Modeling Virtual Agent Behavior in a Computer Game to Be Used in a Real Enviroment

Catalina Roncancio, Jaime Gómez G-B, and Eduardo Zalama

Abstract. Our long term goal is to develop autonomous robotic systems that have the cognitive abilities of humans, including communication, coordination, adapting to novel situations, and learning through experience. Cognitive architectures as theory of the fixed mechanisms and structures that underlie human cognition are the actual mechanism of making a software implementations of a general theory of intelligence. The proposed system incorporates the hypothesis behind cognitive architectures like Soar to model our particular content, an autonomous character and its cognitive processes in normal working situations as hotel bellboy, and simulated in a virtual environment. Through this work, we proposed introduce game development as a test bed for our application.

Keywords: cognitive modeling, computer game, programing by demosntration.

1 Introduction

The growing interest on cognitive models have let the integration on different disciplines, like psychology, linguistics, anthropology, and artificial intelligence, to name just a few, that have characterized the cognitive systems as systems which exhibit adaptive, anticipatory, and purposive goal-directed behavior. We can find various paradigms of cognition, each taking a significantly different stance on the nature of cognition, what a cognitive system does, and how a cognitive system should be analyzed and synthesized [3, 4]. Each paradigm has a history of asking certain types of questions and accepting certain types of answers. And that, according to Allen Newell, is both an advantage and a problem. What we finally need is to work with unified theories of cognition (UTCs) [12] and this is what the cognitive architectures attempts to be [13]. To understand how any computational architecture

Catalina Roncancio · Jaime Gómez G-B · Eduardo Zalama
CARTIF, Parque Tecnologico Boecillo parc. 205 Valladolid, Spain
e-mail: {catron,jaigom,eduzal}@cartif.es

Y. Demazeau et al. (Eds.): Trends in PAAMS, AISC 71, pp. 623–630.
springerlink.com

works, we need to used it to model some behavior [5] (an architecture by itself does nothing, it requires content to produce behavior). Lets consider the next scenario: Sacarino(also named AiE-Agent) is a bellboy robot that will work in a hotel environment. In the field of service robots, there has been an increase in the necessity of developing assistant robots capable of interacting with users and undertaking real life tasks. Notwithstanding the general consensus about the necessity of developing and exploiting the potential of service robots, it stills remains a manifold challenge to do so today. Our assistant bellboy robot for hotels will be designed and developed capable of providing the following services and tasks:

- Accompanying the guests to their rooms.
- Explaining the services available in the room and the hotel (meals, laundry, etc).
- Carrying food, drinks, equipment, newspapers to the rooms.
- Dialogging with guests in defined contexts, taking care of orders, providing useful information (tourist and meteorological information, news, etc.), and sending/receiving messages to/from the reception desk.

The proposed system incorporates game design and artificial intelligence to simulate autonomous characters and their cognitive processes in normal working situations. The combination of computer games with virtual agent systems gives the opportunity to offer virtual environments for improving training and decision-making with virtual autonomous agents in everyday surroundings. One important advantage to bring the real application in to a simulation is that we can exhibit a high degree of anthropomorphism [7] with highly expressive interfaces that are easily adjusted and personalized for each user at a fraction of the cost of a robotic interface. Our paper is structured as follows. Section 2 presents some related work. Section 3 Explains the details by designing a game within the XNA Game Studio. Section 4 describes our approach to modeling the bellboy game ,and shows how we used our model to generate code and section 5 discusses the benefits of our approach and concludes.

2 Related Works

A number of noteworthy architectures for behavioral animation of autonomous characters have been proposed. In many of these techniques, characters behave autonomously by choosing actions through a behavioral model: an executable model that defines how a character should react to its environment. Our approach does not model the behavior explicitly. The behavior emerges based on the interaction. The use of visual modeling environments is not new to the gaming industry but if its use as a test bed of real application. The main objective of developing such systems is be able to work in parallel with the real application and then yet anticipate the most probably simulated situations before getting to work with the robot. For example, the Soar architecture approach is also applying on computer games design [15]. Through this work, they investigate some designs that facilitate tractable reinforcement learning in symbolic agents developed using Soar architecture operating in a complex domain, Infinite Mario. A reinforcement Learning domain developed for Reinforcement Learning Competition 2009, as is a variant of Nintendo Super Mario.

Another related work is [10], where they proposed a system that incorporates virtual reality and artificial intelligence to simulate virtual autonomous characters and their cognitive processes in dangerous working situation on an industry. The cognitive agents are enriched with a planner for selecting actions according to goals, the environment and to the personal characteristics of the agents (time pressure, caution, tiredness, hunger). A work that comes close to our research has been done by Jrg Kienzle et al. [6] on modeling computer games, and Non-Player Characters, to reason about the behavior of a tank pilot for the EA Tank Wars competition, in which Computer Science students compete against each other by writing artificial intelligence (AI) components that control the movements of a tank.

3 Execution Platform

On this first stage, our challenge is to develop a virtual environment that will support the actions of the virtual agent represented by a non-player character in a computer game. This will let us develop a demonstrator where users can interact with our virtual robot service and obtain information about the hotel. The modeling of our robot service Sacarino in a game with appropriate abstraction level using an appropriate modeling language has many advantages: (i) Programming-by-demonstration interface. (ii) Writing consistent, re-usable and efficient AI code. (iii) Provides the user with a social interface that acts as a representative of the services the environment offers. And (iv) enhance human-machine interaction. The game is an interesting tool, as in the real application, it is requires an agent to reason and learn at several levels; from modeling sensory-motor primitives to path-planning and devising strategies to deal with various components of the environment. We propose the integration of Soar theory to define the agent's behavior with a virtual model of the environment (by XNA). The description of the system will be based on statecharts [2, 17], a combination of state diagrams and class diagrams that interpret the differents modules of the system.

3.1 General Game Structure

In the recently years the design of Computer Game becoming an interesting test bead for research in Artificial Intelligence and Machine Learning [14, 15]. Traditionally, the research has concentrated on learning in Board game. A good example of a board game that has seen many successful computerized implementations is Chess [11]. However, recently computer and video games (Real-time games) have received increased attention [8, 16] because they present challenges which are close to real world problems like that of the enormity of information in a highly self contained and circumscribed environment. Real-time games requires users controls one character (or a small number of characters), and plays within a game environment against a set of computer controlled characters (or in multi players games against characters controlled by other players). In such games, the term artificial intelligence is used to designate the algorithms that specify the behavior of computer-controlled game characters, often also called non-player characters (NPC). The ultimate goals is to make the NPCs own actions and reactions to game events seem as intelligent and natural as possible. The central logic for every game includes preparing the environment where the game will run, running the game in a loop until the game ending criteria is met, and cleaning up the environment [1]. The idea of having the main program logic running in a loop is crucial for a game, because the game needs to keep running whether or not it has user interaction. This doesn't happen with some commercial applications, which only do something in response to user input. Comparing XNA with the old way of developing game we can see that the Game project type provides us with a ready made basic game structure, so we can start by including our game-specific code, and focus in the main target (Intelligent behavior and improve interaction). To illustrate the power of our approach, we show in the following sections how we modeled the bellboy robot behavior in a game.

4 Modeling Game

In games or simulations, a character perceives the environment through his senses or sensors, and reacts to it through actions or actuators. For instance, our character might look for the presence of a person and if its seen subsequently decide to make an action. The basic architecture is described in Fig. 1. In the perception module we find the information of processing video and voice recognition. The Cognitive module based on Soar [9], performs the behavior planing separating memories for descriptions of the current situation and its long-term knowledge. The current situations, including data from sensors, active goals, and active operators is held in the working memory. The knowledge that exists independent of the current situation is held in the architectures long-term memory (LTM). LTM is not directly available, but must be searched to find what is relevant to the current situation of the bellboy. Soar distinguishes three different types of LTM: procedural, semantic, and episodic. Procedural knowledge is primarily responsible for controlling behavior and maps directly onto operator knowledge. Semantic and episodic knowledge usually come into play only when procedural knowledge is in some way incomplete or inadequate for the current situation. The Behavior System is the processing component that generates behavior out of the content that resides in the long-term and working memories. The purpose of the Behavior system is to select the next operator to apply. A goal directed behavior corresponds to movement in a problem space from the current state to a new state through the application of an operator to the current state.

4.1 Modeling the State of the Bellboy

The visual scene is divided into a two-dimensional [16 x 10] matrix of tiles. Each tile (element in the scene) can have one of the many values that can be used by the agent to determine it the corresponding tile in the scene is a obstacle (objects), or not, a user, or a receptionist, etc. For every visible user, the agent is provided with

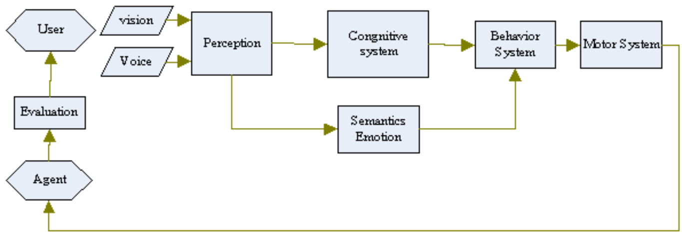

Fig. 1 Architecture

the type of the user, its current location, and its speed in both x and y direction. Once it determines the type of user, the characters initiates a dialog with it and the information to be interchange is store in the agent's memory. At the high level of abstraction, the bellboy has a given physical size, approximated by a bounding rectangle, with eyes (cameras), ears (stereo microphone), and arms. This set of sensors relay information about the state of the character and the surrounding environment to the AiE-Agent. The Chase component tell the AiE-Agent the position of the character in which direction the character is facing and what speed is going at. The mood component shows the current mood and translates them in their physical counterparts. Finally a battery indicator shows the current battery level of the bellboy, and a status indicator reports on the current interaction level.

4.2 Cognitive System

Its clear that if we want to see how Soar contributes to behavior then we need to explore the it in terms of some particular content [9]. Lets consider a simple scenario from the hotel Fig. 2. Sacarino stands around the hotel reception waiting for a guest arrival, and then initiates a dialog. First the guest came into the reception and makes the check in, and then the receptionist indicates to the bellboy, where to go (the guest's room). At that point the bellboy has to initiate a dialog with the user while they go to the room. The dialog is about hotel's information or services (laundry, room service, safe, staff, wake-up call etc). When they are near the room, the bellboy if its necessary indicates to the guest how to use the key card to open the door. Then the bellboy explains the hotel's mealtimes or other information. Finally the bellboy offers the user additional information, like, entertainment, touristic places, leisure and so on.

Just as the architecture is a theory about what is common to cognition, the content in any particular model is a theory about the knowledge the agent has that contributes to the behavior. For our AiE-Agent to act like a bellboy, we will have to give it many different kinds of knowledge. Before our model can dialog its first time, we must find some way to represent and process bellboy's knowledge in Soar. In Soar this structure provides a means for organizing knowledge as a sequence of decisions through a problem space. Some concrete examples of which are:

Fig. 2 Show the participants in our scenario in their usual locations and roles

- K1: Knowledge of the objects in the game. e.g. Lobby, front office, elevator, rooms, bar.
- K2: Knowledge of abstract events and particular episodes. e.g. What the user had mentioned in his previous interaction with the bellboy.
- K3: Knowledge of the rules of the game. e.g. Interaction with the bellboy.
- K4: Knowledge of objectives. e.g. Give information of the hotel's services, leisure, mealtimes and transport.
- K5: Knowledge of actions or methods for attaining objectives. e.g. Chase, Evade,Surround, tell jokes.
- K6: Knowledge of when to choose actions or methods. e.g. If the user don't need information, stop the dialog and go away.
- K7: Knowledge of the component physical actions. e.g. What to express (facial expressions) while dialogue

This list incorporates many different kinds of knowledge that our model must include: knowledge about things in the world (K1 and K2) and knowledge about abstract ideas (K3 and K4), knowledge about physical actions (K7) and knowledge about mental actions (K5), even knowledge about how to use the other kinds of knowledge (K6).

4.3 Behavior System

If we try to imagine (and draw) all the choices Sacarino might have to make during a game, given all the circumstances under which they might arise, we are quickly overwhelmed. Sacarino must make his decisions with respect to the situation at the moment. At the highest level of abstraction,as seen in Fig. 3, the AiE-Agent switches between different operating modes based on events. He starts in Exploring mode, and switches to Interaction mode once the Avatar position is known (and there is still enough battery). If at any point in time the mood status is negative or the interaction level is low, he switches to standby mode. Otherwise, Surrounding is the

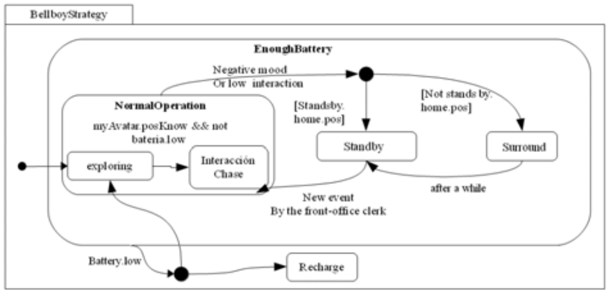

Fig. 3 The bellboy strategy

best strategy. In the event that the battery is low, if the location of the recharge station is known, the bellboy chooses to switch to recharge mode. Otherwise, it is best to continue Exploring, hoping to find a recharge station soon. When the charge is full, the bellboy switches back to whatever he was doing before he was interrupted. The mode changes are announced by sending corresponding events: when Exploring is entered, the explore event is sent, when Interaction is entered, interact event is sent, etc. The motor system is not explained in this work, this first phase is to model the correct behavior of a bellboy robot in this virtual environment.

5 Conclusions

Our decision of design a virtual agent is based on the modern conception of agent, while in the past software agents and robots have usually been seen as distinct artifacts of their respective domains, the modern conception is, in fact, to consider them as particular instances of the same notion of agent, an autonomous entity capable of reactive and pro-active behavior in the environment it inhabits. Our technique combines a form of cognitive modeling, based on the most development architecture in the literature, Soar, with emotions to influence decision making. In this paper we present a novel technique to produce intelligent behavior of an agent by interacting with the environment (simulated specific scenario) and the user. We model this interaction between the guest and a bellboy robot through a programming by-demonstration interface. Our technique produces virtual character behavior that can be quickly adopted when playing the game by non-expert users. Our method is a test bed of the ROBOTEL project a real-world application within hotel environment.

Acknowledgements. This research is based upon works partly supported by the Science and Innovation Spanish ministry (Pr. Nb.DPI2008-06738-C02-01/DPI)), jan-2009 to dec-2011 and CARTIF foundation.

References

1. LoBao, A., Evangelista, B., de Farias, J.A.L.: Beginning XNA 2.0 Game Programming: From Novice to Professional (2008)
2. Harel, D., Kugler, H.: The rhapsody semantics of statecharts (or, on the executable core of the UML), 325–354 (2004)
3. Vernon, D., Metta, G., Sandini, G.: A Survey of Artificial Cognitive Systems: Implications for the Autonomous Development of Mental Capabilities in Computational Agents. IEEE Transactions on evolutionary computation, special issue on autonomous mental development (2007)
4. Garzón, F.J.C.: Arquitecturas de la cognición, Mente = Cerebro + Medio (2008)
5. LehMan, J.F., Laird, J., Rosenbloom, P.: Agentel Introduction to Soar, an Architecture for Human Cognition: 2006 Update. University of Michigan (2006)
6. Kienzle, J., Denault, A., Vangheluwe, H.: Model-based Design of Computer-Controlled Game Character Behavior (2007)
7. Kopp, S., Gesellensetter, L., Krämer, N.C., Wachsmuth, I.: A conversational agent as museum guide – design and evaluation of a real-world application. In: Panayiotopoulos, T., Gratch, J., Aylett, R.S., Ballin, D., Olivier, P., Rist, T. (eds.) IVA 2005. LNCS (LNAI), vol. 3661, pp. 329–343. Springer, Heidelberg (2005)
8. Laird, J., van Lent, M.: Human-level AI's killler application: Interactive computer games. AI Magazine (2001)
9. Laird, J.: Extending the Soar Cognitive Architecture. In: Artificial General Intelligence Conference, Memphis, TN (2008)
10. Edward, L., Lourdeaux, D., Barthès, J.-P., Lenne, D., Burkhardt, J.-M.: Modelling Autonomous Virtual Agent Behaviours in a Virtual Environment for Risk. The International Journal of Virtual Reality (2008)
11. Newborn, M.: Deep blue's contribution to AI. Ann. Math. Artif. Intell. 28(1-4), 27–30 (2000)
12. Newell, A.: Unified Theories of Cognition. Harvard University Press, Cambridge (1990)
13. Langley, P., Laird, J.E., Rogers, S.: Cognitive architectures: Research issues and challenges (2006)
14. Ponsen, M., Spronck, P., Tuyls, K.: Towards Relational Hierarchical Reinforcement Learning in Computer Games. In: Proceedings of the 18th Benelux Conference on Artificial Intelligence, Belgium (2006)
15. Mohan, S., Laird, J.E.: Learning Play Mario. Technical Report CCA-TR-2009-03. Center for Cognitive Architecture. University of Michigan (2009)
16. Spronck, P., Ponsen, M., Sprinkhuizen-Kuyper, I., Postma, E.O.: Adaptive game AI with dynamic scripting and Machine Learning. Special Issue on Machine Learning in Games (2006)
17. UML Resource Page, `http://www.uml.org` (last access December 10, 2009)