

GUIDE 3: Improving Order Understanding in Humans by Adding Anticipated Information to Messages

Salvador de la Puente, Rodrigo Denis Cepeda, Carlos León, Pablo Gervás

Universidad Complutense de Madrid

Madrid, Spain

{deniscepeda,neo.salvador}@gmail.com,

cleon@fdi.ucm.es,pgervas@sip.ucm.es

Abstract

Natural Language Generation systems oriented towards giving instructions to users for certain tasks must maximize the level of understanding of these instructions in humans. In particular, generation of orders in natural language for guiding users in physical environments (either real or virtual) can maximize user understanding of each single order by anticipating information about what the user is about to perceive or by adapting the order so that the human reaches a further state in a better situation or location. This solution to GIVE 2 proposes an implementation of these two approaches to the improvement of generation of textual orders, and it shows the evaluation results obtained from the application of the model in a virtual environment with human users.

1 Introduction

Human understanding of messages issued by a computer is the main objective of Natural Language Generation. This general definition, however, has been the focus of a complex task whose parts have caused the birth of several sub-disciplines related to Natural Language. Among them, the automatic creation of orders that are to be given to human users has recently been a focus of attention, as the celebration of scientific challenges regarding order creation evidences (Byron et al., 2007a).

Results prove that this task is far from simple, and different approaches unveil different inner characteristics and problems inherently linked to order creation. The overall goal is, in this case, the improvement of user understanding while retaining in this user the feeling of “human” interaction.

This solution to GIVE 2 presents two different improvements on automatic generation of orders in Natural Language based on anticipation of the content that the human user is about to perceive. Empirical evaluation of the quality of the system is proposed and carried out, and the results are shown.

2 Previous Work

2.1 Referring Expressions Generation

Determining one object by describing it is a challenge in which Natural Language Generation has a lot to say. Although generating descriptions has been a highly investigated field, not many of them center their work in distinguishing an object by relating it with its neighbours (Hervás, 2009). In this article we expand the previous investigation realized in this field (Dionne et al., 2009), by adding the anticipation of what the user would see if he or she gets closer to the target object.

2.2 Guiding in Virtual Environments

The issue of guiding an user through an environment is already a reality. Many institutions offer virtual tours across their installations, and there are applications which let you move through virtual representations of the real world. Moreover, since the popularization of GPS and similar devices, it is possible to get enough information about the position of a person, and use it to orientate him or her. Arrived to this point, the quality of given instructions determines the usability of these technologies.

Many studies have been developed around this topic. A good strategy is to divide the full environment in smaller parts (like rooms, or neighbourhoods if the area is bigger), and understand the final target as a concatenation of smaller sub-targets (Darken, 1995). Also, visual references and qualitative information are easy to understand for a human being, and they can be much more useful than exact references based on mathematics (Murray et al., 2000).

Another important improvement is to transmit the user the sensation of being guided by another human, making the guide to act in a reactive manner (Dionne et al., 2009). This way, it is not expected to just provide information, but also able to analyze the world and determine which way is the best one to give the information. We have focused in this aspect, as Natural Language Generation is closely related.

2.3 Evaluation of Human Understanding of Referring Expressions

Recent challenges for Natural Language Generation have shown different techniques in the evaluation of human response against machine-generated language.

GIVE Challenge (Byron et al., 2007b) was proposed to test virtual guides in virtual environments. It provided a basic tile-discretized virtual environment and a planner to compute the tasks to be fulfilled. Each participant has to develop a natural language generation server to issue guidance instructions to the user. GIVE’s virtual environment was quite simple allowing the user turn only in four directions and moving tile by tile.

GIVE-2 is the natural evolution of GIVE, providing a new virtual environment. The main difference between this version and the first one is that the world is now continuous. The user has freedom of movement (in the former version, he or she could only turn 90 degrees in each movement, and walk from one tile to the next one), making the experience more real. Another big change is that the world is composed by different rooms, and these ones are already identified (you can ask the world to which room does a region belong). Also, there has been some improvement in the planification module, which gives more accurate routes.

3 Improved Discourse Planning for Order Generation

This section details the computational model for anticipation of information in automatic generation of message containing orders for humans. Two complementary approaches have been studied: *vector combination navigation* and *anticipated perception information*.

Furthermore, it introduces the reader to the *Region Discretization Process* of a continuous world which is central to understanding the algorithms developed.

3.1 Region Discretization Process

For testing, we use the virtual environment given by GIVE-2. In this one, world is described as a set of rooms with multiple shapes. This human-like distribution of space tries to avoid cartesian grid representation of the world giving a more human sense of the environment. Thus, there is not a (X, Y, Z) position for an entity but a *region where the entity is*, where it can be manipulated. See Figure 1 to check this. The component which is responsible of discretizing the world into regions is called the *discretizer*.

All regions are rectangles whose sides are parallel to the X and Z axis. First of all, the discretizer divides rooms in one or more *initial regions* in such a way that two regions A, B are adjacent

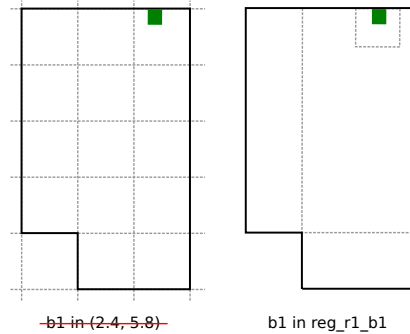


Figure 1: Regions are more understandable for human beings than coordinates.

(i.e. they share one side) only if any point of A is visible from any point of B . In a different order, for each entity there is a region representing the position which is close enough to interact with it. These regions may overlap with the first ones. To avoid this, shared zones are subtracted to the initial regions. That is, if there are two buttons $b1$ and $b2$ in a region $reg1$, there will be regions $reg1.b1$, $reg1.b2$, $reg1.b1.b2$, and $reg1$, representing the positions that are close to $b1$ but not $b2$, close to $b2$ but not $b1$, close to both, and close to neither (Koller, 2010). Thus, the final set of world’s regions is disjoint and discretizer can map now any cartesian position to only one region. Inverse process is coarse approached, we can query for the cartesian coordinates of a region but discretizer just return an approximation of the center of the region given. The example described in this paragraph is represented in Figure 2.

Consider situation depicted in Figure 3. In (a) a bad discretization is shown: line l goes through a wall so regions A and B cannot be adjacent. In (b) we show a correct discretization: A and B are adjacent now but A and C are not.

As we said before, GIVE-2 testing environment includes a planner and the planner generates a route using these regions. So, in order to manipulate an entity we need first to pass through a set of *intermediate regions*.

3.2 Vector Combination Navigation

Vector Combination Navigation (VCN) tries to calculate an always-updated vector with the most suitable orientation the user should face in order to reach some goal position. This position may not be the next region, but the place where next interaction is needed.

Orientation Vector (OV) computation is a simple weighted sum of normalized vectors, see formula 1. It represents the direction the user should take in order to get the goal position walking the minimum path through each intermediate region.

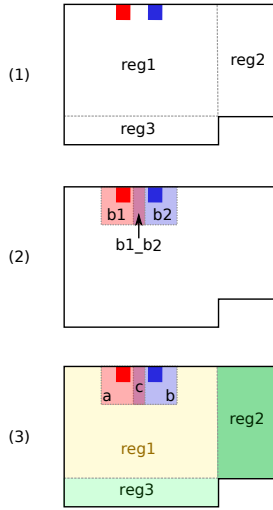


Figure 2: In step (1), the room is divided in regions based on the visibility rule explained in the text, regardless of the contents of the room. Step (2) shows the regions around the objects in the room. In case of a and b , they are so close that a new region appears in their intersection. Finally, in step (3) the regions obtained in the two previous steps are put altogether, avoiding overlaps. Region a is named $reg1_b1$, region b is labeled as $reg1_b2$ and region c is called $reg1_b1_b2$.

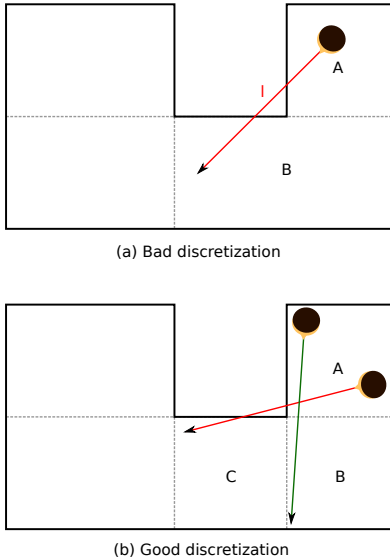


Figure 3: Discretization is a very important step, as it will allow us to know which things are visible from the current region.

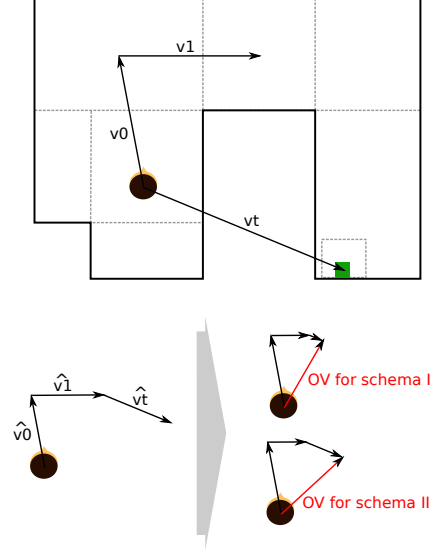


Figure 4: Schemes I & II comparison. Note how Scheme I penalty is greater than Scheme II penalty and more realistic. We expected better results for Scheme I as we will see in Section 3.4.

It can be seen as an approximate anticipation of the goal position.

$$OV = w_0 \cdot \hat{v}_0 + \sum_{i=1}^{n-2} (w_i \cdot \hat{v}_i) + w_t \cdot \hat{v}_t \quad (1)$$

The integer n is the number of regions $\{r_0, r_1, r_2, \dots, r_{n-1}\}$ between the user and the goal. Regions i and $i + 1$ are always adjacent but we cannot prove any other relationships. Vector v_0 points from the user's current position to the center of region 1, and each vector v_i sets its origin in the center of the region i and points to the center of the region $i + 1$. The last vector v_t always points from the user position to the goal's position. Since the only requirement is orientation, normalization is required before weights are applied in order to avoid the scale effect of the vector module.

Furthermore, each weight w_i is a function of the state of the world including both user and goal positions. Weights w_i represent how much relevance does the situation of the region i have to compute OV. In this paper, two schemes of weight are tested. Both use zero weight values for vectors $\{\hat{v}_2, \dots, \hat{v}_{n-2}\}$ and they set $w_0 = 1$ and $w_1 = 0.5$. The main difference is the computation of w_t : *Scheme I* calculates it by considering the inverse of the number of regions between user and goal. *Scheme II* use the inverse of the distance instead. You can see a comparison between both schemes in Figure 4.

At last, in generation stage, OV is compared with user's field of view: if it fits within, then the

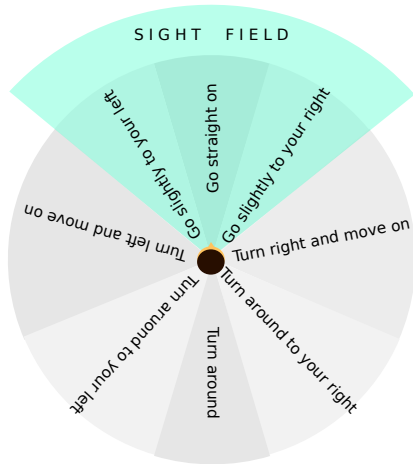


Figure 5: This graphic shows the different messages sent to the user, depending on the obtained OV and the user's orientation.

user is invited to move in some direction, in other words, *to turn while moving*. On the other hand, if vector is outside the user's field of view, then he is forced to correct its orientation by turning first, and then moving. These messages are shown in Figure 5.

3.3 Anticipated Perception Information

Disambiguation is a well known problem. It consists in determining clearly a target item among a contrast set of items with similar properties. Usually, this contrast set is built from a set of contexts that keep, for example, items in the current user's sight field, recently used items, relevant entities, ...

Anticipated Perception Information (API) tries to create *future contexts* simulating the world's state after the user fulfill a navigation order. These new contexts allow the guide to compact the navigation order and the reference expression resulting in a more complex and human-like instruction. Our API implementation determines that the new contrast set contains *what the user will see* after realizing the given movement order.

A complete example is explained in Figure 6. Without Anticipated Perception Information, the situation is the following:

User is in position (1).

GUIDE: Turn right and move on.

User moves to position (2) and can see the target.

GUIDE: Push the red button on your right.

User has enough information to complete the task.

By using Anticipated Perception Information, the given messages should be these ones:

User is in position (1).

GUIDE: Turn right and move on to press the red button on your right.

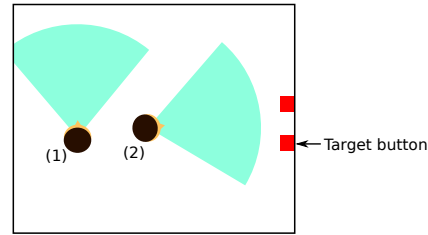


Figure 6: This example shows the improvement in the messages by using anticipated perception information.

User has enough information to complete the task.

Anyway, *present context* is still taken in count, as it may hide the effective impact of API. Let's go back to situation shown in Figure 6. Imagine that, instead of the two red buttons in the right wall there is only the target one, and there is another one in front of the user, in the upper wall. When the user receives the message *turn right and move on to press the red button*, he or she may think that the button in front of him is the target one (even if the order tells to turn right, the user may think that he or she is being directed to the same place, by a longer route). Because of this, it is important to determine in which cases API brings confusion to the user, and disable it. Our guide checks if there is any object similar in type and color inside de user's sight field. If so, the API information is ignored.

3.4 Results

This section shows some comparative results, between using the proposed improvements, and not using them. The data collected is adapted to every modification proposed, and evaluates objective parameters, but also some of them which are dependent on every user's perception (Ruddle and Lessels, 2006).

Results for the VCN and API are analyzed separately, as both ideas are independent. This shows that, together, they can be very powerful, but they also can work on their own.

Vector Combination Navigation

Many people was guided through five different rooms to find a trophy:

Room 1: Small room in which there are a few simple turns.

Room 2: Small squared room in which the trophy is behind the user. Anyway, there are many objects inside the room, making regions a bit irregular.

Room 3: A room with astrange diagonal shape.

Room 4: A very long mix of corridors.

Room 5: A room in which user and trophy are really close in distance, but it is only reachable by rounding through a long corridor.

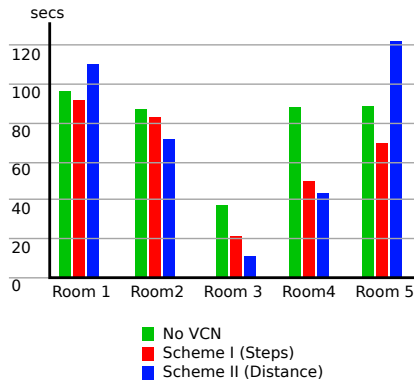


Figure 7: Shows the average time the users needed to solve the task.

Three different algorithms to obtain the Orientation Vector were proven. First one was to take the vector starting in the user's current position, and finishing in the center of the next region (i.e. without using Vector Combination Navigation). The second one is using VCN with scheme I (based on the number of regions between the user and the target), as explained in section 3.2. The third one is using VCN with scheme II (based on the distance from the user to the target). This one is also explained in section 3.2.

Different algorithms are evaluated by three parameters. The first one is the average time used by the users to complete each map, ponderated by their experience in virtual environments. The second one is based on the answers given by every user to a questionnaire after being guided in each map. The questionnaire asks about the clearness and quality of the instructions, pointing from 1 (poor) to 5 (excellent). A weighed average of the answers is calculated. The third one is the average number of times that every user requested help to complete the task.

Results are shown in Figures 7, 8 and 9. Looking at them, we can consider that the three algorithms are very similar in regular rooms. But some shapes, like the Room 3's one, are specially annoying if we do not use VCN. Using scheme I and II, users find their way much more quickly. Anyway, scheme I seems to be difficult to understand for some people, specially older ones (they ask for help frequently). Also, some scenarios are not good for using scheme II, like room 5, as being close to the trophy does not mean that it is reachable.

Anticipated Perception Information

In this case, two different situations have been proposed. The first one is a squared room in which there are two blue buttons on the right of the user, out of his or her sight, similar to the situation seen in Figure 6. This one has been resolved with and without using API. The second situation is inside

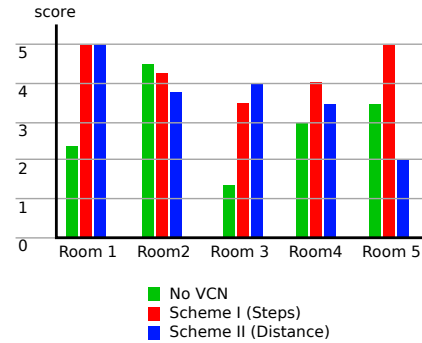


Figure 8: Shows the average score given by the users to the orientation process.

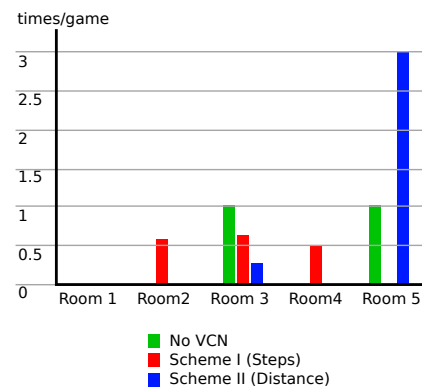


Figure 9: Shows the average number of times the user asked for help.

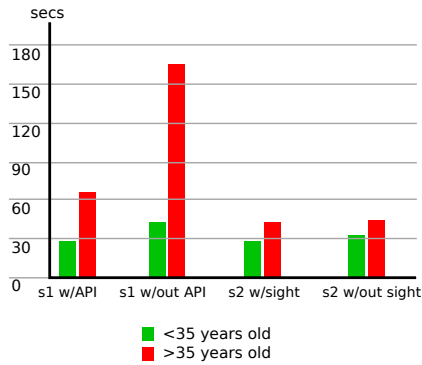


Figure 10: Shows the average time the users took to realize the activity.

an *U* shaped room, in which there is a red button in every side of the *U*. The user is watching one of the buttons, but he has to press the other one. This situation has been resolved using always API, but some times the user sight field has been taken in count to create the instruction, and some others not.

Measures to evaluate the effectivity of these configurations are similar to the ones in the previous section. We have measured the time used by the user to solve the task. Also, a similar questionnaire has been asked to the users. Questions involved how well did they know what they had to do, and the clarity of the instructions. Again, we have counted the average help needing, and the alarms triggered (an alarm is triggered when the user seems to be doing the task wrongly). This time, users have been divided in two groups: the ones younger than 35 years old (and consequently, with more experience in virtual environments), and the older than 35 years old, who are less used to this kind of environment.

Results are shown in Figures 10, 11, 12 and 13. In situation 1, all users seem to agree that disambiguating objects with API is much more useful than waiting to see the object to disambiguate (younger people really like this option!). However, there is division of opinion deciding whether is better to take in count objects in the current sight of view or not. While older people does not appreciate much difference between choosing one option or another, younger people clearly think that instructions that took in count these objects were clearer by far.

Arrived to this point, it looks interesting to have different configuration types depending on the age the user has.

4 Discussion

Analyzing the previous results, it seems than some progress has been done. In common situations



Figure 11: Shows the average score given by the users.

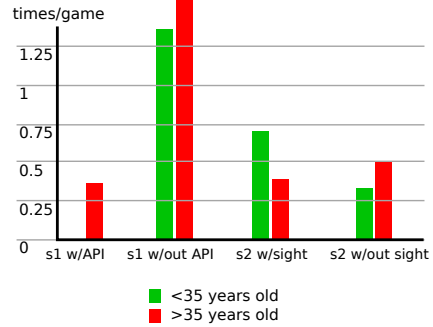


Figure 12: Shows the average number of times that users asked for help.

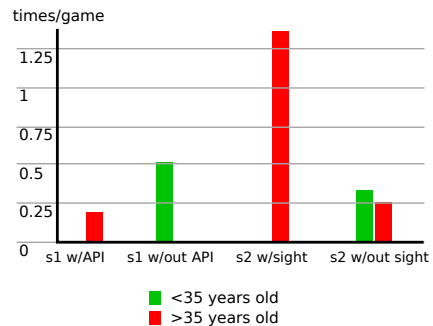


Figure 13: Shows the average number of times that users triggered an alarm.

there is not a big difference in using API and/or VCN or not. But in some cases in which the usual guidance can find problems, these ideas give a better description of the actions to be done. However, there are still some cases in which these improvements are not enough, as different people understand the same world in a different way. Tests realized showed that while most of the people felt that API was a big improvement, there is still people who feel more comfortable with instructions that make them lose more time, but let them advance little by little.

From a different point, we have seen anticipation at guidance level but another kind of anticipation can be found at planning level. GIVE-2 planner is linearized, i.e. to manipulate a button in other room, GUIDE2's original plan guides the user through all intermediate regions. Nevertheless it is possible to rearrange the basic plan in order to *anticipate* some instructions. A discussion of these ideas can be found in (Dionne et al., 2009).

5 Conclusions and Future Work

In this report we have discussed navigation assistance in virtual environments, giving some alternatives to the already existing methods to guide a person.

Vector Combination Navigation provides more accurate directions to orientate the user, by considering not only his or her immediate movement, but also the complete way to solve the given task. Although this way can be interpreted in many ways, in this document we propose two different ones. The first one, which decreases the importance of the target's position as the number of regions between it and the user increases, has shown very good results in the practice. The second one, based on the distance towards the target object, has demonstrated its power in general cases, but in some cases it is worse than not using VCN.

On the other hand, Anticipated Perception Information tries to generate more complex sentences indicating the ubication of the target object by supposing what the user would see when he or she gets close to it. This way, user receives all the information at once, so he or she can solve the task more freely. Anyway, sometimes this seems to be a confusing when what the user is watching now and what he would see are similar (users tend to not trust orientation advices if they already see something similar to their target object). This confusion seems to be deeper in younger people used to move across virtual environments.

Both improvements have demonstrated to work well in most situations, but there is still margin to continue evolving. Also, using different configurations for these ideas makes them more accurate in certain situations. Because of this, it would be in-

teresting to find a way to choose this configuration in an ideal way. We have watched that age and experience can determine this values, but it would also be interesting to let those parameters evolve dinamically, depending of the response of the user.

Also, during the experimentation process, we received really useful feedback from the users guided by the test application. Some of the suggested that information about which direction to take sometimes are already determined by the landscape. If we are in a corridor which turns to the right, our only way to continue our trip is by turning right. In that case, telling the user to turn is obvious, and a simple *straight on* could be a synonym (even though, telling him to just continue walking would be a better choice).

References

- Donna Byron, Alexander Koller, Jon Oberlander, Laura Stoia, and Kristina Striegnitz. 2007a. Generating instructions in virtual environments (GIVE): A challenge and evaluation testbed for NLG. In *Proceedings of the Workshop on Shared Tasks and Comparative Evaluation in Natural Language Generation*, Arlington.
- Donna Byron, Alexander Koller, Jon Oberlander, Laura Stoia, and Kristina Striegnitz. 2007b. Generating instructions in virtual environments (GIVE): A challenge and evaluation testbed for NLG. In *Proceedings of the Workshop on Shared Tasks and Comparative Evaluation in Natural Language Generation*, Arlington.
- Rudolph P. Darken. 1995. Wayfinding in large-scale virtual worlds. In *CHI '95: Conference companion on Human factors in computing systems*, pages 45–46, New York, NY, USA. ACM.
- Daniel Dionne, Salvador de la Puente, Carlos León, Raquel Hervás, and Pablo Gervás. 2009. A model for human readable instruction generation using level-based discourse planning and dynamic inference of attributes disambiguation. In *12th European Workshop on Natural Language Generation*, Athens, Greece, 03/2009.
- Raquel Hervás. 2009. *Referring Expressions and Rhetorical Figures for Entity Distinction and Description in Automatically Generated Discourses (extended summary in English)*. Ph.D. thesis, UCM.
- Alexander Koller. 2010. GIVE-2: Participants' Manual. <http://kenai.com/projects/give-challenge/pages/Manual>.
- Craig D. Murray, John M. Bowers, Adrian J. West, Steve Pettifer, and Simon Gibson. 2000. Navigation, wayfinding, and place experience within a virtual city. *Presence: Teleoper. Virtual Environ.*, 9(5):435–447.

Roy A. Ruddle and Simon Lessels. 2006. Three levels of metric for evaluating wayfinding. *Presence: Teleoper. Virtual Environ.*, 15(6):637–654.