Figure 4: Task success rate across different demographic factors. Light gray: successful games; dark gray: unsuccessful games.

	mean	sd
number of actions	9.03	2.10
number of steps	117.49	32.83
number of instructions	80.29	26.00
seconds	175.15	58.42

Figure 5: Mean and standard deviation of the objective measures collected for successful users.

	world 1	world 2	world 3
mean	1.14	1.10	1.14
sd	0.21	0.27	0.22

Figure 6: Average ratio of the actual number of steps performed by the user and the minimum number of steps necessary to solve the task.

played video games. Compared to the other four system that participated in the GIVE-1 challenge, our task success rate was similar to Madrid's and (significantly) higher than that of the other three systems.

Figure 5 shows the other objective measures collected as part of the GIVE-1 evaluation: the number of object manipulation actions successful users performed, the number of overall steps (object manipulation and movement) successful users performed, the number of instructions they received from our system, and the time in seconds they needed to reach the goal.

Since the three evaluation worlds have different layouts and sizes, these objective measures cannot be directly compared between worlds. Therefore, Figure 6 also shows the ratio of the actual number of steps performed by the user and the minimum number of steps necessary to solve the task, which can be used as a measure of how many mistakes successful users made. Like for task success rate, there is no significant difference between the evaluation worlds.

Similarly, the objective measures are mostly robust with respect to the demographic data collected from the users. The only significant effect we find is that users with only basic English skills (the lowest level) needed significantly more time than other users.

Among the GIVE-1 participants, our system and Madrid's system guided the users to success with the fewest actions, i.e., with the fewest mistakes. However, our system produced more instructions than Madrid's to achieve this and

sis.

both our system and Madrid's were slower than Austin's in term of the time users needed to reach the goal.

Figure 7 summarize the subjective measures collected for our system. None of the measures is affected by the user's English proficiency. The only measure affected by the user's computer expertise is *task difficulty*: less experienced users rate the task as more difficult. And the only measures affected by the evaluation world are *choice of words* and *referring expressions*: in both cases the rating is lowest in world 3 and highest in world 1. Gender affects several measures: women find the task more difficult than men and find it harder to interpret the navigation instructions. They also rated our system as less friendly. On the other hand, a greater percentage of women than men finds that our system's instructions provide just the right amount of information at just the right time.

A comparison with the other systems participating in the GIVE-1 challenge in term of the subjective measures is not so easy since the subjective measure do not distinguish as clearly between systems as the objective measures. One thing that can be said is that the participants were relatively unhappy with our way of wording the instructions. Furthermore, our system's ratings for instruction clarity, especially for navigation instructions, informativity and timing are not in the top tier.

4.2 Error Analysis

We examined the game logs to find out why participants lost. In this section we describe what seem to be the major causes.

In world 2 and 3 our system had some problems generating appropriate object descriptions. In both worlds there are situations where the target landmark cannot easily be distinguished from all other objects in the player's view because other objects of the same type and color are grouped around the same location. In these cases, our system generated overly complex object descriptions. For example, one instruction that led to some problems in world 2 is *Go past the buttons to the red button straight ahead of you to the right of the blue button.*

Another type of instructions that led to problems several times are instructions that lead the user to a button, but do not instruct him/her to press it. Several users pressed such buttons accidentally.

A portion of the lost games is due to participants who completely ignored the instructions as indicated either by their actions or by the reason they give on the questionnaire for not successfully completing the game. For example, the four participants who lost in world 1 gave the following answers: *too slow*, *had to stop*, *Didn't listen*, *I do not want complete the task correctly.*

Finally, our way of handling errors by immediately replanning sometimes leads to timing problems. Some participants make a mistake, immediately realize it, and correct it, but at that point our system has already started to replan and the correction itself will lead to another error message. This is annoying, may lead to confusing situations, and some more impatient users start to ignore the instructions.

4.3 Discussion

Overall our strategy was relatively successful and robust. Generally, we feel that the landmark based instruction giving strategy was useful – even in this environment where due to its grid based organization a path based strategy relying on step counting can also be realized. Some user feedback also suggests that users prefer these higher level instructions.

The two most problematic areas in our current system are the sometimes overly complex object descriptions and the rather inflexible error handling strategy. The object descriptions can probably be somewhat improved by more carefully designing the wording and by making a wider variety of spatial relations available to the referring expression generation algorithm. However, for a more far reaching solution we need to rethink referring expression generation in situated environments like the GIVE scenario and how it interacts with the decision whether to use landmark or path mode. For example, Stoia et al. (2006) found that instead of producing a complex referring expression, human instruction givers guide users to a position where a simple referring expression can be generated.

Our error handling strategy needs to be more flexible in two ways. It needs a method for detecting corrections by the user so that a new plan is only adopted if necessary. Furthermore, the appropriate user of negative instructions (such as *don't press this red button*) could avoid some fatal error before they happen and would alleviate the prob-

mean	
4.6	overall: What is your overall evaluation of the quality of the direction-giving system? (very bad 1 . . . 7 very good)
4.3	task difficulty: How easy or difficult was the task for you to solve? (very difficult 1 . . . 5 very easy)
3.7	goal clarity: How easy was it to understand what you were supposed to do? (very difficult 1 . . . 5 very easy)
3.6	instruction clarity: How clear were the directions? (totally unclear 1 . . . 5 very clear)
3.7	instruction helpfulness: How effective were the directions at helping you complete the task? (not effective 1 . . . 5 very effective)
3.7	choice of words: How easy to understand was the system's choice of wording in its directions to you? (totally unclear 1 . . . 5 very clear)
3.7	referring expressions: How easy was it to pick out which object in the world the system was referring to? (very hard 1 . . . 5 very easy)
3.7	navigation instructions: How easy was it to navigate to a particular spot, based on the system's directions? (very hard 1 . . . 5 very easy)
3.6	friendliness: How would you rate the friendliness of the system? (very unfriendly 1 . . . 5 very friendly)
56%	informativity: Did you feel the amount of information you were given was: just right / too much or little
62%	timing: Did the directions come ... just at the right time / too early or late

Figure 7: Mean ratings for the subjective measures

lem of users accidentally pressing a button. It may also help some of the participants who choose to not follow the instructions, since they may adhere to a warning, even though they refuse to follow instructions.

5 Conclusion and Future Work

We have described one of the NLG systems that participated in the GIVE-1 challenge. The system uses a landmark based instruction giving strategy and only switches to a path based strategy when it does not find a suitable landmark or when the user has problems following the high level instructions.

Overall the strategy works relatively well in terms of task success and the system performs robustly in different evaluation worlds and with different users.

Currently, the system uses an off-the-shelf module for referring expression generation. To improve this system we will investigate ways to better adapt referring expression generation to the characteristics of direction giving in a situated environment. We will also develop a more flexible error handling strategy.

References

C. Areces, A. Koller, and K. Striegnitz. 2008. Referring expressions as formulas of description logic. In *Proceedings of the 5th International Natural Language Generation Conference*, Salt Fork, OH.

D. Byron, A. Koller, K. Striegnitz, J. Cassell, R. Dale, J. Moore, and J. Oberlander. 2009. Report on the first nlg challenge on generating instructions in virtual environments (give). In *Proceedings of the 12th European Workshop on Natural Language Generation (ENLG2009)*, Athens, Greece.

M. Denis. 1997. The description of routes: A cognitive approach to the production of spatial discourse. *Current Psychology of Cognition*, 16:409–458.

C.-W. Hsu, B.W. Wah, R. Huang, and Y. Chen. 2006. New features in sgplan for handling soft constraints and goal preferences in pddl3.0. In *Proceedings of the Fifth International Planning Competition, International Conference on Automated Planning and Scheduling*. http://manip.crhc.uiuc.edu/programs/SGPlan/index.html.

A. Koller, D. Byron, J. Cassell, R. Dale, J. Moore, J. Oberlander, and K. Striegnitz. 2009. The software architecture for the first challenge on generating instructions in virtual environments. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2009)*, Athens, Greece.

P.-E. Michon and M. Denis. 2001. When and why are visual landmarks used in giving directions? In D. R. Montello, editor, *Spatial information theory: Foundations of geographic information science*, pages 292–305. Springer.

L. Stoia, D. Byron, D. Shockley, and E. Fosler-Lussier. 2006. Sentence planning for realtime navigational instruction. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, NY, NY.

A. Tom and M. Denis. 2003. Referring to landmark or street information in route directions: What difference does it make? In W. Kuhn, M. F. Worboys, and S. Timpf, editors, *Spatial information theory: Foundations of geographic information science*, pages 383–397. Springer.