The Dublin-Bremen System For the GIVE-2 Challenge

Niels Schütte
Dublin Institute of Technology
Dublin, Ireland
niels.schutte@student.dit.ie

Nina Dethlefs
University of Bremen
Germany
dethlefs@uni-bremen.de

Abstract

This paper describes the Dublin-Bremen GIVE-2 generation system. Our main approach focused on abstracting over the low-level behaviour of the baseline agent and guide the user by more high-level navigation information. For this purpose, we provided the user with (a) high-level action commands, (b) lookahead information, and (c) a "patience" period after they left the intended path to allow exploration. We describe a number of problems that our system encountered during the evaluation due to some of our initial assumptions not holding, and address several means by which we could achieve better performance in the future.

1 Introduction

Our initial considerations started out from the example agent, which we treated as a baseline. This agent gave essentially step by step instructions. If a landmark was visible for the next step, the agent would use the landmark as reference. If no landmark was available, the agent would give direction-based instructions.

We identified three key issues for improvement:

1. The baseline agent gave exclusively step by step instructions. While this is appropriate for new paths, users can be more efficiently guided to previously visited places by *high-level instructions*.

2. the baseline agent displayed an overall lack of motivation for the actions. The user was given instructions without context or goal, which seemed frustrating or patronising to us. We therefore decided to provide users with *lookahead information* to motivate their intended actions.

3. The baseline agent handled missteps, i.e. straying from the planned path, by immediately replanning and starting to give directions from the new plan. This approach could be perceived as disruptive, since it discourages the player from exploring. We handled this by including a *"patience period"* before starting to replan.

To address the first issue we reformulated instructions so that they would be enriched with information about higher level goals and the purpose of actions. To address the second issue, we introduced the option of a high-level instruction strategy, where the system could give more abstract instructions, if it was confident that the user was capable of performing the instructed actions. This meant that our system utilized a high level/low level strategy of instruction giving, comparable to the different levels of abstraction used by last year's *GUIDE* system. The third issue was addressed by introducing a "patience" period. The agent would not immediately replan when the user did not follow the plan of instruction, but instead wait for a while to see if the user returned to the plan on their own.

2 Instructions

Our approach to instruction generation can be described as follows: the lowest step of navigation is moving from one region to another (as was the case in the baseline agent). If the target region contains no landmark that could be used for reference, the agent produces a directional move instruction (e.g. “turn around and then go forward.”). If there exists a landmark in the goal region, the player is asked to turn and then move towards it. If the player has to move to an object in the current region that is currently visible, a third instruction type is used that uses a more specific referring expression (Section 2.4). High-level navigation is performed whenever possible (Section 2.1), and instructions are enriched with information on their motivation (Section 2.2).

2.1 High-level navigation

High-level instructions were obtained by conceptually segmenting the plan produced by the original planner into high-level segments. For this purpose, we divided this plan based on manipulate-steps and performed aggregation on all move-steps whose purpose it was to move a player to the position where they could perform the manipulate-action. This method is an equivalent to spatial chunking that has been shown to guide human wayfinding behaviour (Klippel et al., 2009), and has been applied in several route generation systems (Richter and Duckham, 2008; Duckham and Kulik, 2003; Cuayáhuitl et al., 2010). (Dale et al., 2005) use a similar mechanism and call it segmentation.

The main application of high-level instruction was navigation between rooms, where the rooms were taken as identifiable goals. For this purpose, we determined for each room the set of objects contained in it (excluding buttons) and formulated a referring expression for the room. For example, a room containing a chair and nothing else would be referred to as “The room with the chair”. We made the assumption that each room would be uniquely identifiable by enumerating the objects contained in it, and that the number of objects would be limited.

During the game we employed the following strategy. Every time a player entered a room, the system would either introduce the room using a newly tailored referring expression (e.g. “This is the room with the plant”) if the room had not been visited previously, or remind the user that this room had been visited before and mention the objects in the room. The intention was to influence users towards associating rooms with referring expressions so that these rooms could subsequently, at later stages in the game, be referenced more easily.

Since we could not rely on that players would remember the names we assigned to rooms in this way, they were given the option to receive detailed, i.e. low-level step by step instructions, by pushing the “help” button three times. In addition to that, we kept monitoring whether the player was following the plan. If the player diverged from it and did not return within a prespecified amount of time, or if they performed an unintended manipulate-action, the agent had to replan and we switched back to low-level navigation mode.

2.2 Motivation information

Motivation, or lookahead information, was provided to users for every high-level instruction in the game. For example, each segment of move-steps leading to a manipulate-step, we generated a general motivation giving text, that gave the player an outline of what had to be done next and to what purpose.

The procedure for generating motivation information was as follows. First of all, we computed if the next manipulate-action was going to take place in the room the player currently was in. Based on this, the agent would inform the player whether or not they had to go to a different room. On the basis of the effects of the next manipulate-instruction, we calculated what the purpose of the action was, i.e. whether the button had to be pushed to open a door, deactivate an alarm, move a painting, etc. This instruction was then verbalized and passed on to the player. One example information text that could result from this process was “We need to go to a different room now. We need to push a button to open up the safe!”. After the user had performed a manipulate-action, the system summarized the effect of the action (“Alright, the alarm is turned off!”).

2.3 ‘Patience period’

We included a ‘patience period’ of four seconds for users to return to the intended plan on their own once they had left it. The motivation behind was that users may just wish to explore the environment more and immediate replanning may be perceived as disruptive. We would, however, immediately replanning if users performed unintended manipulate-actions, because manipulate-actions could have effects that reversed conditions necessary for the plan to be valid (e.g. pushing a

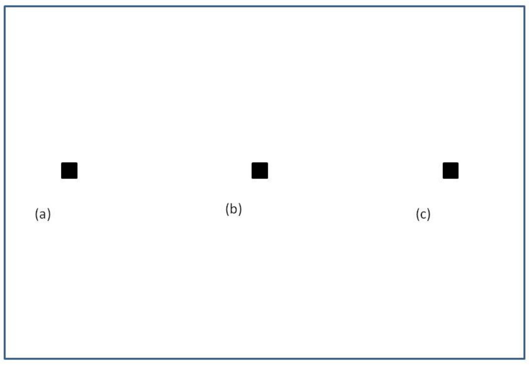

Figure 1: A configuration of three vertical buttons.

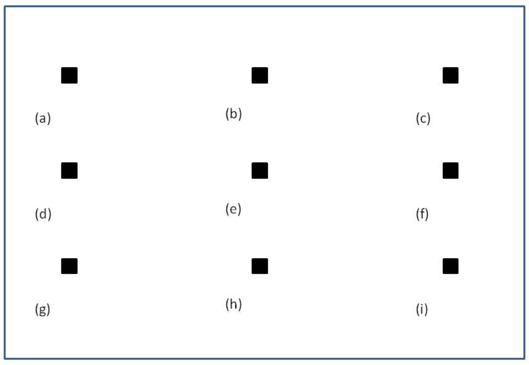

Figure 2: A configuration of nine buttons.

button might close a door the player would have to walk through later on in the plan).

2.4 Referring Expressions

Referring expressions were produced for doors and buttons, and we chose a simple strategy: every time a referring expression was needed, we treated the set of all visible objects as the context set. If the referent was a button, all buttons were selected as input for the referring expression algorithm. If the referent was a door, all doors were selected, excluding doors, that were only visible *through* another door. This is because instructions always referred to the next door the player had to go through.

References to buttons always included the colour of the button, even if it was not necessary for an unambiguous references. We chose this relaxed condition, since recent work in referring expression generation has shown that humans do as well (Viethen and Dale, 2008). If the field of vision contained at least one object that could be confused with the target, we created a referring expression that included information about the relative position of the referent in relation to the other confusable objects. For this purpose we enumerated all confusable objects according to their left-to-right and top-to bottom position on the screen. In Figure 1, button *a* would be referred to as "the 1. button from the left". Button *c* in Figure 2 would be referred to as "the 1. button from the top and the 3. button from the left".

After an instruction including a referring expression had been made, the player had to navigate to the target. If the target disappeared from the field of vision, the player had turned away from the intended target and we obviously needed to switch to a different type of instruction.

3 Problems

The main problems we encountered during the evaluation were due to some of our fundamental assumptions not holding. On the one hand, our strategy of providing users with information motivating their actions led to long and complex instructions that contained many different bits of information. Such instructions were not easily comprehensible for two reasons: (1) displaying long instructions was problematic, because of the fixed display time, which would have made shorter utterance preferable, and (2) presenting instructions incrementally was equally problematic, since displaying each part of the message in turn took a long time during which users could easily grow impatient. This defeated the purpose of providing motivation information, namely to make the task easier for users.

On the other hand, our strategy for generating referring expressions for rooms failed because rooms in the evaluation worlds could contain a multitude of objects. Thereby listing them all led to long and unwieldy referring expressions. Since our high-level instructions were based on these referring expressions, they did not fulfill their purpose of making navigation easier, either. In addition, it appeared that the maps did not require revisiting rooms as much as we had anticipated.

4 Conclusions and future work

We presented our GIVE-2 system whose main contributions were focused on providing the user with high-level navigation information, motivations for their actions and allowing them to explore the environment to a limited extent. Several problems occurred with our implementation of these ideas, namely that our approach to referring expression generation for rooms failed (which high-level navigation relied on), and that our instructions were long and difficult to process. We see the following avenues for future work, which may improve the performance of our system in coming GIVE challenges. Our approach to referring expression generation need to make a selection of the best landmark (or set of best landmarks) to mention, rather than giving a list of all possible candidates. This can be achieved by ranking landmarks based on their salience (Raubal and Winter, 2002) and type (Sorrows and Hirtle, 1999). Further, the choice of *what to say* could be made subject to optimization. In this way, motivation information would be provided to the user only if there is nothing more urgent to convey. In this way, system utterances could be made more succinct and to the point. Additional help (as in the form of motivating actions) may be provided, but only if appropriate. We consider applying machine learning techniques, for example reinforcement learning, to the optimization of system behaviour.

References

Heriberto Cuayáhuitl, Nina Dethlefs, Kai-Florian Richter, Thora Tenbrink, and John Bateman. 2010. A dialogue system for indoor wayfinding using text-based natural language. In *Proceedings of the 11th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing)*, Iasi, Romania.

Robert Dale, Sabine Geldof, and Jean-Philippe Prost. 2005. Using natural language generation in automatic route description. *Journal of Research and Practice in Information Technology*, 37(1):89–105.

Matt Duckham and Lars Kulik. 2003. "Simplest" paths: Automated route selection for navigation. In Werner Kuhn, Mike Worboys, and Sabine Timpf, editors, *Spatial Information Theory*, pages 169–185, Berlin. Springer. LNCS 2825.

Alexander Klippel, Stefan Hansen, Kai-Florian Richter, and Stephan Winter. 2009. Urban granularities - a data structure for cognitively ergonomic route directions. *GeoInformatica*, 13(2):223–247.

Martin Raubal and Stephan Winter. 2002. Enriching wayfinding instructions with local landmarks. In Max Egenhofer and David Mark, editors, *Geographic Information Science - Second International Conference GIScience 2002*, pages 243–259, Berlin. Springer. LNCS 2478.

Kai-Florian Richter and Matt Duckham. 2008. Simplest instructions: Finding easy-to-describe routes for navigation. In Thomas J. Cova, Harvey J. Miller, Kate Beard, Andrew U. Frank, and Michael F. Goodchild, editors, *Geographic Information Science - 5th International Conference, GIScience 2008*, LNCS 5266, pages 274–289. Springer; Berlin.

Molly E. Sorrows and Stephen C. Hirtle. 1999. The nature of landmarks for real and electronic spaces. In Christian Freksa and David M. Mark, editors, *Spatial Information Theory*, pages 37–50, Berlin. Springer. LNCS 1661.

Jette Viethen and Robert Dale. 2008. The use of spatial relations in referring expression generation. In *INLG '08: Proceedings of the Fifth International Natural Language Generation Conference*, pages 59–67, Morristown, NJ, USA. Association for Computational Linguistics.