Intelligent Virtual Environment Using a Methodology Oriented to Agents

Sandra Mateus[1](✉) and John Branch[2]

[1] Politécnico Colombiano Jaime Isaza Cadavid, Medellín, Colombia
spmateus@elpoli.edu.co
[2] Universidad Nacional de Colombia, Medellín, Colombia
jwbranch@unal.edu.co

Abstract. This paper describes a Virtual Environment with Intelligent Agents, using an agent-oriented methodology called Prometheus. This technique was incorporated in the perception and reasoning of a character in the Intelligent Virtual Environment (IVE), in order to react intelligently to given warning signs. The main objective of the character in the developed IVE is finding the optimum path without having to do extra shifts. The used technique achieves an optimal solution using only probability calculations, without requiring additional data to be supplied. This solution is obtained given the repetitions and the number of ants in the character agent. The virtual environment was created with the UDK tool to identify warning signs in a work environment. UDK is used because of its importance in the development of world-renowned games.

Keywords: Intelligent virtual environment · Intelligent agents · Ant colony algorithm

1 Introduction

Intelligent Virtual Environments (IVE) are a hybrid between Virtual Reality (VR) and Artificial Intelligence (AI). They can achieve great capabilities of complex and interactive behaviors to reach a high level of realism [1]. This realism is based on the elements allowing smart performance such as perception, reasoning, learning and communication using natural language.

One of the techniques in artificial intelligence is the so called technique of Intelligent Agents. An agent can be a reactive or deliberative system which has a degree of autonomy. This means that a task can be delegated to it the system itself determines the best way to perform this task. These systems are called agents, because they are active beings, useful producers of actions: they are sent to a given environment to achieve goals and actively pursue these goals, finding out by themselves the best way to achieve these objectives [2].

In the literature, there are intelligent tutoring systems oriented to virtual environments. For instance, Buche and Querrec [3] developed an intelligent tutoring system with some 3D virtual environments. They integrated a tutorial system adaptable with multi-agent systems for pedagogical decisions. Clement et al. [4] presented an intelligent

DOI: 10.1007/978-3-319-39907-2_68

tutoring system as part of an IVE for training. In this system, the proposed student model uses an ontology as a pedagogical and diagnostic approach for conflict resolution, in order to infer learning goals that the student has acquired eventually testing it in a 3D virtual biotechnology laboratory. Moreover, the use of agents has also been applied to intelligent vehicles. They have been used for different applications ranging from the physical simulation to full scale tests, using different virtualization levels for sensors, actors, scenarios and environments [5]. Simulations of evacuation are widely used to represent human emergencies. As an example, Xi and Smith [6] address this topic using a virtual environment with gaming technology and intelligent agents.

In IVEs centered on a character, Liu et al. [7] developed a framework for modeling virtual humans with a high level of autonomy at the behavioral and movement levels in a virtual environment. Their framework included a perception module, a decision module and control module of autonomous movement. Gilbert and Forney [8] developed a tour guided by an avatar in a virtual clothes store in a 3D world of Second Life Game using virtual agents and a robust variant of Artificial Intelligence Markup Language (AIML) having as challenge the Turing Test.

In this paper, we used an agent-oriented methodology called Prometheus, which consists of three phases to follow [9]: system specification, architecture design and detailed design. In the system specification, general objectives, functionalities and scenarios applied to the detection of risks in a work environment were defined. In the architecture design, agents (character agent and virtual environment agent), perceptions, actions, protocol and coupling data are defined. In the detailed design, the agents and their reasoning are described.

This paper is organized as follows: in Sect. 2, the concepts of perception and reasoning are discussed, and the 3D virtual environment created is displayed; in Sect. 3, the Intelligent Agents implemented in the virtual environment is explained; in Sect. 4, the experiments and results are presented; and Finally, the conclusions.

2 Perception and Reasoning

The Intelligent Virtual Environments must reach large capacities of complex and interactive behaviors to achieve a high level of realism [1]. This realism is based on elements enabling intelligent performance such as perception, learning, and communication through natural language and reasoning. According to the above, this paper focuses only on the perception and reasoning, cognitive and behavioral levels of a Virtual Environment.

According to Marthino *et al.* [10], perception is considered as all events of the virtual environment that are filtered according to the interests and location of the character and is based on two principles: (1) A limited perception, in which a character perceives all events, but only perceives its associated area; and (2) An inaccurate perception, in which the character perceives the virtual environment as it is, but only receives relevant events associated with it. They also describe the reasoning as a process performed by a set of production rules which are conditions. These conditions are based on the model of the world, in the state of the target, the characteristic behavior and the internal state information.

Given the above concepts, the reasoning of the character implemented in the IVE is based on the impact of its internal target and priority of the action to perform on the cognitive and behavioral levels.

Based on aforementioned, a model of IVE is proposed using a game engine. This model incorporates AI techniques such as agents. The goal is to produce, from a given perception of a character, a reasoning respect to the Virtual Environment.

In this model, this can be seen as a set classifier S, which takes a set of perceptions P1…Pn and combines them to make adequate reasoning R1…Rn. This reasoning to perform a certain action is supported on one of the techniques of Artificial Intelligence named in the model. Thus the system decides and selects an action A, according to the reasoning made (Eq. 1).

$$S(\{P_1, R_1\}, \{P_2, R_2\}, \ldots \{P_n, R_n\}) \rightarrow A \tag{1}$$

With this model, the following characteristics of an Intelligent Virtual Environment described by [1] want to be achieved: Decisive: any action taken by the character will be reflected in an effective plan; Real Time: The character should respond in real time to the perceptions of the environment and in the same way, adequate reason to perception form received; Ordered: That is, the agent follows the proper sequence in their behavior.

Perceptions were simulated in this virtual environment and their respective actions to perform through reasoning with the AI technique are shown in Table 1.

Table 1. Perceptions and actions that the character executes in the virtual environment

Perception	Action run
Detects fire	Activate the fire alarm
Detects electrical risk	Turn off light switches
Detects wet floor	Call the cleaning staff
Workplace - do not know what to do	Use the intercom for help

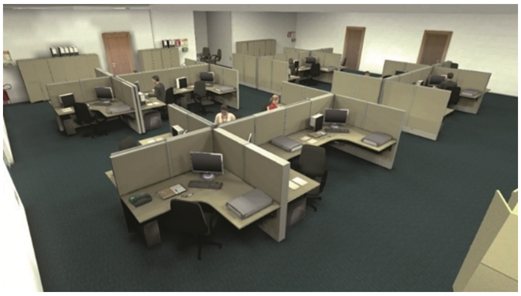

Fig. 1. Virtual environment developed with UDK

In Fig. 1 it is shown a rendering of the Virtual Environment. This environment is the one that the character will interact with as disclosed in Table 1 and to which AI techniques are applied as explained in the following sections.

3 Intelligent Virtual Environment with Intelligent Agents

In this work is used the agent-oriented methodology called Prometheus. This methodology consists of three stages [9]: System specification, architectural design and detailed design (Fig. 2).

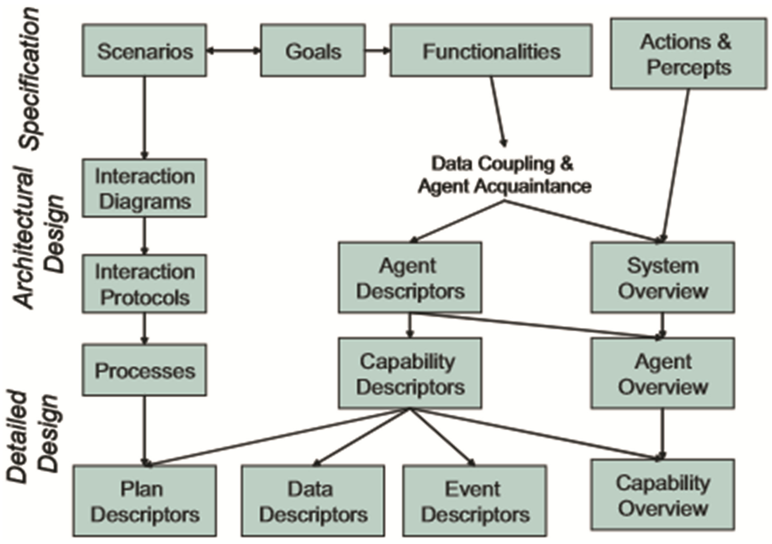

Fig. 2. Stages of prometheus methodology [9]

3.1 System Specification

Based on the proposed and developed virtual environment, Fig. 3 shows a general vision of the system specifications of the IVE developed on this work.

General Objectives. For the study case of providing warning signals on a work environment, the objectives are four: (Objective #1) Providing to people graphical information regarding what should be done upon perceiving a signal that might represent any kind of danger; (Objective #2) based on (Objective #1), maximizing the security on the work environment; (Objective #3) Surpass some kind of obstacle and (Objective #4) avoiding a possible accident.

Functionalities. Two functionalities have been defined in order to manage and operate the interaction during the execution of the virtual environment. They are focused on achieving the proposed objectives: (1) Identify possible obstacles which, for the study

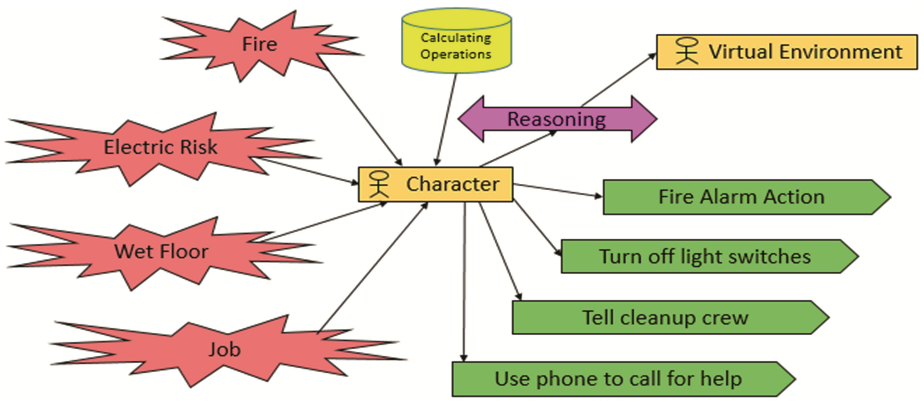

Fig. 3. Diagram of the virtual environment based on Prometheus

case, are the cubicles and walls and (2) Determine the feasible path. This is, to execute the adequate action, according to the received perception and to the randomness applied when executing the action.

Scenarios. The used scenarios are: “There is smoke in a corner”, which activates the perception of detecting fire; “there are electric sparks coming from a lamp” which activates the perception of identifying electric hazard; “there is water on the floor”, which triggers the detection of wet floor and “go to your work place” when there is not any action defined. There are also other scenarios such as “getting closer to the fire alarm”, “getting the environment dark and simulate a blackout”, “talking to the janitor” and “getting closer to the interphone” which corresponds to each action to be performed.

3.2 Architectural Design

Agents. There are two kinds of agents involved in the architectural design: Character agent and virtual environment agent. The character is the main actor, capable to respond, react and interact with the environment by using the ant colony algorithm. This algorithm is used to calculate the adequate path for the action to be performed in real time. On the other hand, the virtual environment agent is the agent which the character agent interacts with.

Perceptions. The perceptions of the character are the same ones described on Fig. 3: detect fire (P#1), detect electrical hazard (P#2), detect wet floor (P#3) and the work place (P#4).

Actions. The actions to perform are defined according to the perception: Start the fire alarm (A#1), turn off the light switches (A#2), call cleaning staff (A#3) and use the interphone to ask for help (A#4).

Protocol. It defines the interaction between character agent and virtual environment agent. Also, it defines the action to be performed or feasible path. It is compsed by the following messages:

- Location update: This is an informational message. It is sent from the character agent to the environment agent. Whenever the character agent changes its location, it must inform it to the environment.
- Environment change: This is a request-response message. Whenever the character agent performs a change on the environment (such as to activate an alarm), it informs to the environment agent which must respond whether this change is allowed or not. Also, it must update the environment information.

Data Coupling. It shows the process of data transfer in the system, in the environment agent as well as to transfer as to receive from the character agent.

Data is stored in a database management system which may be accessed by the two agents. The shared data are: character agent location and actions to be performed in the environment.

3.3 Detailed Design

Character Agent. This is a deliberative agent. It determines the action to be performed given the reasoning that it does from the received perception. The diagram of this agent can be observed in Fig. 4.

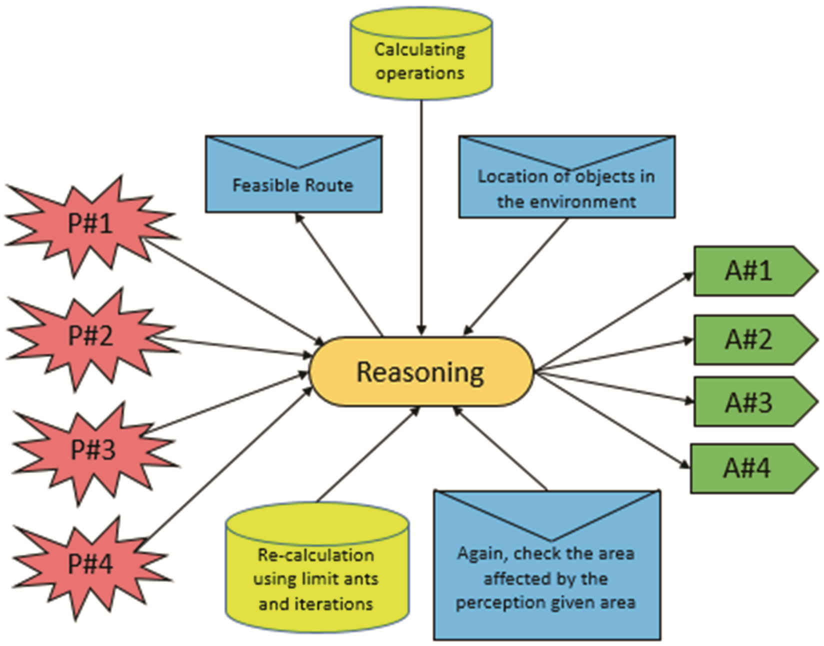
Calculating operations
Feasible Route
Location of objects in the environment
P#1
P#2
P#3
P#4
Reasoning
A#1
A#2
A#3
A#4
Re-calculation using limit ants and iterations
Again, check the area affected by the perception given area

Fig. 4. Character agent

When the agent begins to interact with the environment, it receives information about the location of objects in this environment. The calculation of operations and the re-calculation of the algorithm have the information of the managed data. The action to be performed is calculated by the ant colony algorithm.

The ant colony system has as principle to use the techniques of swarm and pheromone generation as the communication mechanism to create paths with destinations as the same swarm and places with food.

As metaphor, when the creation of a swarm begins, the ants go around in a random way in order to create paths spreading their pheromones. Thus, multiple travels are performed to shape an organized pattern. For this purpose the algorithm proposed by Brownlee [11] was used.

Virtual Environment Agent. This agent provides to the character agent the location of objects in the environment. Also, it is on charge to graphically show the actions performed by the character.

4 Experiments and Results

As explained in the previous section, the ant colony algorithm was implemented in the character agent. This algorithm is shown on Fig. 5.

```
Input: ProblemSize, $Population_{size}$, $m$, $\rho$, $\beta$, $\sigma$, $q0$
Output: $P_{best}$
$P_{best}$ ← CreateHeuristicSolution(ProblemSize);
$Pbest_{cost}$ ← Cost($S_h$);
$Pheromone_{init}$ ← $\frac{1.0}{\text{ProblemSize} \times Pbest_{cost}}$;
Pheromone ← InitializePheromone($Pheromone_{init}$);
while ¬StopCondition() do
    for $i = 1$ to $m$ do
        $S_i$ ← ConstructSolution(Pheromone, ProblemSize, $\beta$, $q0$);
        $Si_{cost}$ ← Cost($S_i$);
        if $Si_{cost} \leq Pbest_{cost}$ then
            $Pbest_{cost}$ ← $Si_{cost}$;
            $P_{best}$ ← $S_i$;
        end
        LocalUpdateAndDecayPheromone(Pheromone, $S_i$, $Si_{cost}$, $\sigma$);
    end
    GlobalUpdateAndDecayPheromone(Pheromone, $P_{best}$, $Pbest_{cost}$, $\rho$);
end
return $P_{best}$;
```

Fig. 5. The ant colony algotihm [11]

This algorithm is known by performing a large amount of recursive cycles to achieving an optimal solution. Because of this, it requires some limits in order to achieve that result. These limits are determined by the amount of ants and a given amount of cycles and repetitions to achieve the optimal calculation. What is desired is that the agent calculation be higher than the randomness in the least possible number of repetitions.

In this algorithm, two structures are used to store data. One structure stores the places that are going to be visited by the ants and the other one stores temporarily the data of the most adequate place, which are the actions to be performed. The maximum number of iterations and the maximum number of ants are considered.

The main function of this algorithm calls the search function (Fig. 6) whose goal is identifying which is the optimal path. That is to say, in each state of the character, the whole algorithm must be solved, and the character agent may only go onto the next state if it fulfills the condition of having an optimal value lower than the proposed one.

The search function has the objective of finding using probabilistic calculations from other functions the best path given some distances. To calculate those distances, the traveler salesman problem algorithm is used.

In the algorithm described in Fig. 6 the random permutation will make a shift of the places. Thus, it will always have a different objective and the calculations will always be different although the goal will be the same. The cost function delivers the calculation of the distance between the places. The pheromone matrix helps in the calculation of the path followed by the character.

Later, calculations are performed according to a random result indicating whether going through a greedy path or a probabilistic path. The differences is that the greedy

```
AgentsSearch ()
    best.vec = permutation_random (places)
    best.cost = cost (best.vec, places)
    init_pheromone = 1.0 / (places.size * best.cost)
    pheromone = init_pheromone_mat ( places.size , init_pheromone )
    For (s_i=0, s_i < max_it, 1) do
        For (s_j=0, s_j < num_ants, 1) do
            candidate.vec = stepbystep (places, pheromone, heuristic,greed)
            candidate.cost = cost (candidate.vec,places)
            If (candidate.cost < best.cost) then
                best= candidate
            EndIf
        EndFor
    EndFor
    Return best
EndAgentsSearch
```

Fig. 6. Search function in the ant colony algorithm

path uses a calculation previously given whereas the probabilistic path recalculates the given probability trying to obtain a more adequate result.

There is also a greedy function which compares some values previously generated to identify which one is the highest within a list and then delivering the next place to go to. The probabilistic selection recalculates, using the previous probability values, the next best place to visit.

Finally, in Fig. 7 the IVE can be seen, when it is executed with the smart agents. In this case, it is executing the reasoning upon detecting electric hazard due to a blackout in the environment.

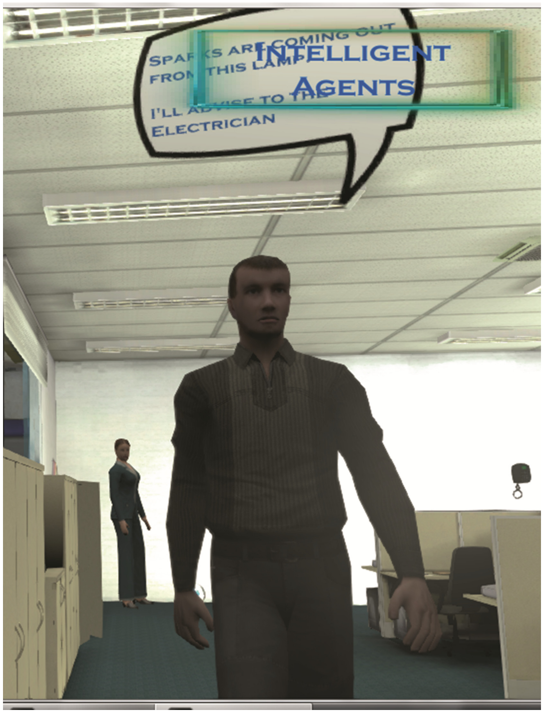

Fig. 7. IVE with agents

5 Conclusions

The objective of the main character in the Virtual Environment is to find the optimal path without requiring extra movements. The technique used in this paper solves this using only probabilistic calculations without requiring that real data be provided. This approach can achieve different optimal solutions, given the repetitions and the number of ants in the character agent. As much time this technique has to solve the problem, it will provide a more adequate result.

The technique has a system of calculations which, in spite of being simple, becomes complex when performing operations with a tool such as the videogame engines. By requiring so many interactions, giving the parameters of maximum number of ants and interactions, the established limits for each tool might be saturated.

In this case, UDK has a limit of iterations for each frame per second. Once it is surpassed, the tool just stops working. This makes that the implementation and potential of this technique becomes reduced by the limitations of the used software. It is proposed as future to evaluate the implementation of this IVE using a different algorithm in the agents.

References

1. Cavazza, M., Lugrin, J.-L., Hartley, S., Renard, M., Nandi, A., Jacobson, J., Crooks, S.: Intelligent virtual environments for virtual reality art. Comput. Graph. **29**(6), 852–861 (2005). Elsevier
2. Bordini, R., Hübner, J., Wooldridge, M.: Programming Mulit-agent Systems in AgentSpeak Using Jason. Wiley, London (2007)
3. Buche, C., Querrec, R.: An expert system manipulating knowledge to help human learners into virtual environment. Expert Syst. Appl. **38**(7), 8 (2011)
4. Clemente, J., Ramirez, J., De Antonio, A.: Applying a student modeling with non monotonic diagnosis to intelligent virtual environment for training/instruction. Expert Syst. Appl. **41**(2), 508–520 (2014). Elsevier
5. Kurt, A., Vernier, M., Biddlestone, S., Redmill, K., Özgüner, Ü.: Chapter 2 – testing of intelligent vehicles using virtual environments and staged scenarios. In: Advances in Intelligent Vehicles, pp. 45–64. Academic Press (2014)
6. Xi, M., Smith, S.: Simulating cooperative fire evacuation training in a virtual environment Using Gaming Technology. In: IEEE Virtual Reality 2014, pp. 139–140. IEEE, Minneapolis, Minnesota, USA: ©2014
7. Liu, W., Zhou, L., Xing, W., Liu, X., Yuan, B.: Creating autonomous, perceptive and intelligent virtual humans in a real-time virtual environment. Tsinghua Sci. Technol. **16**(3), 233–240 (2011)
8. Gilbert, R., Forney, A.: Can avatars pass the turing test? Intelligent agent perception in a 3D virtual environment. Int. J. Hum. Comput. Stud. **73**, 30–36 (2015)
9. Padgham, L., Winikoff, M.: Prometheus: A Practical Agent-Oriented Methodology. Ed. TermLing (2005)
10. Martinho, C., Paiva, A., Gomes, M.: Emotions for a motion: rapid development of believable panthematic agents in intelligent virtual environments. Appl. Artif. Intell. **14**, 14–33 (2000)
11. Brownlee, J.: Clever Algorithms - Nature Inspired Programming Recipes. Creative Commons, Australia (2012). ISBN 978-1-4467-8506-5