# NON-INSTRUCTIONAL LINGUISTIC COMMUNICATION WITH VIRTUAL ACTORS

Marc Cavazza, Fred Charles and Steven J, Mead
School of Computing and Mathematics
University of Teesside, Middlesbrough
United Kingdom
E-mail {m.o.cavazza; f.charles; steven.j.mead}@tees.ac.uk

## Abstract

*In this paper, we explore a new paradigm for natural language communication with autonomous agents. While the dominant paradigm is to use natural language instructions from which the agent behaviour is generated, we investigate how natural language input can influence a pre-existing plan-based behaviour, by interfering with the various types of sub-goals in such a plan. Using as a test-bed a fully implemented interactive storytelling application based on virtual actors, we identify various forms of communicative actions and how these can influence actors' behaviours. We report early results from the use of a speech processing system, used within the storytelling application to influence the behaviour of the artificial actors.*

**Keywords:** Natural language interfaces, Speech Understanding, Virtual Actors, Interactive Virtual Environments, Artificial Emotions, Entertainment robots.

## 1 INTRODUCTION

Natural Language (NL) communication with robots (both "real" and "virtual" robots, i.e. those evolving in virtual environments) is a traditional endeavour of human-robot interaction. Natural language is the most appropriate way to exchange information at a high level of abstraction required to communicate with autonomous agents generating their own plan-based behaviour. Traditionally, human-robot communication has followed an *instructional* model, where the robot would carry actions from user instructions. This will range from executing simple instructions [8], [12] to generating complex plans for action in response to a high-level instruction [14].

We want to investigate novel forms of linguistic communication with virtual robots. Namely, how autonomous agents in virtual world, which have their own behaviour, can see their behaviour influenced, rather than instructed, by natural language input from a user.

The kind of agents we are discussing here are autonomous actors in virtual worlds. They share most of the characteristics of autonomous robots in terms of intelligent behaviour and interaction with their environment. One essential difference lies in their sensing of their environment, for in virtual worlds it is possible to grant the agent direct access to the virtual environment graphic database (though some authors have advocated the use of synthetic vision).

In the next section, after presenting the context for our experiments and our plan-based implementation of agents' behaviour, we describe how NL input can influence these behaviours at various stages of the planning process.

## 2 Plan-based Agent Behaviour

The context of our experiments is an interactive storytelling system, in which virtual actors generate variations from a global storyline by interacting with each other [5]. Following previous research in interactive storytelling, our system is essentially character-based [15]. Each character's role in the story is represented by a plan, though the plan accounts for many variations in the agent behaviour, hence in the plot. For instance, in a simple sitcom-like scenario, which serves as a test bed for our experiments, the main character, Ross, wants to invite Rachel out for dinner. His plan comprises various sub-goals, such as acquiring information about Rachel, finding a way to talk to her in private, etc. The plan is represented as a hierarchical task network (Figure 1), whose bottom nodes correspond to "terminal actions", i.e. physical actions taking place in the virtual environment. The graphic environment is entirely developed using the Unreal Tournament™ engine.

We use search-based planning to compute agents' behaviour [2] [11]. As a consequence of plan decomposability, the task network for an agent (Figure 1) can be directly searched with a graph-search algorithm such as AO* to produce a solution plan [13]. We have implemented a real-time variant of AO* [9] [10] that interleaves planning and execution, enabling the agent to re-plan new solutions as the situation is altered (e.g. due to other agents' or user interaction) [6]. As AO* is a heuristic search algorithm, the heuristic function actually selects which sub-goals to solve first or which actions to undertake. In this case, heuristics do not represent solution optimality, as alternative actions just correspond to

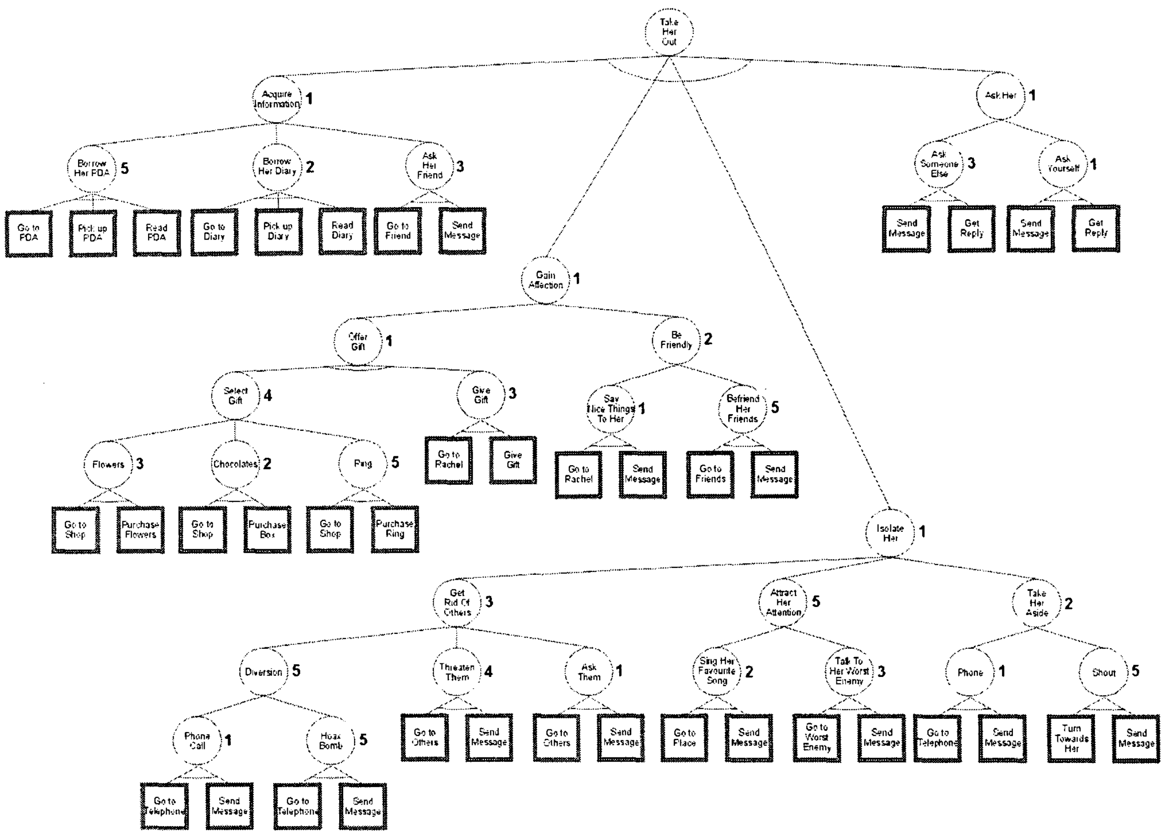


**Figure 1: Plan-based Representation of Agent Behaviour**

different instances of the storyline. What determines the choice of an action is more the "style" of an agent, i.e. a "shy" agent will not interrupt other agents' activities (e.g., reading, conversation). These heuristics can thus represent "personality profiles" for the agent, and dynamic changes in personality profile or "mood" can be reflected by dynamic updating of the heuristic values. The set of heuristic values (these are static, as the task networks are explicit graphs) can be subsumed by an "emotional status" for the agent.

In addition to these plan-based behaviours, there exist scripted reactive behaviours, which detect certain situations and alter the emotional states of the agents, hence their further behaviour. For instance, Rachel can become jealous if she sees Ross talking to Phoebe, hence changing her emotional status.

Further, the real-time plans governing character behaviours are also enhanced with situated reasoning capabilities, for instance for the treatment of exceptional circumstances that might arise dynamically from the interaction of various actors. One such example is that Ross may bump into Rachel before he has carried out the first part of his plan, which consists in getting information on her. In this case, his plan should be interrupted and he must either interact with Rachel or avoid her, but he cannot simply ignore her and keep walking. Situated reasoning is implemented through a library of scripted actions.

The user can interfere with the agents' plans, hence altering the storyline. Two modes of user intervention are *physical intervention* and *linguistic intervention*, which is the main focus of this paper [7]. In the former, the user would for instance steal the diary that Ross was about to read, forcing him to devise a new plan to get information about Rachel. While physical intervention can only interfere with resources for action at the level of terminal actions, linguistic intervention can interfere with an agent plans at various levels: sub-plan selection, sub-goal resolution, choice of terminal actions and changes to "emotional states". The various targets for intervention correspond to different parts of the task network, which by nature comprises both sub-goals and actions. We shall illustrate the various modes of intervention in the next sections, after introducing the NLP techniques required to process the NL input.

The paradigm we want to explore is that of a user influencing, rather than instructing, an agent. Executing simple commands does not involve much intelligence on the agent side, though, on the other hand, motion parameterisation to fit the instructions can be a challenging task [1]. Executing complex instructions involves devising a plan out of an utterance or a sequence of instructions, which requires sophisticated planning abilities [14]. A major difference between NL instructions and the approach presented here is that in the case of instructions, agent plans are generated as the semantics of the NL instructions. In our current approach, the NL input is influencing the behaviour of a pre-existing actor's plan, through the various mechanisms that contribute to generating a solution plan, i.e. sub-goal selection, sub-goal satisfaction.

## 3 Linguistic Processing

The user can communicate information to the artificial actors in the form of natural language statements. These can be input as textual entries or through a speech recognition system (we are using the EAR SDK from Babel Technologies™, Figures 2-3). In previous work [3], we have described parsing techniques and the interpretation of language instructions in terms of animation primitives. The system was essentially generating scripted actions, from which the agent behaviour was determined. However, we have re-used the same mechanisms for the early processing steps of the current system, i.e. syntactic and semantic parsing.

The final goal of parsing is always to produce a semantic representation for the sentence, syntax being used to aggregate semantic content [3]. The requirements on parsing depend on the need to establish functional relations. For instance, we have shown in previous work that the proper processing of spatial expressions required syntactic capabilities such as the disambiguation of prepositional phrases attachments (e.g. "Rachel's diary is on the table in the living room"). Other NL utterances require parsing for different reasons: one such example is "don't let Rachel see you with Phoebe", whose precise meaning depends on the correct allocation of functional roles.

The first step consists in parsing the input sentence to produce a representation from which the user's speech act can be identified by the system. This Natural Language Processing (NLP) step is faced with a number of difficulties, which arise from the conjunction of two factors. The first one is the need for a correct attribution of functional roles when processing advice statements such as "don't let Rachel see you with Phoebe". The second one is the inevitable occurrence of speech recognition errors that make traditional parsing problematic. In previous work, we have developed language-enabled characters that had to face only one of the above problems at a time [3]. In one of these systems, using natural language to control artificial characters in traditional computer games, reliable speech recognition or keyboard input made possible to use traditional parsing techniques. In the present setting, because the speech recognition system used is based on the definition of input templates, it is possible to use similar templates for the analysis of user utterances. This is equivalent to the use of a simple finite-state machine to parse the recognised input. The other important aspect is that the basic tokens can be integrated expressions rather than just isolated words. Each token can be associated a small set of semantic features (e.g. :advice_for_action, :character_information, :narrative_object), which in turn are assembled during the parsing process. As we shall see, these features are mainly used to identify the speech act by matching it to compatible sub-goals in an agent's plans. The identification of roles is part of the template-filling process but is also supported by some of the semantic features attached to words and expressions.

We have seen that the target for NL input can correspond to various stages of the planning process or the associated representations. It is thus necessary to identify the communicative nature of the linguistic input, i.e. whether it constitutes information, advice or direct instruction. One such possibility is to use speech acts. Speech acts are traditionally used for human-computer dialogue, but formalisms derived from speech acts are also employed to formalise agent-agent communication. They categorise the communicative action in terms of communication primitives that can be tailored to the application. In this way, they can be used to distinguish between information provision, requests, advice, etc. In this context, speech acts would provide a uniform treatment of the nature of NL input and the target representation in the task network.

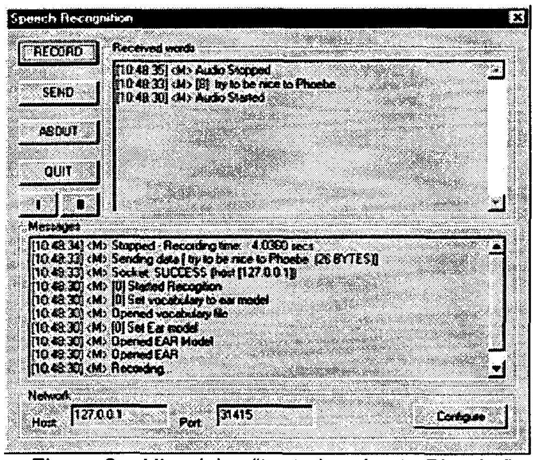
It can be difficult in the general instance to identify speech acts on the sole basis of their surface form. In the case of human-agent dialogue, we have proposed a content-based approach to speech act identification, by monitoring the updating of semantic representations [4]. In the present application, we suggest to identify the speech act by comparing the semantic structure of the utterance to the structure of sub-goal nodes in the agent's plan. Semantic compatibility between the content of the utterance and the nodes in the task network will not only determine the target but also the kind of communicative action, as this is related to the node targeted. This will be illustrated by some examples in the next section.

## 4 Results

From the description of the plan-based behaviour for virtual agents, we have seen that there are many targets for linguistic intervention. The actual influence of the linguistic input on the agent behaviour is a combination of i) the nature of linguistic input as described above and ii) its target representation in the agent's plan, sees as a resource for the agent behaviour. We now describe some of the possible interventions that can affect agents' behaviour. In each case, we identify the relevant NLP problems and the identification of the corresponding speech act on the basis of semantic matching between the NL utterance and the task network's nodes.

Firstly, while the agent's behaviour derives from it solving the various sub-goals of its task network, it is possible for the user to provide it with a solution to a sub-goal using NL input. For instance, one of the sub-goals of Ross is to acquire information about Rachel. In order to do so, he can for instance read Rachel's diary or ask her friend Phoebe. But the information Ross is looking for can be directly provided by the user (e.g. though an input such as "Rachel is free on Wednesday night"). This can be recognised by matching the semantic structure of the utterance to the currently unsolved nodes in the task network. The node corresponding to the "find information about Rachel" sub-goal is tagged with semantic categories such as (:status (:object :Rachel))[1], which can be easily unified with the semantic structure of the above utterance. As a result of this unification, the sub-goal node is labelled "solved" and this is propagated by the AO* algorithm. It should be noted that NL intervention is in this case only made possible by the interleaving of planing and execution in our real-time variant of AO*. In order to solve that sub-goal ("get Information about Rachel") the agent would normally carry out an action its environment, such as walking to Rachel's diary. This action (or, for that matter, alternative ones) takes a significant amount of time to be completed, during which the user can provide NL input. Once the node has been marked as solved by the agent it cannot be influenced any longer by NL input, unless that input can also affect future, not yet solved, nodes.

Sometimes an agent goal can be solved through interaction with other agents. For instance, if Ross' sub-goal is to isolate Rachel from the rest of the group, this can be achieved by asking others to leave (something a "shy" Ross might be unable to do). If the user asks e.g. Phoebe to leave, Ross' sub-goal will be satisfied. In this case there is no requirement to identify the speech act at plan level, as it is not directed towards Ross' plan.



```
[10:48:35] <M> Audio Stopped
[10:48:33] <M> [8] try to be nice to Phoebe
[10:48:30] <M> Audio Started
```

```
[10:48:34] <M> Stopped - Recording time: 4.0380 secs
[10:48:33] <M> Sending data [ try to be nice to Phoebe [26 BYTES]]
[10:48:33] <M> Socket: SUCCESS (host [127.0.0.1])
[10:48:30] <M> [0] Started Recogition
[10:48:30] <M> [0] Set vocabulary to ear model
[10:48:30] <M> Opened vocabulary file
[10:48:30] <M> [0] Set Ear model
[10:48:30] <M> Opened EAR Model
[10:48:30] <M> Opened EAR
[10:48:30] <M> Recording
```

Host: 127.0.0.1 Port: 31415

Figure 2a: NL advice "try to be nice to Phoebe"

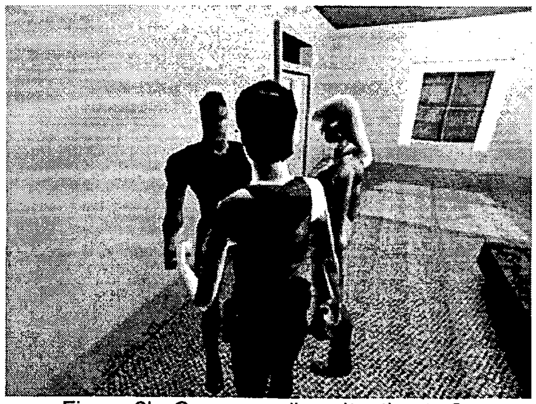


Figure 2b: Corresponding situation to 2a

1 Not for the purpose of computing a solution plan, but strictly in order to determine the target of NL input.

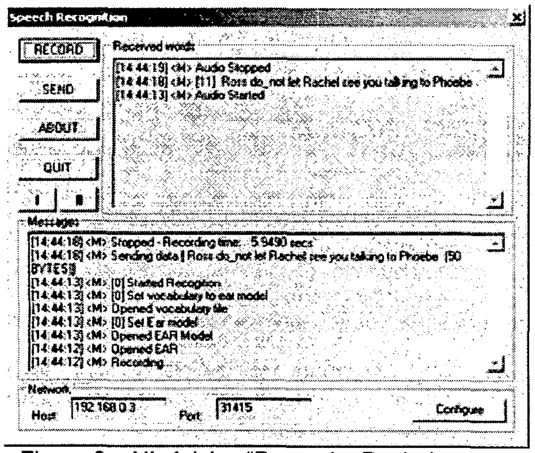
Apart from the above example, it is still possible for the user to directly instruct an agent to carry out a particular action. For instance, by instructing Ross to move to a certain location (e.g. living room), the user will make him meet Phoebe, which could create a comic *quiproquo* situation if he is spotted by Rachel. The standard mechanisms for carrying out direct action do not differ from those described in [3].

Another form of influence consists in providing information that might influence the choice of the next action. For instance, to satisfy the goal of "acquiring information about Rachel", one of Ross' options is to go and read her diary (Figure 1). It can be specifically instructed to do so (which might override its standard personality profile) if the user mentions the diary.

NL input can also be used to alter the emotional status of one of the actors. This is implemented by changing the personality profile, i.e. by updating the heuristic values associated with the task network. The semantics of some NL input can be associated emotional connotations that can directly trigger changes in emotional/personality profile. This is implemented straightforwardly by associating updating procedures to emotional semantic features. A more complex case however consists in encouraging (resp. discouraging) statements that require some inference to gain an emotional interpretation, such as "Rachel does not seem to be available", etc. Such kind of NL interpretation should be related to the agent's goals or some form of intentional model (though our plan-based formalism is not based on intentions).

Certainly the most challenging, but interesting, use of NL input is to provide high-level advice that can guide the agent action throughout its evolution in the virtual environment. This form of generic policy statements has been termed *doctrine statements* by Webber et al. [14]. One such example would be "try to be nice to Phoebe this time" or "don't let Rachel see you with Phoebe". Clearly, this corresponds to rules that would have to be matched against the situation at every possible decision-taking point in the computation of an agent's behaviour. For instance, when addressing the "talk to Rachel in private" sub-goal, if Rachel is talking to Phoebe (Figure 2a and 2b), Ross can choose to interrupt their conversation and ask Phoebe to leave. This might backfire by changing Rachel's emotional state as well. The various possible actions in Ross plans are categorised as "friendly" or "rude", and these categories can be target for NL advice. However, strict compliance with the rule assumes that the agent also identifies the other parties, in that case Phoebe. This might lead to significant difficulties at every step of processing, and we have not found a generic solution to this problem. The processing of doctrine statements is quite difficult in the general case, as it sometimes involves generating persistent behaviour that matches dynamic situations. However, there are several empirical mechanisms that can simplify their interpretation. In other cases, doctrine statements can have procedural interpretations. The advice "stay out of reach of Monica" can be successfully implemented by activating a pre-defined behaviour that *avoids* Monica. This is simplified by the fact that avoidance behaviours can be part of the set of standards behaviours that are just parameterised with the character to avoid. In other words, the procedural semantics of doctrine statements takes advantage of the limited set of pre-defined complex actions (such as avoiding or hiding from another character) introduced by the use of situated reasoning. Another possible solution is to under-specify the requirements, such as making Ross to act nicely with everyone by changing its personality profile, so that its plan will select the actions tagged as "friendly". This is a form of under-specification of the meaning of the NL advice that can provide a computationally economic solution, though not a fully satisfactory one.

The rationale for under-specification also lies in the limited rationality of the agent: it might not have enough reasoning capabilities to analyse every situation at the proper level of granularity. For instance, the advice "don't let Rachel see you talking to Phoebe" can be enforced by either meeting Phoebe hidden from Rachel, or simply staying away from Phoebe to avoid potential trouble. Hiding assumes that the agent determine in real-time whether Rachel can see him, etc. On the other hand, staying away from Phoebe restricts the agent further solutions, but avoids the main problems the advice was targeted to, which is to prevent Rachel from becoming jealous (Figure 3a and 3b).



```
[14:44:19] <M> Audio Stopped
[14:44:18] <M> [11] Ross do_not let Rachel see you talking to Phoebe
[14:44:13] <M> Audio Started
```

```
[14:44:18] <M> Stopped - Recording time: 5.9490 secs
[14:44:18] <M> Sending data [ Ross do_not let Rachel see you talking to Phoebe (50
BYTES)]
[14:44:13] <M> [0] Started Recognition
[14:44:13] <M> [0] Set vocabulary to ear model
[14:44:13] <M> Opened vocabulary file
[14:44:12] <M> [0] Set Ear model
[14:44:13] <M> Opened EAR Model
[14:44:12] <M> Opened EAR
[14:44:12] <M> Recording...
```

Host: 192.168.0.3 Port: 31415

Figure 3a. NL Advice "Do not let Rachel see you talking to Phoebe"

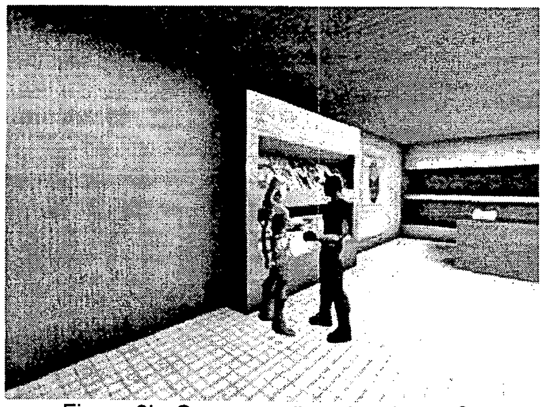


Figure 3b. Corresponding situation to 3a

## 5 Conclusions

We have introduced a new form of NL communication with autonomous virtual agents, based on influencing plan-based behaviours at various sages of the planning process. We have chosen to experiment from a fully implemented interactive storytelling system in order to better study the NL requirements. The NL components are still under development, though we have obtained early results for some of the examples presented here. More challenging is the generic processing of doctrine statements, for which we have not devised a generic approach at this stage, under-specification having been used as an early solution. There are two kinds of limitations for the system we have described here. One is naturally the overall processing performance, which is strongly influenced by the accuracy of speech recognition. The other one is more fundamental: character behaviour can only be influenced by generic advice and it is not possible to refer to potential situations or to a specific course of action when formulating recommendations. Implementing this feature might even require extensive modifications to the planning system itself, providing a level of meta-knowledge representing the actual progression of the plan, which is more of a long-term perspective. It might be the case that in the absence of common sense reasoning abilities on the agent side, future solutions will comprise a set of diverse approaches to the processing of NL information.

## References

[1] Badler, N., Zhao, L., Cosa, M., Vogler, C. and Schuler, W., 2000. Modifying Movement Manner Using Adverbs. *Autonomous Agents Workshop on Communicative Agents in Intelligent Virtual Environments*, Barcelona, Spain.

[2] Bonet, B. and Geffner, H, 1999. Planning as Heuristic Search: New Results. *Proceedings of ECP'99*, pp. 360-372.

[3] Cavazza, M. and Palmer, I., 1999. Natural Language Control of Interactive 3D Animation and Computer Games. *Virtual Reality*, 4, pp. 85-102.

[4] Cavazza, M., 2000. From Speech Acts to Search Acts: a Semantic Approach to Speech Act Recognition. *Proceedings of GOTALOG'2000*, Gothenburg, Sweden, pp. 187-190.

[5] Cavazza, M., Charles, F., and Mead, S.J., 2001a. Characters in Search of an Author: AI-based Virtual Storytelling. *First International Conference on Virtual Storytelling, to appear.*

[6] Cavazza, M., Charles, F., Mead, S.J. and Strachan, A., 2001b. Virtual Actors' Behaviour for 3D Interactive Storytelling, *Eurographics 2001* (short paper), *to appear.*

[7] Charles, F., Mead, S. and Cavazza, M., 2001. User Intervention in Virtual Interactive Storytelling. *Proceedings of VRIC 2001*, Laval, France, to appear.

[8] Ferre, M., Macias-Guarasa, J., Aracil, R. and Barrientos, A., 1998. Voice Command Generation for Teleoperated Robot Systems. *Proceedings of the 7th IEEE International Workshop on Robot and Human Communication (ROMAN'98)*, pp. 679-685.

[9] Nilsson, N.J., 1980. *Principles of Artificial Intelligence*. Palo Alto, CA. Tioga Publishing Company.

[10] Pearl, J., 1984. *Heuristics: Intelligent Search Strategies for Computer Problem Solving*. Reading (Massachusetts), Addison-Wesley, 1984.

[11] Pemberton, J.C. and Korf, R.E., 1994. Incremental Search Algorithms for Real-Time Decision Making. *Proceedings of the 2nd Artificial Intelligence Planning Systems Conference* (AIPS-94).

[12] Shinayama, Y., Tokunaga, T. and Tanaka, H., 2000. "Kairai" – Software Robots Understanding Natural Language. *Proceedings of the Third International Human-Computer Conversation Workshop*, Bellagio, Italy, pp. 158-163.

[13] Tsuneto, R., Nau, D. and Hendler, J., 1997. Plan-Refinement Strategies and Search-Space Size. *Proceedings of the European Conference on Planning*, pp. 414-426..

[14] Webber, B.N., Badler, N.I., Di Eugenio, B., Geib, C., Levison, L., and Moore, M., 1994. *Instructions, Intentions and Expectations*, IRCS Technical Report 94-01, University of Pennsylvania.

[15] Young, R.M., 2000. Creating Interactive Narrative Structures: The Potential for AI Approaches. *AAAI Spring Symposium in Artificial Intelligence and Interactive Entertainment*, AAAI Press.