# An efficient synthetic vision system for 3D multi-character systems *

Miguel Lozano[1], Rafael Lucia[2], Fernando Barber[1], Fran Grimaldo[2], Antonio Lucas[1], Alicia Fornes Bisquerra[1]

[1] Computer Science Department, University of Valencia,
Dr.Moliner 50, (Burjassot) Valencia, Spain
[2] Institute of Robotics, University of Valencia,
Pol. de la Coma s/n (Paterna) Valencia, Spain
{Miguel.Lozano}@uv.es

**Abstract.** This paper deals with the problem of sensing virtual environments for 3D intelligent multi-character simulations. As these creatures should display reactive skills (navigation or gazing), together with the necessary planning processes, required to animate their behaviours, we present an efficient and fully scalable sensor system designed to provide this information to different kinds of 3D embodied agents (games, storytelling, etc). Inspired in Latombe's vision system [5], as recently presented by Peters [1], we avoid the second rendering mechanism, looking for the necessary efficiency, and we introduce a fully scalable communication protocol, based on XML labelling techniques, to let the agent handle the communication flow within its 3D environment (sense + act). The synthetic sensor system presented has been tested with two plausible local navigation formalisms (Neural Networks and Rule based System), whose models and results have been also reported.

## 1 Introduction

Artificial humans or 3D intelligent characters normally display their intelligence through their navigation skills, full-body control, and decision-taking formalisms adopted. The complexity involved in these kinds of agents, normally suggests designing and executing them independently of the 3D graphics generator, so the agent could be focuss on their control problems (figure 1). As Thalman pointed out [4], the major challenge now is to integrate in the same framework a set of various techniques which are required to simulate behaviours of virtual humans. We are working in that direction, that is, towards the creation of a robust simulation framework for Intelligent Virtual Environments [2], where different 3D embodied agents are able to sense their environment, to take decisions, to navigate in a dynamic scenario and finally to carry out the necessary motor actions which will animate their behaviours. This can be considered as a requirement for 3D multi-character simulations, as described in storytelling domains.

* partially supported by the GVA-project CTIDIB-2002-182 (Spain).

In this paper we propose a general way to handle the relationship between the agent and its 3D environment. Sensing a 3D environment normally relies on the behavioural model supported by the simulation environment. As classical time based environnments with a *scenegraph* manager (OpenGL-Performer, Java3D, etc), lacks a consistent communication protocol between the geometry encapsulated in their nodes, we have adopted an *event driven* 3D environment (Unreal Tournament [3]) which offers a general mechanism to manage the necessary environmental changes, as it has been used previously in this context [9]. In this way, we consider storytelling domains as an adequate simulation framework for 3D intelligent characters or 3D multi-character system, so from this particular point of view, the agent sensory systems designed should be able to manage the two information sources, typically considered when studying artificial humans. Firstly, the **low level**, as the information required for reactive behaviours (visual attention, reactive navigation, etc), and secondly, the **high level** or symbolic one, which represents the visible state perceived from any object/agent accepted by the vision test (eg.: actor1:free, door1:open, etc), or another external stimuli that the agent should know, in order to be consistent with the story or the simulation (audio events, user-interactions, etc).

The sense-act system implemented has been designed according to these requirements, and has been integrated in the Unreal Tournament (UT) graphics engine, where it will be tested as an efficient method to support 3D multi-character navigation tasks.

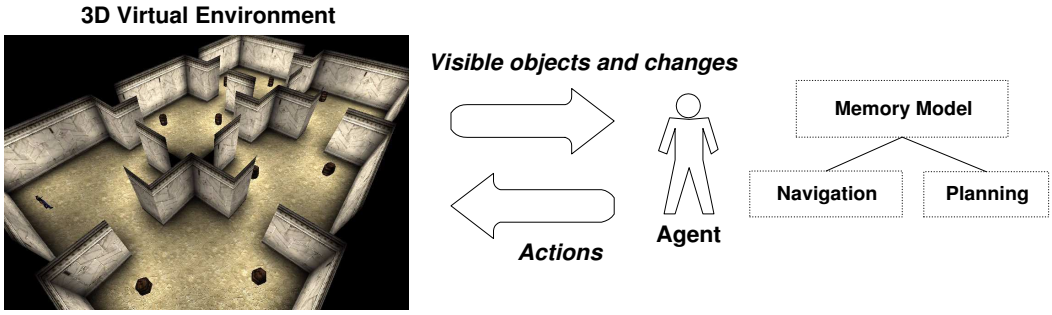


**Fig. 1.** Agent scheme

The paper has been organized mainly in three sections. Firstly we review previous works on 3D characters or humanoids which reproduce a local information flow. Secondly we explain the synthetic vision system implemented and its integration in the Unreal Tournament engine. Finally we briefly present the Neural Network and Rule based system formalisms adopted to test the vision system in a 3D dynamic environment, where both types of agents will navigate only according to the local information provided by their vision systems.

[3] Epic Games, USA

### 1.1 Related work

The number of 3D characters or humanoids which reproduce a local information flow, is still reduced, as these kind of agents can normally access the complete visual data base. This situation has motivated a higher exploration of global approaches for human character navigation. From the games community, the problem of animating 3D intelligent characters has been approached mainly with global heuristic search algorithms (*A* family*) in 2D-grids [6]. On the other hand, the characters will receive the necessary environmental events (collisions, other NPCs,...), to be informed about the state of environment at a higher representational level. This behavioural side is typically managed by finite state machines (FSM), which normally support the reactive character behaviours.

Tu and Terzopoulos have researched artificial sensing with individuals and schools of fish [12]. They have used the double-rendering and false-color techniques to simulate the synthetic vision processes. This model has been also ported to virtual humans [11]

Renault et al. introduces a synthetic vision module for animating 3D actors, focused mainly on navigation tasks. The scene is again rendered from the agent's point of view, and the output is stored in a 2D array. This array stores first the pixel at that point, secondly the distance from the actor, and lastly an identifier for any object at that position. As the goal of this vision system is to allow the actor to navigate in a corridor with obstacles and actors, they used DLA's (Displacement Local Automata), which encapsulate the necessary knowledge to face normal navigation situations [3].

Noser et al. extended the previous work introducing new memory mechanisms, although the vision processes remain the same. They consider global navigation with a simplified map for pathplanning tasks in a static world, so in order to deal with the environmental changes, they introduce a local navigation system, which will receive the navigation sub-goals from the global system. As the local navigation formalism has no environmental model, the actor will create an octree data structure to represent its long-term memory during simulation [4].

Kuffner and Latombe present a perception-based navigation system for animated actors. A record of perceived objects and their properties are kept as the character explores an unknown virtual environment, providing a relatively compact and fast representation of the actor's internal world, as recently presented in [1]. Again a combination of global and local systems lets the agent explores 3D environments.

## 2 Synthetic Vision and Perception

Synthetic vision has been considered as the process of capturing the list of visible *actors* (objects or agents) from the agent point of view. Although the possibility of directly accessing the database is simple and fast, it suffers from scalability and realism problems. On the other hand, the application of a second rendering from

the agent viewpoint using a false coloring object identification, offers an elegant way to achieve this list (simulating a human retinal image) [12][3][5][1]. However, it is normally the bottleneck of the system, so finally a single or small number of agents can be simulated in real time. To avoid this bottleneck, without losing visibility consistency, we are simulating inter-*actors* occlusion in local domains based on simple and fast geometric techniques extracted from [10]. In this way, the synthetic vision process has been mainly undertaken in two phases.

Firstly we extract the nearest *actors* from the Unreal Tournament 3D environment map area (UT). Secondly, as NPC's during games are normally allowed to see through walls and other objects, we have filtered this `UTLst` (see figure 1) removing the occluded *actors* from the agent point of view (`Visible List`). Both processes have been fully integrated in UT through the necessary scripts to finally provide the `Observation List` for the main agent process.

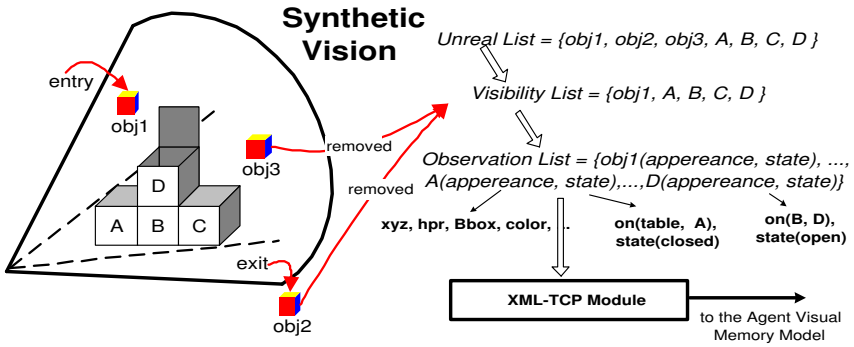


**Fig. 2.** Synthetic vision model

### 2.1 Integrating sense/act agents in UT

Assuming the general procedure for distributing all the simulation changes in the 3D environment (event-driven), and in order to be consistent with the behavioural model of such graphic engine, we have implemented the following communication modes to let the agent deal with its vision system:

- **Event-driven mode** (on property change): Vision can inform the agent anytime an actor or an object property has changed. To support this mode, the synthetic vision modules compares, at regular time intervals, the current `Observation List` against the previous one. In this way, when the vision system detects a change inside the agent field of view, that is, when a visible *actor* exits, a new *actor* enters, or a visible *actor's* property is updated, the correspondent XML token is sent to the main agent process.

- **Fixed time mode** (on time): The vision system can be programmed to send its current `Observation List` at fixed time intervals.
- **Agent's demand** (on demand): As the sensor system presented will be always managing the current environmental information, it can provide this information when needed.

In this way we let different type of agents handle autonomously the sensory information perceived from their fields of views. Actions, observations and interaction modelling are normally big agent problems. We have introduced easily scalable XML labelling techniques to be able to manage a general and robust mechanism for the knowledge embedded in 3D multi-character simulation systems. Currently we are applying it to:

- **The communication mode**: Any agent formalism provided with a typical socket connection (IPaddr, Port), is allowed to instantiate its vision module in UT, simply by sending the XML token defined for this purpose. The vision system is continually listening to the map entry port, so when an agent wants to participate in a simulation, a new vision system is created in UT to handle an independent communication flow between them. This initial message will also include the communication mode (on events/on time/on demand), together with the agent properties interested in receiving information (appearance/state/all).
- The **actor's properties**: We consider the semantic list approach adopted, similar to [5] [1], as an adequate technique to manage the necessary knowledge for 3D real time multi-character simulation systems. This knowledge is currently located in the *actor* properties, and is divided into two main groups, `appearance` an `state`. Appearance include position, orientation, speed, vertex list (bounding box), size, etc. State representation covers the visible semantic information the *actor* is representing and will be useful for the agent planning tasks. Currently, it is based on tuples $< atom_k, val_1, val_2 >$, eg. box::$< state, closed, NULL >$, or box::$< on, table, a >$, although for current navigation purposes this is logically disabled. As previously cited and according to the communication mode defined, the *actor* properties changed can shape an XML token which will be finally sent to the agent (Figure 2).
- **The agent actions**: The communication established between the agent and its 3D environment can be also used to inform the motor agent system about the actions to be carried out. We have used classical STRIPS representation for action modelling, in previous experiments focussed on Heuristic Search Planning (HSP) for character behaviour animation [8]. According to this, we are using the representation previously introduced which is sufficient for simple keyframing based actions. For instance, if an agent tries to open a box, the UT box-script, which controls this action, should know its effects `(state(open))`, so, as it control the action execution it will be allowed to produce a change on the box state properties, depending on action results (success/failure). Furthermore, the labelling technique used will let us include some *execution conditions* in character actions, which will be evaluated

during action execution. Although this paper has been focussed on the efficiency of the vision system, tested for navigation tasks, however we expect to integrate the HSP system designed, for resolving storytelling problems as soon as possible.

The communication model presented is simple (text based tokens) and can be easily scalable to new information resources, such us audio, non-verbal communication, or another virtual human environmental perception requirements. Furthermore, the parsing of XML text sequences is a robust technique which can be easily integrated in both logic based agent systems (Prolog, Lisp) and procedural languages (C++). Finally, this semantic-list approach clearly facilitates the integration of layered agent systems, where both reactive and deliberative layers can work in parallel using the information provided by their sensor and memory models.

## 3 Reactive navigation formalisms

In order to test the visual information system designed, we have implemented two reactive navigation formalisms.

**Feedforward 3-layered Neural approach** [7]: Angular velocity maps have been used as it provides for an accurate vision memory model for navigation tasks. Basically they are collections of classical neurons, which represent all the possible navigation directions from an egocentric view. As figure 2 resumes, parsing the bounding box of each agent observation a couple of vectors (rays), which cover any plausible obstacle from its point of view, can be extracted. This information is enough for the 3-layered (G-C-M) feedforward design presented in [7], where the goal following layer (G) will compute the goal direction, the correction layer (C)will manage the forbidden directions anytime, and lastly, the motor layer (M) will give us the current agent direction, as a combination of previous layer activities. This feedforward metaphor could be also used to model visual attention, as the correction layer could be used, in this case, to represent the degree of interest in each direction (according to the perceived object properties, such us, position, size, color, etc).

**A rule based system** based on qualitative perceptions has been also implemented.The rule based system we are using for navigation works with the classical cycle of simulation: sense - think - act. The way it communicates therefore with the UT world is on demand and not driven by events like the neural network approach. Every time a cycle begins, the system sends a petition to the UT world for sensor information. The system processes this information and sends back to the UT world the action to be performed.

All the sensor information received from UT is stored in the working memory of the rule based system. We have used a qualitative approach for the representation of this knowledge so the rules are more simple to write and to understand. The information received is translated to a qualitative representation from the agent viewpoint before it is stored in the working memory. Once stored, the

inference engine is launched and the action to be carried out is deduced from the sensor information. For the implementation of the agents with a rule based system we have used Prolog, although the syntax for the rules is similar to the CLIPS rules, so they can be easily ported.

## 4 Results

Figure 3 summarizes the results obtained. In the first one (a), we are comparing the two navigation formalisms briefly introduced, with a single character which has to perform a 6 lentgh plan. Both navigation models, the Neural Network agent (NN) and the Rule based one (RB), handle a simple Short Term Memory model by simply adding a timeout to the received observations. As the figure 3 shows, this is enough to solve some local minima situations, as characters can remember previous object locations for solving the current navigation decision. Figure 3b mainly shows the internal memory model (from rays to 2D boxes) managed by the NN agent. In this case we have introduced a small 3D pylon to show the plan sub-goals (1-6), so the agent has to avoid them too. Figure 3c and 3d shows multi-character results

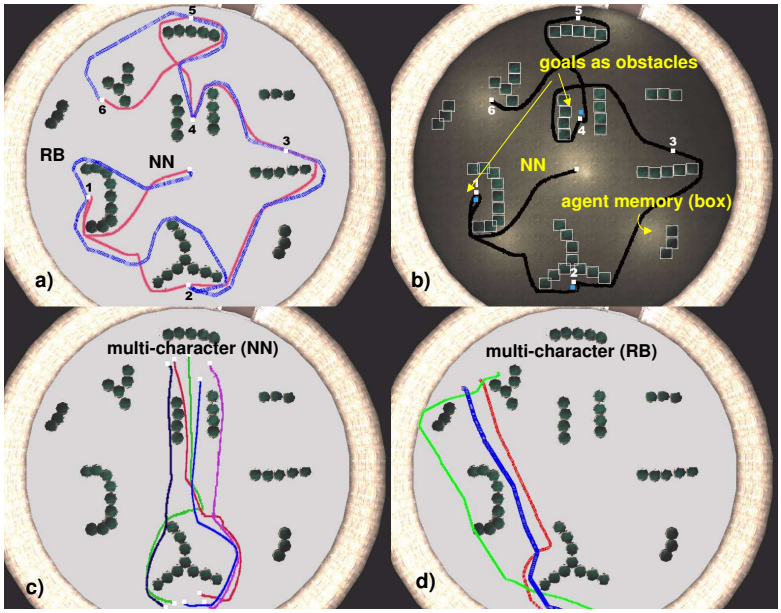


**Fig. 3.** Navigation results

## 5 Conclusions and Work in progress

The sensory system presented constitutes an efficient approach for 3D multi-character simulations, as it let us distribute the information represented in the 3D environment according to normal vision processes. In this way, a finite number of agents can share a common and unknown 3D environment managed by the graphic server. All the results has been obtained in a dual PC-PentiumIV with a standard 3D graphics accelerator board.

The full integration of the HSP system together with the neural network design roughly described here, constitutes the work actually in progress. As a result of this integration, the intelligent characters created will be able to perceive its environment, and animate autonomously their behaviours for solving different *storytelling* problems.

## References

1. C. Peters, C. OSullivan: Synthetic Vision and Memory for Autonomous Virtual Humans. Computer Graphics Forum Volume 21, Issue 4 (November 2002)
2. Aylett R. Luck M. -"Applying Artificial Intelligence to Virtual Reality: Intelligent Virtual Environments". Applied Artificial Intelligence, 2000.
3. Renault, O., D. Thalmann, and N.M. Thalmann.: A vision-based approach to behavioural animation. Visualization and Computer Animation, Vol. 1, pages 18-21, 1990. 2, 3
4. H. Noser, O. Renault, D. Thalmann, N. Magnenat Thalmann, *Navigation for Digital Actors based on Synthetic Vision, Memory and Learning*, Computers and Graphics, Vol.19, No1, 1995, pp.7-19.
5. J. Kuffner and J.C. Latombe. " Fast Synthetic Vision, Memory, and Learning for Virtual Humans". Proc. of Computer Animation, IEEE, pp. 118-127, May 1999.
6. M. Cavazza, S. Bandi, I. Palmer *Situated AI in Computer Games: Integrating NLP, pathplanning and 3D animation.*, AAAI Spring Symposium on AI and Computer Games, 1999.
7. M. Lozano, J. Molina. *A neural approach to an attentive navigation for 3D Intelligent Virtual Agents.* IEEE System Man and Cybernetics October 2002, Tunisia.
8. M. Lozano, Mead, S.J., Cavazza, M. and Charles, F. *Search Based Planning: A Model for Character Behaviour.* Proceedings of the 3rd on Intelligent Games and Simulation, GameOn-2002, London, UK, November 2002.
9. Michael Young *An Overview of the Mimesis Architecture: Integrating Intelligent Narrative Control into an Existing Gaming Environment* In The Working Notes of the AAAI Spring Symposium on Artificial Intelligence and Interactive Entertainment, Stanford, CA, March 2001.
10. Joseph O'Rourke.: Computational Geometry in C (Second Edition). Cambridge University Press ISBN: 0521649765.
11. T.F. Rabie D. Terzopoulos. "Active perception in virtual humans,". Proc. of the Vision Interface (VI'2000), Montral, Quebec, Canada .
12. X. Tu and D. Terzopoulos, "Artificial Fishes: Physics, Locomotion, Perception, Behavior", Computer Graphics, SIGGRAPH 94 Conference Proceedings, pp. 43-50, July, 1994