VIRTUAL-ME: A Library for Smart Autonomous Agents in Multiple Virtual Environments

VIRTUAL-ME: A Library for Smart Autonomous Agents in Multiple Virtual Environments

Roberta Castano, Giada Dotto, Rossella Suma, Andrea Martina, Andrea Bottino
Department of Control and Computer Engineering, Politecnico di Torino
Corso Duca degli Abruzzi, 24
10129 Torino, Italy

ABSTRACT

Emulating human behaviour is a very desirable characteristic for virtual agents. There is plenty of literature that focuses on a single specific aspect of human behaviour emulation, but it is quite rare to find a collection of implementations encompassing several aspects of the problem. In this work we present VIRTUAL-ME (VIRTUal Agent Library for Multiple Environment), a library that provides programmers with a complete set of classes that assembles various human characteristics and makes it possible to build smart agents. The assessment of the library capabilities to populate a generic virtual environment is also discussed through the analysis of different case studies.

Categories and Subject Descriptors

H.5.1 [**Multimedia Information Systems**]: Artificial, augmented, and virtual realities

General Terms

Algorithms, Management, Performance.

Keywords

Autonomous Virtual Human, Intelligent Virtual Agents, Interactive Virtual Environments

1. INTRODUCTION

Artificial intelligence (AI) has received increasing attention over the last thirty years. It has been used successfully in many fields such as finance, medicine, games, robotics and the web. In particular, in recent years, a quite complex area of research has emerged: the simulation of the autonomous behaviour of characters in computer graphics. More than any other field in AI, this one requires careful investigation into human behaviour and cognitive psychology in order to be able to model and reproduce accurate simulations. Over the last three decades, scientists have attempted to model all kinds of human behaviour, using simulation and visualization, primarily aimed at creating educational and training systems. Nevertheless, human behaviour simulation is also widely used in programs with entertainment, commercial and non-educational purposes. A large variety of applications adopt some kind of human behaviour emulation, from crowd control and evacuation planning to traffic density and safety [13].

The general problem of simulating (or creating) intelligence has been broken down into a number of specific sub-problems [11]. It is very common to find in the literature articles focusing on particular traits or capabilities desirable in an intelligent system. Investigations in the following fields appear to be the most popular: affective computing, the emulation of human needs mechanism, the environment perception, the autonomous navigation of a virtual environment, memory models, event management and the overall agent behaviour mechanisms.

Emotions are a fundamental trait of human personality. How we distinguish one emotion from another is a much debated issue among researchers in human science. The classification of emotions has mainly been considered from two fundamental viewpoints. The first treats emotions as discrete and substantially different constructs (an emotion is completely individual), while the second asserts that emotions can be characterized on a dimensional basis (an emotion is composed of interacting elementary components, [9]). Emotions can have different roles in driving agent behaviour, for example they can be used to select the next action or to control memory [12]. There is plenty of literature about these different roles of emotions in agent behaviour (partly surveyed in Rumbell et al. [9]). Researchers agree that the choice between basic or dimensional emotions - and which specific emotion within this category - should be based on the primary function of the agent and on the specific purposes of the application. For example social regulation requires a complex emotion mechanism, which can allow social interactions to be fully represented. On the other hand, alarm mechanisms require a basic emotional model, due to the necessity of an instantaneous and drastic effect. In other words, it is not possible to elaborate a model capable of approaching every situation and environment, but it is reasonable to develop a model that can cover as many roles as possible.

Needs-based AI is an alternative way to drive an artificial agent which is simple to understand and implement. The next action picked is based on the agent's internal state and on the environmental inputs. Need fulfilment have been used, for instance, by Terzopoulos to drive virtual pedestrians' behaviour [15] and one of the most famous implementations of this mechanism is represented by the game *The Sims* [19]. Another possible mechanism is to define the agent behaviours in terms of responses to events, as discussed in [14].

Other researchers have been focusing on the problem of enabling the AI component to perceive and explore its environment. For example, Tu presented in his work [17] a framework to simulate artificial fishes in which the perception relied on a visual sensor spanning a 300-degree angle around the fish head.

In addition to a vision system, a smart agent should be equipped with a navigation system, whose purpose is to provide a path without obstacles from one point to another in the environment. This task is usually broken in two subtasks: global navigation, which uses a pre-learned map of the space, and a local navigation which, on the other hand, provides the ability to avoid unexpected obstacles along the path and mainly relies on the vision system. The majority of researchers, faced the problem of navigation adopting a central collision avoidance system that controls the agents' movements and positions.

The agent's capability to influence the environment is defined by a set of possible actions that reflects on changes to the state of the environment itself. All actions that humans undertake in an environment are influenced by their emotional and physical state and by their personality. To create believable virtual characters, these factors must be taken into account. This makes the creation of agents emulating the rich complexity of humans a real challenge.

Among the possible behaviour management models, several researchers recommend the use of the *Behaviour Trees* (BTs). The Behaviour Tree is a "simple data structure that provides graphical representation and formalisation for complex actions" [6]. The first implementation of BTs appeared in 2004 in a one-act interactive drama called Façade, and since then, they have been increasingly used by game AI programmers to create more exciting and complex characters. Their effectiveness is witnessed by the fact that important games, like Spore (in 2007), Halo3 (in 2008), and NBA '09, adopted this approach.

The contribution of the software library VIRTUAL-ME is the capability to deal with different aspects related to the management of autonomous characters behaving like humans. Based on the analysis of the peculiarity of these aspects, this work proposes an organic, all-encompassing and real-time solution that can facilitate programmers and scientists in populating a virtual environment with a crowd of smart agents. This library enables the creation of different worlds populated with several independent agents, incorporating general cases that bring together most human behaviours and that can be easily extended to deal with other peculiar cases.

The rest of the paper is organized as follows. Section 2 describes the different elements concurring to define the behaviour of the autonomous agents and discusses their integration. Section 3 presents some experiments aimed at assessing the effectiveness of the library. Finally, Section 4 concludes the paper and outlines future works.

2. THE VIRTUAL-ME LIBRARY

The VIRTUAL-ME library was created after an in-depth analysis of the state of the art of various fields. In particular, the library proposes solutions to deal with the following key issues:

- agent behaviour mechanisms;
- affective computing;
- the emulation of human needs mechanism;
- the environment perception problem, especially the vision;
- the problem of navigating a virtual environment;
- memory models;
- event management.

All these elements concur to compose a reproduction as accurate as possible of human intelligence. In the library, the virtual human management is implemented by a Behaviour Tree (BT). This structure is employed to drive the AI character's behaviour while the accomplishment of the chosen actions is dictated by a combination of both needs and emotion emulation techniques. Furthermore, the characters have been equipped with a perception mechanism and the capability to explore the environment in order to fulfil their goals.

All these elements, and their integration, are detailed in the following sub-sections.

2.1 Affective Computing

An emotion is defined as a complex, subjective experience coupled with biological and behavioural changes. Emotions are capable of altering attention, or the likelihood of a certain behavioural response, activating associative memories, influencing the learning process and aiding social behaviour [5]. Defining emotional states appropriately can determine a more accurate representation of human behaviour.

While most models are tailored to a specific scenario, the solution implemented in our work offers a generic emotion model designed to be a good compromise between simplicity and granularity (in terms of emotion description).

The model conceived in this project was inspired by the work of Thayer [3] and Russell [10]. It is controlled by two dimensions: *Activity* and *Mood* (see Figure 1). Low values of the Activity parameter represent a more phlegmatic attitude, whereas high values identify a hyperactive disposition. For example, in some people, a negative event can induce a despairing reaction, with a sense of paralysis, while in other people it can rouse a furious reaction. The Mood parameter, on the other hand, identifies a negative valence with low values, while a positive value identifies a pleasant demeanour. As an example, in an emergency situation some people's reaction is negative, driving them to despair and other people might remain serene and in control.

To create a more varied and heterogeneous (*i.e.* a more believable) collection of agents, each of them was generated with a particular attitude or character disposition. In other words, every agent possesses a "basic" personal emotion that is chosen randomly. Then, as in everyday life, interactions with other people or with the environment can alter the current agent emotion, inducing a change in its behaviour. Once a change has occurred, the agent will return to his primal emotion after a period of time. This idea stems, again, from the observation of human nature. Even if some events can modify the actual emotions of individuals, in many cases, this change is just temporary and, as time passes, they return to their own proper attitude and temper.

2.2 Emulation of Human Needs

The implementation of needs is one of the most common techniques used in video games to drive autonomous agents. In this frame, an agent has a set of ever-changing needs that demand to be satisfied. To this end, the agent figures out what can be done on the basis of what is available in the surrounding environments. Inside the library, the application designer can define, for each type of agents, a specific set of needs, their domains and how they would evolve.

As an example, in our test cases, human needs reflecting the work of the humanist psychologist Abraham Maslow were implemented [7]. He elaborated a hierarchy, depicted as a pyramid, which suggests that people are motivated to first fulfil basic needs before moving on to other, more advanced, needs. Therefore, according to the characteristics of the specific simulation, we considered most of the basic needs (like hunger, thirst or physiological

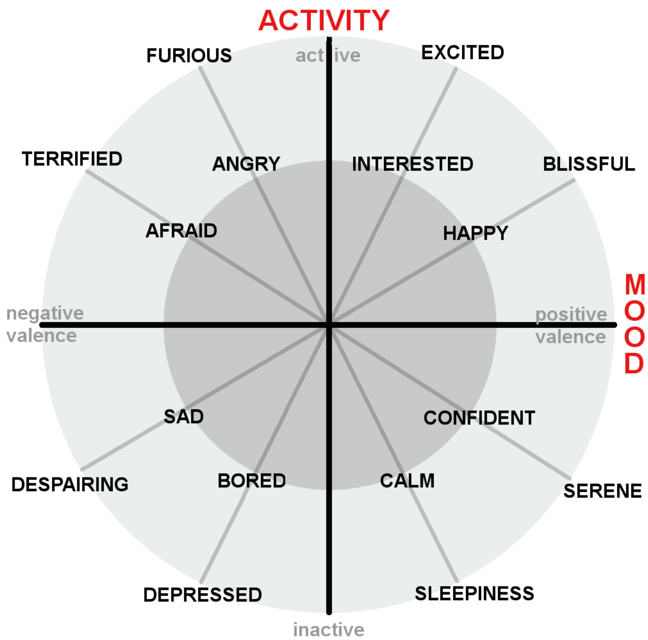

Figure 1: Emotion Representation

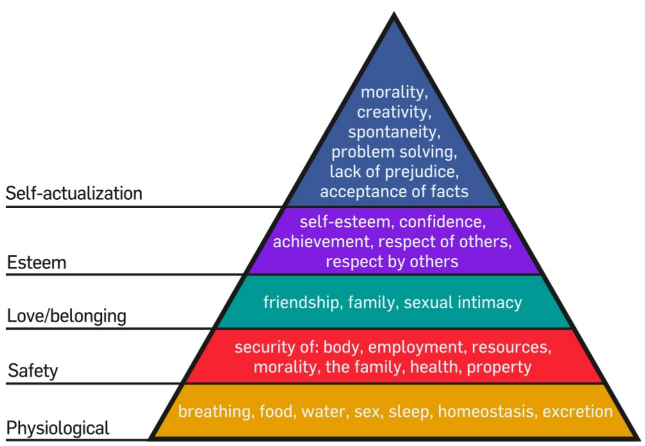

Figure 2: Maslow needs hierarchy pyramid.

needs) and some of the higher level needs (such as friendship and self-preservation) described in the Maslow's pyramid.

2.3 Navigation.

The VIRTUAL-ME Library implements a "decentralized" navigation system that allows the agents to be autonomous in their choices. A two levels navigation model was designed. The first level manages the global navigation problem relying on the A* algorithm [2]. A* requires to map the walkable area with a graph, which can be done only when obstacles are known *a priori*. Thus, the second level is responsible of handling dynamic obstacles, such as moving objects, agents or new hurdles created during the simulation. Agents have been equipped with a visual system that provides the ability to perceive the environment: an object is "seen" if at least one of its vertices falls into the agent's field of view and its vertices are not occluded by another object. Therefore, while A* determines the main path as a sequence of nodes, vision allows agents to walk from one node to another avoiding unexpected obstacles.

In order to preserve the agent autonomy, *i.e.* to avoid a centralized management, a force-based collision avoidance mechanism has been implemented. In brief, if a potential collision is detected by the vision system, a force is applied to the agent to change the direction and speed of its motion. The design of our local navigation system was inspired by many techniques like [8], [1] and [18] and improved in order to correct some of the difficulties affecting them, proving, for instance, to be able to properly avoid obstacles getting rid at the same time of reciprocal dances and virtual agent deadlocks. As an example, in Figure 3 we show the results of a test conducted to verify the ability of the agents to avoid deadlocks. In the experiment, 120 agents were arranged around a circle and instructed to reach their antipodal position. The congestion formed in the middle was quickly resolved as the agents reached their goals.

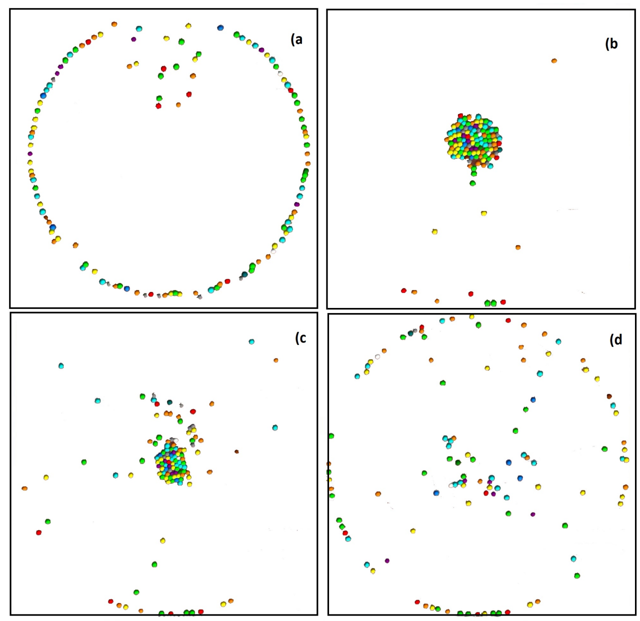

Figure 3: Example of deadlock avoidance: (a) 120 agents on a circle are instructed to reach their antipodal position; (b) a congestion forms in the middle of the circle but (c-d) it is quickly resolved.

The choice of a two levels navigation model was driven by efficiency purposes. In fact, while a navigation system merely based on vision is indeed able to drive the characters to their destinations, its performance drop with respect to the proposed solution is severe (see Figure 4).

2.4 Memory Model

The most important purpose of a memory model is to allow agents to remember past information. The lack of a storage mechanism could lead to mistakes or damaging repeated behaviours which would make agents appear less believable.

The memory model has been implemented in a very simple way. It consists of two structures: 1) a FIFO list keeping track of all seen objects and of all seen agents and 2) a FIFO list of the objects the agent interacted with and, for each object, a piece of information summarizing the changes in the character emotion, *i.e.* the emotional impact of the interaction itself. The processing of this

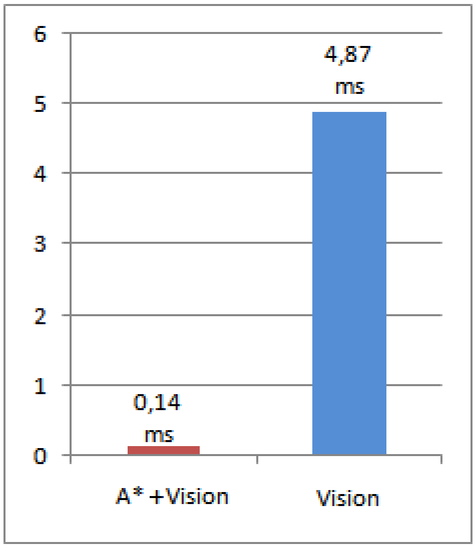

Figure 4: Comparison of the average per frame computational times of the two levels and of the vision-based only navigation systems in a reference environment (the Casino described in Section 3)

information can contribute to drive the character behaviour, since it allows agents to learn from past experiences.

In order to simulate the agent process of forgetting information, the lists have a constant length and items are removed after a certain time.

2.5 Events

An event is an occurrence that takes place in a virtual world. It is caused by environmental factors or agents' actions.

In VIRTUAL-ME, an *event generator* starts and ends the events and eventually makes them evolve in time. Each agent can be associated with an *event responder*, which contains all possible actions that the agent can do, according to his personality and to the event urgency (or priority), when a specific event is notified.

The event managing mechanism will be further discussed in the next paragraph due to its interconnection with the agent behaviour. A detailed example will be also discussed in section 3.3.

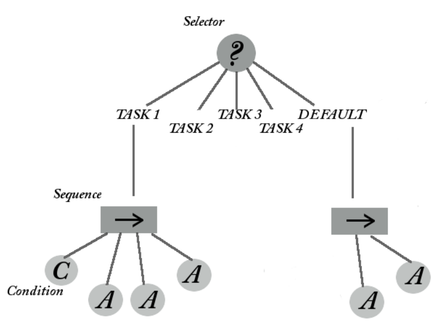

Figure 5: An example of Behaviour Tree

2.6 Interactions with the surrounding world

The agent behaviour is influenced as well by its interaction with the surrounding environment. To model these interactions, the environment and the objects it contains (including other agents) can expose different attributes. Those attributes are defined in an XML file associated to each object (or to each category of objects) and an agents can query them. The retrieved information can then be used to make decisions or to control the agent actions. For example if an agent is willing to gamble, it will look for an available gamble table by querying the ones in its surrounding to find one that is active and has free seats. Each table returns these pieces of information and the possible locations of the free seat. Then, when a table has been found, the agent walks towards the location of a free seat.

2.7 Agent Behaviour

As described in the previous sections, the demeanour of our virtual humans is determined by their emotions, needs and memory (*i.e.*, their internal state) and by the state of the environment (*i.e.*, their external state). Furthermore, agents have the ability to navigate through the environment, interact with it and with the objects it includes and to respond to events.

In order to manage all these elements, the agents' behaviour has been implemented using Behaviour Trees (BT). A BT defines and manages agents' *tasks*, which are related to the fulfilment of their needs and to their reaction to events.

In the BT, an agent task is represented by a sub-tree defining a sequence of child nodes, which are leaves of the BT. When a task is activated, the leaves are executed one by one (in order from left to right). The first child of the task sub-tree is always a *Condition* node that tests the agent's internal state (e.g. being hungry, willing to socialize) and the external state (e.g. having enough money to buy food, availability of other agents to whom he can talk) to check if the agent wants to or can realize the specific task. If the condition succeeds, the task execution is broken down into a set of sub-tasks managed by *action* nodes (*e.g.* search a seat in a restaurant, walk to the seat, seat, call the waiter, and so on). An active action node has one of the following three states: *Success* (the leaf has been completed correctly); *Running* (the action will continue during the next simulation step); *Failed* (the leaf has ended incorrectly). A task is successful only if all of its children succeed, otherwise, it reflects the state of the current active child (*i.e.* Failed or Running).

All agent tasks are children of the BT root, which acts as a selector, sequencing the different tasks on a priority order based on the child position (from left to right). The agent behaviour is managed as follows: the tree root calls tasks in order of priority. If the task condition is not verified or one of task's actions fails, the next task is started. If all tasks fail, the default task (which is the only one not starting with a condition) is executed. When a task is completed, the BT restarts task scheduling from the one at highest priority. An example of an agent's BT with five tasks is shown in Figure 5.

This control flow can be modified by events. After receiving an event notification, the event response is immediately served if it has a priority higher than that of the current task being executed which will consequently terminate. Otherwise, the event will be possibly served when the BT activates the corresponding task.

With this structure, the BT manages the agent's behaviour at two different levels: a higher level that includes needs, emotions and memory; and a lower level formed by the agent's motion ability and perception system. The high level plans the agent's future actions, while the low level manages the way the agent reach physically its goal (path finding and object avoidance).

3. RESULTS

VIRTUAL-ME Library has been implemented in C# and integrated into the Unity game engine (Unity Technologies, 2013). Simulations were carried out to test and explore the potentiality and effectiveness of the library. The two following environments have been used.

3.1 Environment 1: the Casino

The first world is a virtual reconstruction of The Sands Hotel in the 60's, one of the most prestigious and oldest resort casinos in Las Vegas, which was entirely reconstructed in 3D for a previous Virtual Heritage project. Two of its main spaces were used for the simulations: the *Gamble room* with the gambling tables and the bar lounge, and the *Copa Room* where shows were performed every night. The Casino is a complex scenario consisting of more than 3.7M vertices.

Different types of agents were populating the environment: customers, barmen, waiters, croupiers and artists. Customers have by far the most complex behaviour. Besides managing events and satisfying basic internal needs (eating, drinking, resting or going to the toilet) they can also enjoy their time at the casino by playing and gambling, attending shows and interacting and speaking with other customers or casino personnel. Barmen and croupiers join their working place when there are customers to be served and interact with them according to customers' requests. Waiters have a slightly more complex behaviour, since they have to manage table occupancies, customers orders and item delivery.

Two main events were scheduled in the simulation: a show in the Copa Room that agents can freely choose to attend, and an emergency situation, where all the agents have to reach their closer emergency exit from the casino.

Figure 6: An image of the casino simulation.

3.2 Environment 2: the Park

This environment depicts a park with a lake and other facilities, such as benches to rest, news-stands, a running path and some street food shops. A one way road and two pedestrian crossing were inserted, enabling agents to cross the road forcing the cars to stop. The agent types in this simulation are only two: drivers and pedestrians. The drivers have only a default task, which is driving across the park and stopping if any pedestrian crosses the road. Pedestrians behaviour include different tasks, such as eating, drinking, resting, running, watching the lake or buy a newspaper and read it. No events were registered for this environment. The number of vertices is 0.24M, far below those of the Casino.

Figure 7: An image of the park simulation.

3.3 Agent behaviour analysis

Although it is difficult to analyse all the details of the simulations, especially when the number of agents populating becomes very large, our observations allow us to state that in all the simulations the agents behaved as expected, acting according to their internal state and reacting properly to events.

Some examples observed in the Casino environment are the followings. When the thirst level is high, agents reach the bar, or call a waiter if seated at a table, to place an order. When an agent is in a good mood, he has money to buy fiches and he is willing to play, he reaches a gambling table. The amount of time he spend playing depends on the game evolution. Wins and losses can change the agent emotional state, affecting his will to stay longer. Effective interactions between agents have been also observed when an agent wants to start a conversation. First, he looks for a potential partner, and if the counterpart agrees to have a chat, the conversation begins. According to the dialogue evolution, which was randomly selected among a set of possible options, the current emotions of the two agents can change.

Agents' response to events was also working as expected. Attending the show was a medium priority task. Thus, agents executing tasks at higher priority (such as having dinner at the tables) did not respond to the event until they completed their actual tasks, while agents involved in tasks with the same priority of the show (e.g. gambling) were equally deciding to attend the show or continuing their activities. When the emergency event was triggered, given that this event has the highest priority among all tasks, all the agents rapidly left the casino through the nearest exit door.

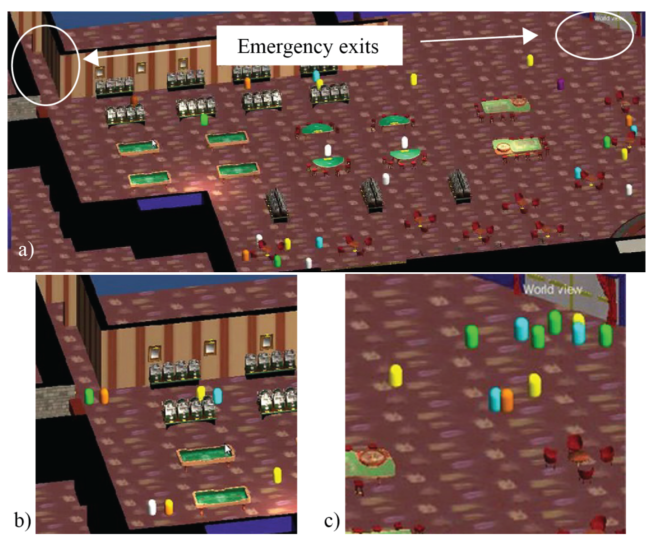

Figure 8: (a) the agents in the casino before the emergency alert, (b-c) the agents evacuating the building through their closest emergency exit.

3.4 Performance analysis

Two simulations were run in each environment using different representations of the virtual humans. In the first case, referred to as *Humans*, the avatars were represented with a complete mesh, having an average resolution of 3.000 vertices, and avatars were animated with Motion Capture data. In the second case, referred to as *Capsules*, avatars were represented with an inanimate capsule (400 vertices). This choices aimed at separating the complexity of AI management from that of the agent graphical representation.

All simulations were run on a 64-bit intel core i5-2410M 2.3 GHz architecture with 6 GB of RAM and an NVIDIA Ge-Force GT 540M graphics card. The simulation results are summarized in Table 1.

The first tests were aimed at understanding how many agents could be managed in different simulation scenarios. To this end, we run the simulation keeping on adding agents, until the frame rate dropped below 10 FPS. The Humans simulations could handle up to 130 agents for the park environment and 70 for the Casino, which, given the complexity of the environment and of the behaviour of its agents, can be considered as a reasonable result. The Capsules simulations reached up to 95 and 250 avatars for, respectively, the Casino and the park, showing that reducing the level of details of the avatar models results in a major increase of the library performances, especially in the park, where the agents have a less complex behaviour.

Further tests were aimed at collecting values of the resources used by the library. To this end, we run each simulation with 50 agents profiling the application. As it can be seen from the results, both Humans and Capsules simulations run at real-time, with a worst case of 14.86 ms (67.3 FPS) for the Humans Casino. The library functions use a percentage of the CPU time that spans a range between 36% and 48% in different cases. The estimated costs per agent shown in Table 1, obtained as the average CPU and library times per agent, provide an indication of the increase of the computational burden related to the population growth. The average memory allocated per agent is 1.38 MB and 0.94 MB in the Humans simulations, and 0.58 MB and 0.43 MB in the Capsules simulations, for, respectively, the Casino and the Park.

Table 1: Overview of the performance results: maximal agent number (first row); computational resources for 50 agents (second row); computational resources per agent (third row). CPU is the total execution time, while VMe is the amount of time allocated to the VIRTUAL-ME functions. The last row shows the average amount of memory allocated per agent.

		Casino		Park	
		Hum	Caps	Hum	Caps
Max N. of agents		70	95	130	250
CPU	(ms)	14.86	11.2	11.26	11.17
	(FPS)	67.3	89.3	88.8	89.5
Virtual-me	(ms)	6.19	5.4	4.2	4
	(%)	41.7	48.2	37.3	35.8
Time for Agent (ms)	CPU	0.29	0.21	0.18	0.17
	VMe	0.122	0.106	0.071	0.067
Memory/agent)	(MB)	1.38	0.58	0.94	0.42

3.4.1 Comparison with other approaches

A thorough comparison with the many approaches to agent's behaviour management is virtually impossible. First, standardized reference benchmarks in this area are still missing or have not been widely acknowledged ([4]). Second, while VIRTUAL-ME provide a compromise between believability, autonomy and performances to manage real-time simulations, several approaches pursue different objectives, such as managing large crowds, usually simplifying the AI and adopting centralized solutions, or providing more complex behaviour mechanisms, which however often hamper the management of large number of agents.

The two approaches most similar to our research, in terms of agent complexity and number, were the work of Terzopoulos [15] and the RAIN library [16]. As for the first work, the maximal number of autonomous agents that can be handled in real time is approximately 100. To obtain a measure of merit on the RAIN library, we modeled agents with a simple behaviour (similar to the one implemented for the Park) and kept on adding them to the simulation, founding as 120 their limit number to maintain an interactive frame rate.

Concluding, though more complex tests in more challenging environments are required, our results suggests that VIRTUAL-ME is a valuable solution for the implementation and management of a quite large population of different types of smart agents, each with its own peculiar behaviour.

4. CONCLUSIONS

The emulation of the human behaviour is a difficult task that requires to face many different issues, which in the literature have been often addressed individually. In this paper, we described a system capable of combining and integrating the solution of many of the problems related to the management of autonomous human-like agents, such as: the development of a navigation system capable of handling both static and dynamic objects, the characterization of agents by means of an original affective model enriched with an emulation of human needs mechanism, the ability to respond to external events and the introduction of a memory model which helps agents to adapt to, and to learn from, the environment they "live" in.

The result of our research is VIRTUAL-ME, a software library that takes into account many human capabilities and characteristics. The library is highly flexible and can be easily adapted to different environments and used to depict many different kinds of agents. Our experimental tests have shown the good potentialities of the library.

Despite that, some of the library features are actually implemented according to simplistic models. Future work could involve providing more complex implementations of these functionalities. As an example, the memory model can be expanded in order to allow agents to "store" more complex information to enhance the learning process. Furthermore, in order to improve the usability of the library and to simplify its management for non skilled users, it would be useful and desirable to implement a graphical interface to help defining the agent behaviours.

5. REFERENCES

[1] Paolo Fiorini and Zvi Shillert. Motion planning in dynamic environments using velocity obstacles. *International Journal of Robotics Research*, 17:760–772, 1998.

[2] Aron Granberg. A* pathfinding project, 2013. [http://arongranberg.com/astar/].

[3] Byeong Jun Han, Seungmin Rho, Sanghoon Jun, and Eenjun Hwang. Music emotion classification and context-based music recommendation. *Multimedia Tools Appl.*, 47(3):433–460, 2010.

[4] Alexander Kleiner, Alessandro Farinelli, Sarvapali Ramchurn, Bing Shi, Fabio Maffioletti, and Riccardo Reffato. Rmasbench: benchmarking dynamic multi-agent coordination in urban search and rescue (extended abstract). In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, AAMAS '13, pages 1195–1196, Richland, SC, 2013. International Foundation for Autonomous Agents and Multiagent Systems.

[5] R. Levenson. *Human emotion: a functional view*. Oxford Univarsity Press, 1994.

[6] Daniel Markowitz, Joseph Kider, Alexander Shoulson, and Norman Badler. Intelligent camera control using behavior trees. *Motion in Games*, pages 156–167, 2011.

[7] A. H. Maslow. A theory of human motivation. *Psychological Review*, 50:370–396, 1943.

[8] Craig W. Reynolds. Flocks, herds, and schools: A distributed behavioral model. pages 25–34, 1987.

[9] Timothy Rumbell, John Barnden, Susan Denham, and Thomas Wennekers. Emotions in autonomous agents: comparative analysis of mechanisms and functions. *Autonomous Agents and Multi-Agent Systems*, 25(1):1–45, 2012.

[10] James A. Russell. A circumplex model of affect. *Journal of Personality and Social Psychology*, 39(6):1161–1178, December 1980.

[11] Stuart J. Russell and Peter Norvig. *Artificial Intelligence - A Modern Approach (3. internat. ed.)*. Pearson Education, 2010.

[12] M. Sceutz. Useful roles of emotion in artificial agents: A case study from artificial life. In MIT Press, editor, *Proceedings of the national conference on artificial intelligence*, pages 42–48, 2004.

[13] Sharad Sharma, S. Otunba, and Jingxin Han. Crowd simulation in emergency aircraft evacuation using virtual reality. *2012 17th International Conference on Computer Games (CGAMES)*, pages 12–17, 2011.

[14] Catherine Stocker, Libo Sun, Pengfei Huang, Wenhu Qin, Jan Allbeck, and Norman Badler. Smart events and primed agents. In *Intelligent Virtual Agents*, pages 15–27. Springer, 2010.

[15] Demetri Terzopoulos. A reality emulator featuring autonomous virtual pedestrians and its application to distributed visual surveillance. In *Proc. IEEE Virtual Reality*, pages 1–4, 2008.

[16] Rivel Theory. Rain.

[17] Xiaoyuan Tu and Demetri Terzopoulos. Artificial fishes: Physics, locomotion, perception, behavior. *j-COMP-GRAPHICS*, 28(Annual Conference Series):43–50, July 1994.

[18] Jur van den Berg, Ming C. Lin, and Dinesh Manocha. Reciprocal velocity obstacles for real-time multi-agent navigation. pages 1928–1935, 2008.

[19] Will Wright. The sims code, 1997. [http://www.donhopkins.com/home/images/Sims/.