Natural Language Communication With Virtual Actor

MUSLIM Aries, KARYATI Cut Maisyarah (1)
Gunadarma University
Jakarta-Indonesia amuslim@staff.gunadarma.ac.id,
csyarah@staff.gunadarma.ac.id

KURNIAWAN Robby (2)
Gunadarma University
Jakarta-Indonesia
Robby_kurniawan@student.gunadarma.ac.id

Abstract

The development of realistic virtual actors in many applications, from user interface to computer entertainment, creates expectations on the intelligence of these actors including their ability to understand natural language. Based on our research in that area over the past years, we highlight specific technical aspects in the development of language-enabled actors. The embodied nature of virtual agents lead to specific syntactic constructs that are not unlike sublanguages: these can be used to specify the parsing component of a natural language interface. However, the most specific aspects of interacting with virtual actors consist in mapping the semantic content of users' input to the mechanisms that support agents' behaviours. We suggest that a generalisation of speech acts can provide principles for this integration. Both aspects are illustrated by results obtained during the development of research prototypes..

1. INTRODUCTION

The increased visual realism of virtual agents naturally creates expectations on their intelligence and, as many of these are either interface agents or virtual actors, their ability to understand human language. In this paper, we focus on some key technical problems aspects in the design of language-enabled virtual agents.

Virtual agents are embodied in a physical (although virtual) environment: apart from the properties of any specific task they have to carry, this embodiment is at the heart of understanding the requirements for NLP. The embodiment of virtual agents requires that their understanding of language is entirely translated into actions in their environment. Although this problem has been described as early as 1970s in the SHRDLU system, no systematic account has been attempted until the mid-90s.

The most generic representation of an agent behaviors is a *plan*. This is why the semantics of actions can be described as relating the utterance content to plans to be executed by the agent. Previous work from Webber et al. has classified various forms of language statements in terms of the complexity of actions that should result from them. This classification distinguishes, among others, doctrine statements, purpose clauses and procedural instructions. *Doctrine statements* express "general policy Natural Language Communication with Virtual Actors 149 regarding behavior in some range of situations", such as avoid confrontation as much as possible. These very high-level statements can only be understood by an agent possessing sophisticated reasoning mechanisms.

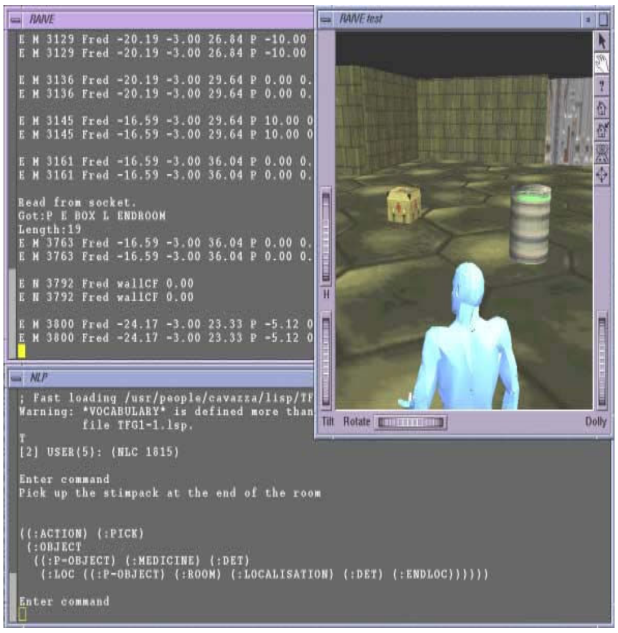
Purpose clauses are instructions that convey the goal of an action. One example in a computer games corpus is shoot a barrel to get rid of most of the pink demons. It is not so much the explanatory nature of this statement that matters as the implicit instructions that it carries. In other terms, it means that the character should wait for the pink demons to come in close proximity to the barrels before opening fire. Both doctrine statements and purpose clauses require complex inference mechanisms that can only be implemented within autonomous agents with intentions.

Procedures correspond to actions to be taken immediately or in the near future, subject to specific pre-conditions being met. These can however relate to complex action sequences, including some variability due to specific configurations or changes in the virtual world. In this paper, we investigate two main aspects of interacting in natural language with embodied virtual actors. We do so through different research experiments we have been conducting over the past few years, whose evolution reflects the progress in the integration between natural language processing and the agents' behavioral mechanisms. The first one deals with the basic requirements of linguistic processing and explores how traditional parsing problems should be approached in this context. The latter attempts to relate the semantic content of natural language input to the mechanisms that support agent behaviors.

2. THE THEORY

There are still few real-world applications in which a user would interact with a virtual actor. In order to study the corresponding technical requirements in a realistic environment,we explored the possibility for a human player to control the characters in a computer game using natural language instructions. Computer games provide large scale environments and limited but well-defined tasks; we selected a classical game at the time of our experiments, DOOM™, for which many on-line resources were available, and designed a natural language interface for part of the game. The first step was logically to carry a corpus study in order to establish a list of the most relevant linguistic phenomena. The DOOM™ "spoiler" corpus we used was an on-line corpus available from http://www.gamers.org. It described in natural language the traversal of DOOM™ levels. Typical spoilers alternate the description of landmarks, item locations, and describe sequences of actions to be taken by the player. Here is a typical excerpt from a DOOM™ spoiler:

Enter the door with the skull on it and push the switch. Walk out of the room and turn
right. There are now stairs going into the wall, which is fake. Enter the teleporter,
you're now in a circular room; find the secret door (the wall with the face on it) to go to the next circular room and enter the teleporter.



```
E M 3129 Fred -20.19 -3.00 26.84 P -10.00
E M 3129 Fred -20.19 -3.00 26.84 P -10.00

E M 3136 Fred -20.19 -3.00 29.64 P 0.00 0.
E M 3136 Fred -20.19 -3.00 29.64 P 0.00 0.

E M 3145 Fred -16.59 -3.00 29.64 P 10.00 0
E M 3145 Fred -16.59 -3.00 29.64 P 10.00 0

E M 3161 Fred -16.59 -3.00 36.04 P 0.00 0.
E M 3161 Fred -16.59 -3.00 36.04 P 0.00 0.

Read from socket.
Got:P E BOX L ENDROOM
Length:19
E M 3763 Fred -16.59 -3.00 36.04 P 0.00 0.
E M 3763 Fred -16.59 -3.00 36.04 P 0.00 0.

E N 3792 Fred wallCF 0.00
E N 3792 Fred wallCF 0.00

E M 3800 Fred -24.17 -3.00 23.33 P -5.12 0
E M 3800 Fred -24.17 -3.00 23.33 P -5.12 0
```

```
; Fast loading /usr/people/cavazza/lisp/TF
Warning: *VOCABULARY* is defined more than
         file TFG1-1.lsp.
T
[2] USER(5): (NLC 1815)

Enter command
Pick up the stimpack at the end of the room


((:ACTION) (:PICK)
 (:OBJECT
  ((:P-OBJECT) (:MEDICINE) (:DET)
   (:LOC ((:P-OBJECT) (:ROOM) (:LOCALISATION) (:DET) (:ENDLOC))))))

Enter command
```

Fig. 1. Natural Language Instructions to a DOOM™ game Emulator

Most importantly, they correspond to some kind of briefing that would be given to a player before his gaming session. Such a briefing incorporates advice along a description of a temporal sequence of actions to be taken, including the consequences of previous actions (e.g. "Enter the teleporter, you're now in a circular room"). These actions are in limited number and essentially include various displacements, collecting objects as well as combat moves. Yet, there is a great deal of variability in issuing instructions to carry out these elementary actions, which justifies the use of linguistic processing.

This corpus shows many regularities suggesting sociolectal aspects, which could be characterized as a sublanguage [4]. This would bear significant implications in terms of natural language processing. On the other hand, a common method to design natural language interaction is by means of habitable languages [5]. These are formally defined controlled languages, which are designed to facilitate language processing in a given application by making parsing tractable. They approach natural communication by defining a sufficient number of formally specified variants of standard expressions that can be encountered in the task domain. In habitable languages, the practical approach consists in identifying the system actions targeted, investigating the most frequent surface variants for the associated commands, and generating a set of variation rules.

Communication with virtual actors finds itself in-between these two paradigms: on one hand, depending on the nature of the application (e.g. computer games), it is possible to recognise the emergence of actual sublanguages. On the other hand, limitations in speech recognition and parsing might make recourse to habitable language a necessity.

3. RESEARCH METHOD

We have based our parser on a simplified variant of Tree-Adjoining Grammar (TAG) [9], Tree-Furcating Grammar (TFG) [10]. TFG have been shown to be less powerful than TAG, as some constructs cannot be represented in TFG [11]. This, however, does not affect our parser whose coverage is meant to be limited to the habitable language we have defined.

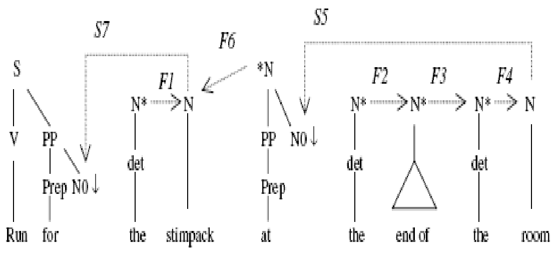


Fig. 2. Parsing in the TFG Formalism

3. 1 The Parsing Algorithm

Parsing consists in combining all trees in a forest until a single tree of root S can be produced or no further operations are possible. In the TFG formalism, trees are combined through two elementary operations: substitution and furcation. Substitution replaces a pre-defined substituable node, acting as a placeholder (e.g. the N0 node in Figure 2.) with a compatible tree of similar category (e.g. a N tree). From a semantic perspective, the substituable nodes are often placeholders for action parameters. For instance in the tree Run-for-N0, N0 stands for the object to be collected.

As an example of tree fusion operations, in Figure 2, the nominal phrase the stimpack, of type N, will be substituted in the initial tree at leaf N0. Furcation adjoins an auxiliary tree to its target tree, thus adding an extra branch to it. It is a simplified variant of the adjunction operation that was initially described by De Smedt and Kempen [12]. While substitution can only take place at determinate node, nodes for furcation are determined dynamically.

For instance, furcation of an auxiliary tree of type N takes place on the rightmost N leave of the target tree [10]. One of the advantages of furcation is that it results in trees of moderate depth, which speeds up tree traversal at further stages of parsing for successive furcations. As we have seen, furcation generally involves modifiers such as adjectives (of N* root), which can add their semantic information to the tree they modify during the furcation process. The "flatter" trees obtained with furcation evidence the prevalence of dependency over constituency structures.

Adjacent trees in a forest are thus combined left-to-right, until the forest is reduced to a single tree of root S, or no further operations are possible. Part of the parsing algorithm is actually compiled into a compatibility table that states for each pair of adjacent trees the kind of fusion operation that can be applied to them. We have previously described the frequency and importance of prepositional phrases for natural language instructions to virtual actors, and the associated syntactic ambiguities they generate. Spatial prepositions attachment (e.g. N-at-N0, see Figure 2.) is based on a nearest-neighbour heuristic that states that the attachment should relate to the closest compatible noun phrase.

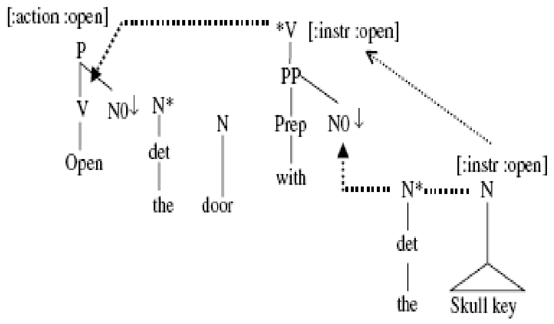


Fig. 3. Syntactic Disambiguisation with Selectional Restrictions

3.2 Integrating Syntax and Semantics

After we discussed about how natural language processing flow works, how do we implemented the parsing result into virtual shape?. That kind of question can be answered by using syntax – semantics integration process.

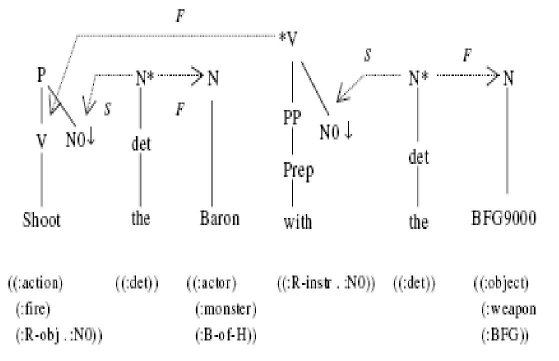


Fig. 4. Syntax-Semantics Integration

Semantic processing is carried out in parallel with syntactic parsing. Two elementary semantic operations support the construction of semantic structures. The first one is the establishment of semantic relations: it mainly corresponds to substitution in verb phrases or furcation of *V groups such as *V-with-N0, which associate actions with their instruments. The other one is the aggregation of semantic content through furcation operations, e.g. for the processing of nominal descriptions.

While the semantic relations in the semantic representation provide the argument structure for the message, there is a need to identify system actions and objects from the set of semantic features in the initial semantic representation. System actions are usually straightforward to identify on the basis of their feature descriptions, which appear on top of the semantic representation. Most of the interpretation is hence dedicated to the identification of discourse objects. The main specificity of reference resolution in this kind of agents' environment is that it cannot be entirely performed on a sole linguistic basis, as it contains indexical elements or elements referring to the agent's situation in its environment.

Objects can be identified by aggregating the semantic features of their nominal description, such as the door with a skull on it or more simply the large red door. As we have seen, these features are initially part of the semantic structure for each lexicalized tree, which represents the semantic content of the main anchor (see Figure 4). The integrated parsing process produces more complex feature structures, as features are aggregated from nominal descriptions, adverbial phrases, etc. Upon reference resolution, the NLP module can thus pass directly relevant object identifiers to the animation module. This is mainly the case for landmark objects whose designation is unambiguous (doors of a given colour, with specific patterns, specific walls or stairs, etc.). For instance, when processing the command go to the door with a skull on it, the reference resolution process can unambiguously return a single object identifier. It is thus passed to the animation system through its identifier. Reference resolution is not always possible on the basis of linguistic information only. Some designations are highly contextual, depending for instance on the relative position of the character in the virtual world. This is for instance the case for spatial expressions such as the barrel near the door on the right, which refer to the relative orientation of the character and can only be computed by accessing its actual position. As a consequence, reference resolution is a dynamic process, which is shared by the natural language interpreter and the animation system.
The overall goal of parsing is to produce a semantic structure rather than a syntactic one. In most of the cases, this semantic structure describes an action to be carried out by the agent, i.e. a case structure with the action arguments and parameters. These can be used to trigger corresponding scripts to be executed by the virtual actors.

4. From Semantics to Agents' Behaviours

The interactive story is inspired from a popular sitcom and consists for the main character "Ross" to invite the main female character ("Rachel") on a date. Each character's role is based on a plan, which is implemented using Hierarchical Task Networks (HTN) Planning. HTN planning is a knowledge-based formalism supporting forward-search refinement planning and is well-adapted to applications that have a strong knowledge content. This means that they accommodate the baseline authoring of the story rather than generate agents' behaviours from first principles. The baseline plans for the characters contain the sequence of tasks that constitute their role, though the actual choice of tasks as well as their outcome is determined dynamically and underlies story variability. For instance, Ross' tasks to invite Rachel out consist in gaining information about her, gaining her friendship, finding a way to talk to her in private, etc. The system is implemented as a real-time 3D animation using a computer.

The storytelling dimension of speech influence mostly consists in either contrasting or favouring the perceived actions of the virtual characters. In that sense the spoken input should take the form of realistic advice rather than commands and be embedded within the story. For instance, rather than saying "go talk to Phoebe" the user will say something like "Phoebe has the information you need". These more natural forms of expression, based on implicit background information, characterise the influence paradigm of speech interaction as another implementation of speech acts. The speech act nature of spoken advice can be illustrated by considering the meaning of the same sentence in different contexts. An utterance such "Phoebe is in Rachel's room" will convey different information depending on the context in which it is uttered. If Ross is trying to reach Phoebe in order to obtain information about Rachel, it will give him Phoebe's location (information provision). However, if Ross is trying to acquire the same information by stealing Rachel's diary in her room, it can also signal to Ross that he won't be able to do so, because Phoebe will object to that (warning).

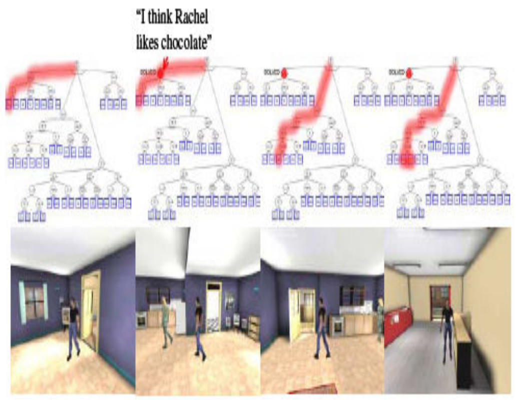


Fig. 6. Providing Virtual Actors with Information

Natural language interaction with autonomous virtual actors is a complex process in which the semantic content of user utterances has to match an agent's representations for actions. The linguistic processing benefits from the description of appropriate sublanguages, in which spatial expressions play a significant role. This makes possible to design efficient parsers integrating syntactic and semantic processing, as the ultimate goal of parsing is to produce a semantic structure for the user instruction. The original work of Webber et al. has provided a first classification of natural language interaction with an agent's plan. We have extended this work by actually relating the semantic content of linguistic input to the implementation of agents' plans. In doing so, we have however considered plans as control structures rather than as resources as initially suggested. The latter approach, while useful as a descriptive tool for analysis, is still open to too many interpretations to support a proper implementation. We have introduced a speech acts approach to the interpretation of linguistic input, which also opens several research directions for the mapping of semantic content to descriptions of the plans' operators.

5. Implementation

DiNAbot (Non-human Digital Assistant) is a program chatterbot text-based database, which means that the program code that is divided into several classes require a script as a text-based vocabulary and grammar rules for the chatterbot. DiNAbot is a chatterbot derived from the previous chatterbot Eliza (Charles Hayden), which also created using the Java programming language, although created with Java are not closing the possibility of making chatterbot using other programming languages such as C, Phyton, VB etc.. . As well as other chatterbot-chatterbot, the concept of combining method is DiNAbot parsing logic with AI implemented into the conversation, so DiNAbot able to understand the results of input from the human form of grammar and responded with a grammar that is also understandable that people stranded human conversation properly with the man. The fundamental difference between.

DiNAbot with chatterbot-chatterbot other is located on the type of language understandable. DiNAbot at stake to learn and to understand the conversation in Indonesia, also a process of response and word processing into a whole sentence. Another difference between the chatterbot common with DiNAbot is in the function, DiNAbot system is designed to become experts who answer questions for a website (in this essay, DiNAbot will be implemented to the site in the SME credit) so that the vocabulary of the DiNAbot will be limited cloning answers questions from the public about the site. To interface, DiNAbot use the applet tag as the frame.

Eliza,_Eliza class (for reasons of clarity of the original program, the authors deliberately create classes with the original name of the program) contains the rules and parameters including Rhaglennig info. In this class also declared Public Void blocks that determine the function execut Rhaglennig and the beginning of the program. In this class also declared url reserve where the script is positioned to buffer when the script called in the main failed to tag Rhaglennig (after the pilot appeared to prioritize the Java script that is running the script, which is located on a server with the same class and Rhaglennig). Block, which runs the function and Rhaglennig url is as follows:

```
static String scriptPathname = "c:\\cch\\eliza\\script";

   static String testPathname = "c:\\cch\\eliza\\test";

   static String scriptURL =
"http://www.monmouth.com/~chayden/eliza/script";

   static String testURL =
"http://www.monmouth.com/~chayden/eliza/test";

   //static String testURL =
"http://www-gbcs.mt.att.com/~cch/eliza/test";



   boolean useWindow = true;

   boolean local = false;

public void start() {
    String script = getScriptParam();

    String test = getTestParam();

    if (local) {

        script = scriptPathname;

        test = testPathname;

    }

    showStatus("Loading script from " + script);

    eliza.readScript(local, script);

    showStatus("Ready");

    if (useWindow)

        eliza.runProgram(test, this);

    else

        eliza.runProgram(test, null);

}


public boolean handleEvent(Event e) {

    return eliza.handleEvent(e);

}


String getScriptParam() {

    String script = getParameter("script");

    if (script == null) script = scriptURL;

    return script;

}


String getTestParam() {

    String test = getParameter("test");

    if (test == null) test = testURL;

    return test;

}


public String[][] getParameterInfo() {

    String[][] info = {

        {"script", "URL", "URL of script file"},

        {"test", "URL", "URL of test file"}

    };

    return info;

}


public String getAppletInfo() {

    return "Eliza v0.1 written by Aries,Cut,Robby";
}
```

6. Acknowledgements.

Research in natural language instructions and the development of the natural language interface to DOOM™ were carried out in collaboration with Ian Palmer (University of Bradford). The (ongoing) research in Interactive Storytelling is joint work with Fred Charles and Steven J. Mead at the University of Teesside.

7. References

[1] Cavazza, M. and Palmer, I.J., 1999. Natural Language Control of Interactive 3D Animation and Computer Games. *Virtual Reality*, 3, pp. 1–18.

[2] Cavazza, M., Charles, F. and Mead, S.J., 2001. AI-based Animation for Interactive Storytelling. *Proceedings of IEEE Computer Animation*, Seoul, Korea.

[3] Webber, B., Badler, N., Di Eugenio, B., Geib, C., Levison, L., and Moore, M., 1994. Instructions, Intentions and Expectations. *Artificial Intelligence Journal*, 73, pp. 253–269..

[4] Sager, N., 1986. Sublanguage: Linguistic Phenomenon, Computational Tool. In: R. Grishman and R. Kittredge (Eds.), *Analyzing Language in Restricted Domains*, Hillsdale (New Jersey), Lawrence Erlbaum Associates.

[5] Ogden, W. C. and Bernick, P., 1996. Using Natural Language Interfaces. In: M. Helander (Ed.), *Handbook of Human-Computer Interaction*, Elsevier Science Publishers (North-Holland).

[6] Zoltan-Ford, E. 1991. How to get people to say and type what computers can understand. *The International Journal of Man-Machine Studies*, 34:527–547.

[7] Microsoft. Guidelines for Designing Character Interaction. Microsoft Corporation. Available on-line at http://www.microsoft.com./workshop/imedia/agent/guidelines.asp

[8] Wauchoppe, K., Everett, S., Perzanovski, D., and Marsh, E., 1997. Natural Language in Four Spatial Interfaces. *Proceedings of the Fifth Conference on Applied Natural Language Processing*, pp. 8–11. 162 M. Cavazza

[9] Joshi, A., Levy, L. and Takahashi, M., 1975. Tree Adjunct Grammars. Journal of the Computer and System Sciences, 10:1.

[10] Cavazza, M. 1998. An Integated TFG Parser with Explicit Tree Typing, *Proceedings of the Fourth TAG+ Workshop*, Technical Report, IRCS-98-12, Institute for Research in Cognitive Science, University of Pennsylvania.

[11] Abeillé, A., 1991. *Une grammaire lexicalisée d'arbres adjoints pour le francais: application a l'analyse automatique*. These de Doctorat de l'Université Paris 7 (in French).

[12] De Smedt, K. & Kempen, G., 1990. Segment Grammars: a Formalism for Incremental Sentence Generation. In: C. Paris (Ed.) *Natural Language Generation and Computational Linguistics*, Dordrecht, Kluwer.

[13] Nau, D.S., Smith, S.J.J., and Erol, K., 1998. Control Strategies in HTN Planning: Thoery versus Practice. *Proceedings of AAAI/IAAI-98*, pp. 1127–1133.

[14] Cavazza, M., Charles, F. and Mead, S.J., 2002. Interacting with Virtual Characters in Interactive Storytelling. *Proceedings of Autonomous Agents and Multi-Agent Systems 2002*, Bologna, Italy, in press.

[15] Cavazza, M., Charles, F. and Mead, S.J., 2002. Sex, Lies and Video Games: anInteractive Storytelling Prototype. *AAAI Spring Symposium in Artificial Intelligence and Interactive Entertainment*, Stanford, USA.

[16] J. G. Carbonell and J. Siekmann, *Extraction in the Web Era – Natural Language Communication for Knowledge Acquisition and Intelligent Information Agents,* Springer. 2003.