Constructing World Abstractions for Natural Language in Virtual 3D Environments

Carlos León[1], Salvador de la Puente[2], Daniel Dionne[2], Raquel Hervás[1] and Pablo Gervás[3]

Abstract Human-Computer interaction can be clearly leveraged by using Natural Language systems, specially in interactive environments. In this paper a system which interprets a 3D virtual world in terms of human abstractions like "corridor" or "crossroad" and differences between world items basing on the properties the user refers to, is presented. This interpretation is oriented to human orders understanding, making it possible for the system to be interactively guided by the user. A partial implementation of the algorithm and its results are shown, and discussion about how the system could be empowered for better Natural Language Processing is detailed.

1 Introduction

Human-Computer interaction is known to be a challenging field in Computer Science, and there exist several approaches for leveraging the interactive experience when a human user communicates with a machine. Among them, Natural Language Processing is perhaps one of the most interesting approaches because most humans use natural language for ordinary communication.

Usually, interactive systems which implement Natural Language Understanding receive user input which is not explicitly represented in the computer database. For instance, the computer could be showing the user a 3D world internally represented with *tiles*, and the user could ask for information about a *corner* he sees. In this situation, even if the corner really exists from the user's *point of view*, the computer would not understand the order.

Departamento de Inteligencia Artificial e Ingeniería del Software, Universidad Complutense de Madrid, Spain e-mail: [1]{cleon, raquelhb}@fdi.ucm.es, [2]{neo.salvador, dionnegonzalez}@gmail.com, [3]pgervas@sip.ucm.es

Alternatively, if the system has to describe a certain location to the user, or to give him a particular instruction, it may be restricted to using the conceptual vocabulary that is explicitly represented within the reference 3D world. For instance, the system may describe the location or express the instruction in terms of the tiles it covers or the sequence of tiles that has to be traversed. However, for the user, it would be much more intuitive to talk about abstract concepts such as corridors, doors, corners, or rooms.

It is possible to encode a hard-wired world representation by which all possible definitions of every part of the world are already included in the representation. Doing this by hand is very effort consuming and it may not be valid in worlds that change dynamically over time. For these environments, a dynamic, computerized system can help the machine to understand a larger set of commands or dialog parts, or to be able to formulate instructions or descriptions in human abstractions, and either way create a better user experience.

For the communication to be more human-like, thus improving the quality of the system, it is necessary to recognize input messages expressed in abstract terms, or to formulate output messages in correspondingly abstract terms. In this paper a system for Natural Language abstractions processing in virtual 3D environments is shown. The user has visual 3D information, and the system has explicit knowledge about the world in terms of low level graphical representation (tiles, objects, valid positions). The system is designed to follow a set of commands like moving the camera representing the user or focusing on something interesting, and/or to guide the user through the 3D environment by providing instructions. Orders from the user and descriptions from the system could possibly include high level abstractions (from the low level graphical information point of view) like *door*, *corridor* or *corner* as the user sees them. Correctly understanding these orders involves an abstraction process in which the 3D graphical world representation is *interpreted* in a human-like manner. The system should also be capable of *formulating* its instructions in abstract terms along the same lines.

Figure 1 shows a comparison between an architecture with direct Natural Language Recognition (top) and an architecture with the presented system in the design. The main benefit of including this system for order recognition consists on the fact that some properties which are expressed by a human user are not explicitly present on the virtual world, and thus they can not be identified by the simpler version. Basically, the main addition to the system is the *extension* module, which receives a world representation and expands it to automatically contain more information, following algorithms like that shown in Figure 1.

2 Previous Work

Natural Language Understanding has been a research topic for many years, starting with the very beginnings of AI. Since then, it has gone to several stages, full of high hopes at the start, followed by disillusions when problems proved much more

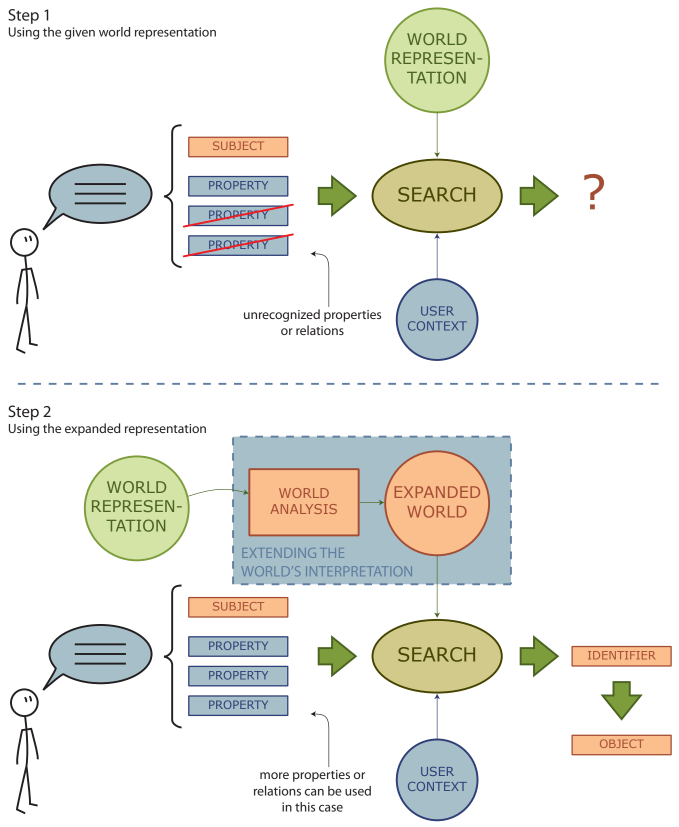

Fig. 1 Architecture comparison between a simple order recognizer (top) and the Natural Language Understanding in Virtual Environments System (bottom).

difficult than expected. The field has reappeared as a dignified research topic in recent times, partly as a result of the improved results in parsing achieved by statistical parsers, and the surge of new solutions arising from the application of machine learning techniques to the area. In general terms, this transition in practice pushed the state of the art past the original obstacles related to parsing, to be stopped again by problems related to semantics.

In this context, natural language interaction in virtual environments provides an interesting case study because the 3D representation of the world that is being used to graphically render the environment can double up as an explicit representation of the semantics of the world.

Solving Natural Language Understanding in Virtual Environments problems has been tackled from several different points of view. In [8], how people refer to similar objects using disambiguating characteristics, and how this can be applied to a recognizing system is studied. Other Natural Language systems are applied robotics, [12], where spatial utterances recognition are solved by using state information gathered by the robot's sensors.

For referencing elements in Natural Language in a 3D environment (whether it is real or virtual) several projects try to add logic information about certain relations between items in the space. Creating a logic description for *near* or *by* usually helps systems to better understand human utterances about 3D spaces [11]. Referring expression generation studies not directly related with virtual environments also apply in this field, as many Natural Language Generation issues are general enough for including this kind of systems [7, 10].

Ontologies play a significant role in this field. Formally describing what is present in a virtual world is usually needed, and related work can be found in the field of spatial ontologies [9, 4, 3, 5]. Most of them try to create a reasoning system based on ontologies which are not necessarily constrained by three dimensions.

Logical spatial reasoning shown in [6] presents a logical framework for identifying spatial properties of objects (or between objects). [2] presents basic spatial relations between object which set the base for spatial logical reasoning.

All this systems are related to Natural Language and interactive environments in some sense. However, they assume a fixed, explicit or implicit representation of objects and a "labeling" for them. Next sections explain an approach for identifying these labels for a subset of 3D constructions which could help these systems to understand new different parts of a 3D space.

3 Constructing World Abstractions from Graphical 3D Information

This section shows a system which exchanges instructions in Natural Language with a user, and either identifies correctly what the user wants to say when he is referring to complex terms like *corner* or *wide corridor*, or is capable of using such terms when giving instructions to the user. The system relies on ground graphical 3D information from the world and the current user state to correctly compute what parts of the world the user wants to interact with, or how to refer to the parts of the world that the system has to mention.

3.1 Internal World Representation

The system is designed to handle a very low level world representation based on a 2D tile grid, which can be considered to be unbound (although real computational requirements will demand the opposite). This space discretizes space into four directions (North, South, East and West) that the user can face (Figure 2).

On this 2D grid, there are defined a set of walls whose layout can be considered to form rooms and corridors (and thus corners, doors, and so on). Discrete items (like lamps or chair, for instance), are also laid out all along the virtual world. These items have properties which difference them from other items of the same class in the world. The representation also includes a set of rules which precise how the world dynamically changes when the user performs some actions: *the user pushes a button → a wall moves, uncovering a new corridor*. Figure 2 shows a schematic diagram representing the virtual world and its elements.

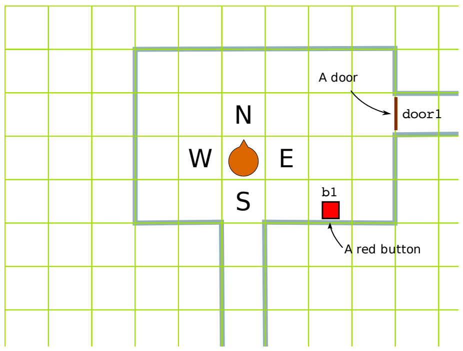

Fig. 2 Walls, items like the door and the button, and properties like the red colour of the button.

Two basic actions for this world are defined: *movement* and *interaction*. The user can **move tile by tile** forward, backward, to the left and to the right. He can **turn left or right** too, facing each of the four discretized orientations explained before. The user can **focus on an world element** and interact with it. Interactions include pushing buttons and taking objects.

Note the world does not include any information related to different rooms, corridors, corner and other human space concepts like these. All this information must be inferred from the basic world representation.

3.2 Abstracting 3D Information in Understanding

When the system receives an order in Natural Language, and after a simple preprocessing stage where the order and the parameters are identified, the recognition stage starts.

The goal is to match human-like orders including basic actions, which will be interpreted using the world information. The system pre-processes complex orders by simply splitting them into simple ones, which will be issued to the recognizing algorithm one by one. The first stage, **instruction matching**, applies simple first-order logic like templates to the received order and fills some slots for it to be useful. For instance, an instanced template could have the following form: $go(corner)$ which will be interpreted by the system as the desire of the user of going to some corner.

As said before, there are world elements which are not explicitly declared as world structures. However, the user could refer to them because a human can recognize these elements if they are inside his visual field. The system uses some predefined **keywords** from the instanced action template to trigger certain types of **search agents**. This agents are just simple threads which look for items (lamps, for instance) or world parts (corners) which match some properties present in the user's order.

Different types of search agents return different sets of matching items. These agents could find items (buttons, red things, lamps...) which are simple, referenced things in the world whose properties are used to disambiguate between them, and world parts like corridors, corners, and so on, which are physical abstractions which have to be inferred from the world layout.

For instance, consider the next sentence: "*Go to the left corner*". The word *"corner"* will trigger a spatial search agent to identify possible referred corners and the word "*left*" will trigger another spatial search agent discerning between elements at left of right from the user's position and orientation.

Consider another instruction: "*Push the red button*". The word "*red*" will trigger a property-based search agent to discover red elements. The "*button*" word triggers a button-oriented search agent which will find all buttons inside the user visual field.

Several geometrical search agents for rooms, corridors and corners identifying have been developed. Algorithm 1 shows as an example the basic procedure for rooms. When the user refers to a "room" and a search agent finds a room in his visual field, for instance, the algorithm maps that reference ("room") with the set of tiles, and then this information can be used for the order execution: once the algorithm knows that the user is referring to the subset of tiles which are enclosed inside a square bounded by the coordinates (x_i, y_i), (x_j, y_j), the system can move the camera to the center tile inside that room.

A small trace in a square room with a small corridor could be like this: first, the cursor is set in any valid location and orientation: $(2,2)$, facing positive y. Then, it advances until $(2,4)$. A new room is added, and the recognition starts: the cursor turns right to $(4,4)$. The to $(4,0)$, then to $(0,0)$, and then to $(0,4)$. Now, the cursor finds a gap in $(1,4)$ which end in $(2,4)$, the starting point. With this process, a new room has been identified.

Once every search agent is done with its search, they all have a set of matching elements: a "button" agent has a set of buttons, a "red" agent has a set of red items. The intersection of these set lets the system obtain a new set which will contain zero, one or more references to items or parts of the world. In case this intersection contains only one element, the matching process has succeeded and there is only one possible item the user is referencing to. If the result are zero or more than one items, the system will have to ask the user for a more detailed order.

Search agents can be applied following no particular order. A successful set of agents always come up with just one element but the order affects performance. If the set of agents does not find one element only there is an **ambiguity** and the system has to give feedback to the user for him to specify what he wants in more detail.

Algorithm 1 Looking for rooms.

```
1: cursor ← user position
2: orientation ← user orientation
3: advance cursor facing orientation until it finds a wall
4: start location ← wall location
5: turn right
6: add new unexplored room to the room list
7: while there are unexplored rooms in the room list do
8:     while cursor is not at start location do
9:         advance cursor facing orientation until it finds a corner
10:        if cursor comes up with a concave corner then
11:            turn right
12:        end if
13:        if cursor comes up with a convex corner then
14:            if there is a door at cursor location then
15:                add a door at cursor location
16:                add new unexplored room to the room list
17:            else
18:                turn left
19:            end if
20:        end if
21:    end while
22:    add discovered border to the room
23: end while
```

Let us say that a Natural Language processor receives an order: “go to the corner near the window”. After processing, it outputs the recognized structure, in the form: $go(corner), isnear(corner, window)$. Several search agents are created. One of them, following a process analogous to the one shown in Algorithm 1, tries to find a corner inside the user’s visual field. Another one looks for a window. If the search agent finds two corners, following the expression, the system will identify the corner which is nearer the window as the referred corner by intersecting the two returned sets from the search agents.

The system stores a set of rules for interpreting the valid set of verbs and references. Meanings for predicates like *go* and *isnear* are hard-coded in the interpretation system, so no previous processing is needed. Although this is far from the best possible alternative, it has been proved valid for this prototype. Improving it is a main part of the future work.

Applying the algorithm in a real system could produce, for instance, if the camera represented the user position in the virtual environment (imagine a first person system), camera movement towards the corner. A planning system, in this case, would also be needed, but this functionality is beyond the scope of this paper.

3.3 Abstracting 3D Information in Generation

These algorithms for world parts and item identification can also be applied to Interactive Systems which need Natural Language Generation where, for instance, a planning algorithm must show its solution in terms of high level Natural Language.

For this to be possible, the orientation of the system must be inverted. Instead of having an order to be executed, the system can have a plan to be explained to some user. This plan could be represented as a list of steps the user must follow in order to reach some point (for instance, when guiding a person the system could know where and how to go, and it could have to explain it to that person in Natural Language). Each step should be realized without ambiguities. This option is carried out by substituting search agents with referring agents, which receive a low level order (like *push button number 5*) and has to perform a search in the user visual field and consider all possible objects to correctly identify the desired button. If the system detects three buttons in the user's visual field and they all have different colours, the system could choose to say "push the green button". The algorithm is complementary to the previous one: each referring agent looks for a property or item ("button" or "red"). Then, from these sets, the system chooses the definition that uniquely defines the desired item.

This process, however, is not necessarily an inverse mapping from the understanding process. Several other issues must be taken into account. For instance, the plan or the task the user is to do, should be clearly, briefly explained in the beggining for the user to have a general idea of the task. Also, a guiding system like this one must consider the possiblity of the user being lost, and it has to create a solution which has to be explained to the user in terms of the abstracted world.

For demonstration of this being possible, the abstraction module has been implemented and tested in the GIVE Challenge (*Giving Instructions in Virtual Environments*) [1]. In this challenge, the main objective is to guide the user in a virtual interactive changing 3D environment. In this virtual world, the system only receives low level graphical information, and it is necessary to feed the user with high level orders, pretty much the inverse to understanding high level user orders. This challenge has clearly demonstrated that correctly identifying graphical information and creating abstractions for communicating with human users leverages the interactive experience.

4 Discussion

The presented system tries to fill a small but important gap in the field of interactive systems which communicate with the user using Natural Language. The presented algorithm has been created to add new functionality to Natural Language systems, specially those related with dialogs and orders in dynamic, interactive environments.

Abstracting information in real-time is a must for dynamic virtual environments. Offline world reconstruction for finding abstractions is only useful in static worlds, where everything remains the same independently of the user action. For a Natural Language recognition system to be fully deployable in an interactive environment, online capabilities are a hard requirement. However, offline addition of Natural Language information can still be useful in static systems (or even partially useful in dynamic systems). However, creating this virtual information by hand is tedious and time consuming. Using the algorithm for this purpose automates the process and can be applied in many different domains: equivalent low level representation is present in many different interactive systems, and Natural Language Understanding can be used for many tasks.

The whole paper references *graphical information* as the internal representation of the 3D virtual world. As show in Section 3.1, this information is not purely graphical. The system is handling *tiles* and not just plain points in a 3D space. Although many virtual environments can be accessed assuming this level of information can be accessed, for a system to fully understand Natural Language sentences in a wide set of environments, this assumption can not be always made. Further research about how connecting low level graphical algorithms with this recognition systems can be very interesting.

In case of a uninterpretable order such "do something with that thing", the system discards it considering it has found an **instruction ambiguity**. The system, then, must give feedback to the user, asking him for a more detailed explanation about the action he is really referring. When the action is identified we can move to the next stage. It is however possible that such a sentence is not lacking meaning if the user just used some item (which could be mapped to *that thing*) and that this item only permits one operation (thus *something* could be mapped to that operation). This level of Natural Language Understanding is planned as future work.

The agent oriented design that has been followed during the development of the system tries to create a framework that scales well. Agents must implement a simplistic interface, which is basically composed by functions receiving the world state and return new high level elements. Thus, the scalability complexity is dependent on the high level concept (it might be more difficult to identify a round door than a square window) and not on the architecture itself. However, this has not been exhaustively tested and it is planned as further study.

It is important to note that the current implementation of the algorithm can recognize many typical 3D patterns, but not every possible layout which could be referenced by humans. This first approach has been proved to be useful in certain environments which are not extremely complex. For instance, the algorithm would not recognize sinuosity in a corridor. Studying different environment constructions is a fundamental part of the future work.

5 Conclusions and Future Work

A system which generates world abstractions (in the form of *corridor* or *corner*) from basic 3D information (like *there is a wall in (x, y)* has been detailed. The system is conceived as a part for Natural Language Interaction systems which receive orders in virtual interactive 3D environments.

Several future research lines are being developed to empower the system. Linking the system output with a spatial ontology for better understanding of concepts expressed by the user could leverage the success rate, thus creating a more useful system. Also, a better Natural Language Processing system can be plugged into the system in order to obtaining a more complex, richer interactive system. Identifying 3D constructions which are referenced by human users is fundamental to build a reliable and power system. Current ongoing research on this is on process. It would be very interesting to carry out a deep study about what specific parts of the world humans refer to when interacting in a virtual 3D environment.

References

1. Generating Instructions in Virtual Environments (GIVE) (2009). URL http://homepages.inf.ed.ac.uk/v1akolle/proj/give/
2. Clarke, B.L.: A calculus of individuals based on connection. Notre Dame Journal of Formal Logic (1981)
3. Coenen, F., Beattie, B., Bench-capon, T.J.M., Shave, M.J.R.: An ontology for linear spatial reasoning. In: In, pp. 718–727. Springer Verlag (1996)
4. Coenen, F., Visser, P.: A core ontology for spatial reasoning
5. Coenen, F., Visser, P.: A generic ontology for spatial reasoning. In: In Research and Development in Expert Systems XV, proc. of ES98, pp. 44–57. Springer Verlag (1998)
6. Cohn, A.G., Bennett, B., Gooday, J., Gotts, N.M.: Representing and reasoning with qualitative spatial relations about regions
7. Dale, R., Reiter, E.: Computational interpretations of the gricean maxims in the generation of referring expressions. Cognitive Science: A Multidisciplinary Journal **19**(2), 233–263 (1995). DOI 10.1207/s15516709cog1902_3
8. Gorniak, P.: A visually grounded natural language interface for reference to spatial scenes. In: In Proceedings of the International Conference for Multimodal Interfaces, pp. 219–226. ACM Press (2003)
9. Grütter, R., Bauer-Messmer, B., Hägeli, M.: Extending an ontology-based search with a formalism for spatial reasoning. In: SAC '08: Proceedings of the 2008 ACM symposium on Applied computing, pp. 2266–2270. ACM, New York, NY, USA (2008). DOI http://doi.acm.org/10.1145/1363686.1364227
10. Piwek, P.: Meaning and Dialogue Coherence: A Proof-theoretic Investigation. J. of Logic, Lang. and Inf. **16**(4), 403–421 (2007). DOI http://dx.doi.org/10.1007/s10849-007-9046-1
11. Reddy, V., Neumeier, K., McFarlane, J., Cothren, J., Thompson, C.W.: Extending a natural language interface with geospatial queries. IEEE Internet Computing **11**(6), 82–85 (2007). DOI http://dx.doi.org/10.1109/MIC.2007.124
12. Stopp, E., Gapp, K.P., Herzog, G., Laengle, T., Lueth, T.C.: Utilizing spatial relations for natural language access to an autonomous mobile robot. In: KI–94, Advances in Artificial Intelligence, 18th German Annual Conference on Artificial Intelligence, pp. 39–50 (1994)