

Multi-agent Systems and Sandbox Games

Sergio Ocio¹ and Jose Antonio Lopez Brugos²

Abstract.

In recent years, several games have presented non-linear game-play systems; they are also known as sandbox games. Players are offered big, open, full of life worlds where they have a high degree of freedom to choose what they want to do to progress through the game. Multi-agent systems can help providing game designers with the means to achieve their creative visions and build more complex environments.

1 INTRODUCTION

First videogames were shown in 1970's. Since then, the growth of the industry has been exponential. For instance, videogame and console sales reached 18.85 billion dollars in 2007, far outpacing growth of film and music industries [1]. Sales also outstripped those of DVDs in 2008 [2]. Complexity has increased in a similar manner. From games like *SpaceWar!* or *Pong*, to the massive universes presented in recent games like *Fallout 3* or *Spore*, computers' evolution has provided creative teams with new tools to achieve their visions.

1.1 Motivation

Artificial Intelligence techniques used in a game are one of key factors that contribute to make it actually fun to play, while conveying a sense of an alternative, yet plausible, world where players want to spend their time in, which is the ultimate objective of every development team.

Building an AI system that is suitable for a videogame is a very complex and dependant on the nature of the game process [6]: for example, the AI of an strategy game is completely different to that in an FPS (First Person Shooter).

There is a growing divergence between what usually is the subject of study in educational scopes (such as machine learning or evolution), and what it seems the videogame industry is interested on (as film-like techniques, where a world, a story, is recreated, trying to make the player take an active role on it). A great example of this type of game is *Bioshock*, where players are presented with an alternate world, set up under the sea, where the attempt to achieve a perfect society, formed by the elite of humanity, destroyed itself. From the very first moment, when the main character suffers a plane crash, leading him to discover Rapture (the underwater city), the game is trying to convey a sense of grandeur and immersion: every step leads to discover the whole plot, as in a big Hollywood production (rather than a videogame).

Building such systems is a complex process, as they depend on hundreds of variables, and are also in a continuous relationship with

players (who judge the effectiveness and quality of them every second). Ultimately, they pursue two defined objectives: achieve a solid AI and provide a fun challenge. Many times the most important thing is not to get an invincible enemy, but to balance "intelligence" and player's satisfaction adequately. It is, then, clear that the kind of AI techniques used in games are completely different to those used in other fields of science.

Here comes the dichotomy: do we really want to build an unpredictable AI, which can surprise players with unexpected actions, or are we looking for a system that just follows a predetermined script (although players do not lose the feeling of facing a real intelligence)? Some research is being conducted regarding this matter, so a Visibly Intelligent Behaviour [3] (i.e. one that not only solves problems effectively, but appears to be intelligent: it is perceived as a human-like intelligence) is achieved. So, the final goal is to try to avoid cases where evolution of agents [4][5] produces non-realistic results, even though they could be getting correct results (in terms of problem solving).

1.2 Sandbox games

Sandbox games have become one of the most successful type of games in recent years. They cannot be classified into any of the traditional genres, like, for instance, strategy, adventure, shooter, sports or driving games, and most of the times they are a conglomerate of different gameplay experiences. They can also be described as systems that allow players to do whatever they want, without having to follow a strict linear story.

Grand Theft Auto is the best example of what a sandbox driving game can offer. It presents a city where players can do almost everything they want: drive cars, cycles, ride trains, fly helicopters... The game is divided in different missions that do not necessarily follow a strict order, and represent a way to progress through the main plot. So, one of the keys to the success of the franchise is the freedom it offers, which has its foundations on the recreation of a whole city (or region), which is full of traffic, presents day/night cycles, is populated by many agents (police, pedestrians), has its stores...

However, this *life* is just an illusion: traffic is limited, police acts mostly triggered by scripts or a small subset of player behaviours, pedestrians only appear in a short number, etc. But not all of this is negative, as the game offers what it has been designed for. Nonetheless, it seems that sometimes that is not enough.

As games become more and more complex, AI systems beneath them must be updated to deal with a wider range of situations and offer human-like responses to the problems they are trying to solve. The creation of a game is an expensive process that can take several years of hard work. Thus, this paper presents a high-level view of what a multi-agent based city should be, decomposing it in its basic elements, which could be further studied in the future.

¹ University of Oviedo, Spain, email: sergiocio@gmail.com

² University of Oviedo, Spain, email: brugos@uniovi.es

2 INTELLIGENT AGENTS

Software agents have their origin in Software Engineering, Artificial Intelligence and Human-Computer Interaction fields. They have some roots in “actors”, a technology presented in late 70’s. They are, basically, objects with an internal state and the ability to communicate and interact with their environment. We can quote several definitions offered by AI researchers:

- An intelligent agent is a computer system that replaces a person or process to carry out an activity or complete a requirement. This entity is capable of making decisions, which are similar to those described by human intentions. An intelligent agent can operate within the limits of a general necessity or precisely represented within the limits of a certain area of information [7].
- From a general point of view, and considering its behaviour, an agent is something that perceives its environment through sensors and acts on it through effectors. Regarding rationality, an agent can maximise its efficiency based on the evidence offered by a sequence of integrated knowledge and perceptions it might possess. An intelligent agent is autonomous because its actions and preferences depend on its experience, rather than on external knowledge built in the environment created by its designer [8].
- An agent is a piece of software that is able to perform flexible and autonomous actions in certain environments. In that case, it is considered that flexibility has reactivity, proactivity and sociability, where [9]:
 - A reactive system maintains a continuous relationship with its environment and responds to its changes.
 - A proactive system focuses its behaviour on achieving certain goals. Therefore, it is not only controlled by events, but it can also take the initiative.
 - Sociability refers to the fact that as a multiagent environment, there are certain goals that cannot be achieved without the existence of inter-agent cooperation.

3 THE CITY AS A MULTI-AGENT SYSTEM

In order to convert a sandbox city into a multi-agent system, it is important to know what this type of systems are. According to Demazeau [10], they can be defined as an organised group of agents that interact in a common environment. So they are made up of four key elements:

- Organisation
- Agents
- Environment
- Interactions

3.1 Organisation

As the resulting system will be huge, an action hierarchy [11] will be used to try to organise the way things work.

Decisions are dispersed along the chain, going from long-term, high-level decisions, to immediate low-level ones. So, higher levels control the ones below them, while these inform their superiors about the state of the world. This will allow both logic organisation and performance to be improved. The proposed levels are:

- The city itself, which represents the highest level and can work as an agent perceiving whatever is happening inside it as an input.

- A control layer will be the second level, as a fundamental means of communication between the city and players.
- A third level composed by actual game’s NPCs (non-player characters), as traffic, pedestrians, cops.
- The lowest level everything that can be seen as dispensable is found. However, these elements, shall they appear in the game, would increase the sensation of reality. Examples of this type of elements could be animals (cats, rats, birds...), public transportations (metro system)...

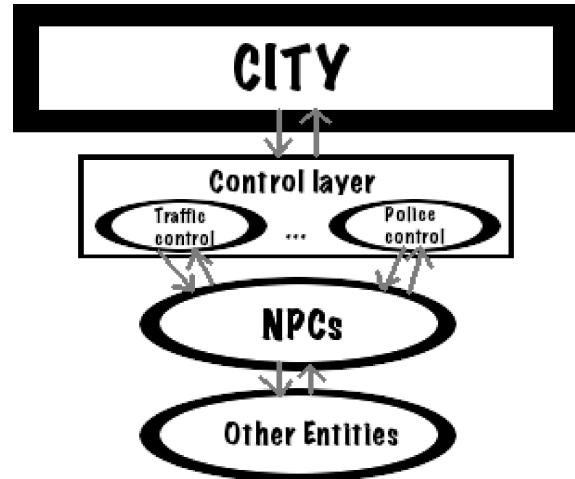


Figure 1. Structure of the city. Lower levels transmit their state to higher ones, while these emit orders.

3.2 Agents and environment

3.2.1 The City

The city is the brain of the system. The biggest difference with traditional multi-agents systems is that it is the environment and, at the same time, an agent, and as such it must be able to respond to different stimuli; these inputs would be bonded to different necessity conditions (city’s health). Its reactions must, also, be adequate to players’ game style.

This way the city would be ruled by two fundamental variables: equilibrium maintenance and adaptation to players.

Maintenance The city must maintain an optimal pace of operation at all times and avoid undesirable situations. For example, it should avoid severe traffic problems and maintain law enforcement in the city. This is achieved by creating the action plan for the level immediately below it in the hierarchy of the AI, i.e. the police.

While drawing up plans, the city must know how to cope with different situations, such as keeping an optimum number of policemen in the world to control any problem adequately, or modify its traffic lights patterns to solve traffic jams.

Adaptation to players Equilibrium maintenance in the city could be made even perfectly, avoiding any attempt to break it by players, but this could lead to a lack of interest for the game. Therefore, it is necessary to find a threshold for keeping the welfare state at acceptable levels, without disturbing players.

Moreover, not every player faces a videogame the same way, so it is interesting that the city takes into consideration players' styles to try and develop a strategy that maximises fun for players. For example, if a player breaks the virtual law he will be chased by AI-controlled policemen, and the system must decide whether to use an aggressive strategy, with greater numbers of units and/or more effective ones, or use an easier approach for the player, in case this has been classified as being reluctant to face these encounters.

3.2.2 Control layer

In order to be able to receive information about its state, and to maintain optimal stability values, the city must be able to communicate with the NPCs layer adequately. As several types of NPCs can be present, this layer would contain a set of different specialised controllers, like a traffic controller, a pedestrian controller, police controller...

Focusing on the former, police is managed at a high level by its controller, which prepares a general plan of action, such as "chase the suspect" for each individual; however, at a low level, police itself must be able to cope with problems using their own reasoning.

Each unit would be a police officer who can communicate with the whole city's police department: they know about general state of the city, because it has been passed to them, and perceive further information through their senses. They must also be able to travel between two locations in the city in an optimal way, as well as recognise what problem they are facing in every situation.

3.2.3 NPCs

They are undoubtedly the soul of the city, and those that make it feel alive. Moreover, they represent the AI which players would have most of the contact with. Each vehicle is controlled by an agent, which is, in fact, a pedestrian driving its car. Drivers must know their destination, which path to follow, comply with traffic rules (as traffic lights and pedestrian crossings), avoid accidents, try to optimize their trip (overtaking, changing lanes, searching for alternative ways...). In addition, each agent shall have its own personality, which would affect his driving style (aggressive or conservative, how much risk to take...), how they would behave against other agents...

Agent personality A system shows artificial personalities when it is possible to recognise that its characters are expressing personality archetypes. According to Ellinger [12]

"A personality archetype is a set of clear, recognisable and consistent behaviours users can describe with a single word"

In order to achieve this, it is necessary, first of all, to define which archetypes shall be implemented. Each of them must be easily recognisable and different from each other, so it is not advisable for a videogame to present a extremely high number of them. Random behaviours must be minimised, as it is difficult to tell different archetypes apart when they are present. On top of this, an agent (character) must not change its archetype, unless the game's context calls for such mutation explicitly.

Strategy games usually feature archetypes easily differentiable, where each type of unit can present its archetype. For instance, a harvester, which just mines for resources and can be considered just a worker with no military force, or a soldier specialised in long-range attacks, that would try to avoid close combat with the enemy, which

would make him more vulnerable. In *Black & White*, the archetypes shown try to imitate the behaviour of their owners (players), introducing basically two types of creatures: Good and Evil; although, in this case, the game will try to blend both archetypes to create a third one, depending on how inclined towards one of the sides the Creature is.

In a driving sandbox game, personalities will be most noticeable on agent driving styles. Each agent would have its set of personality traits, which would decide the way it drives. An example of personality characteristics could be:

- How likely the agent will drive on pavements.
- How likely the agent will ram other cars.
- How likely the agent will decide to make u-turns.
- How actively the agent will look for shortcuts.
- Whether the agent will prefer to drive at a fast or a slow speed.

Combining these values, different personality archetypes could be produced. This would allow designers to create a richer experience easily, whilst the code remains unchanged.

3.2.4 Other entities

This latest set of entities includes everything that makes the environment feel alive, but which absence would not affect the overall impression of the world.

For example, a dog could be implemented: if it is frightened by some noise, it would run in the opposite direction, without noticing the presence of a road, where an agent was driving his vehicle towards his destination; this agent shall add a new goal to its list, "obstacle avoidance", trying not to run over the animal, although this could end up causing an accident against another car.

This is, more possible situations could be added, improving gaming experience and the ability to surprise the player, but they would not add anything strictly required for the game to work.

3.3 Interactions

Once the agents, environment, and organisation are defined, a suitable communication system must be implemented. There are two different types of communication to be taken into consideration.

3.3.1 Agent-environment communication

Agents must be able to communicate with their environment, as this is one of the characteristics that define a multi-agent system. There are many options to do so:

- A first method is called Smart-Objects [13]; in this case, objects themselves pass their information to agents. This is similar to what is done in *The Sims*. This approach offers a clear separation between agent and object, as well as a knowledge decentralisation.
- Another method consist in taking interaction further, and develop a communication system that imitates nature [14]: agents will be able to interact with objects taking into account past experiences, that allow them to have some degree of knowledge about objects they had never seen before. SOTAI (Smart ObjecT-Agent Interaction) represents this method. Every object will be defined as a set of actions, each of them associate to a series of tokens. A token is a symbol that represents a stimulus. Agents will have several streams, or stimuli receptors. From this information, the system will build contexts, or cause-effect relations between these stimuli

and consequences. Contexts will group in strings, which represent knowledge nets and that make it possible to know the characteristics of an object from a set of input stimuli. Agents can use these nets to face new types of objects, that where unknown, more efficiently.

3.3.2 Communication among agents

Achieving a great communication between agents is really important, so they can work as sensors for the city. They can convey feelings, emotions, ask for help... and the city would be listening to this state (that is, in short, its own) and trying to act accordingly.

Communication among agents can be implemented as a messenger system [15], which can also be used for any other game system. A message is composed by an information type, emitter and recipient identifiers, as well as some additional information fields. A delay field can be added, so messages are not processed immediately after reception. A message would be similar to this:

```
// Message types
enum EMessageType
{
MT_MOVE,
MT_ATTACKED,
...
};

// Message
struct TMessage
{
EMessageType type;
Identifier senderID;
Identifier receiverID;
float delay;
...
};
```

Each game entity would implement a HandleMessage method, that would process received messages.

```
// Recipient
struct IMessageRecipient
{
virtual void HandleMessage(
const TMessage& msg ) = 0;
};

// Entity
class CEntity : public IMessageRecipient
{
...
public:
virtual void HandleMessage(
const TMessage& msg );
};
```

Using this kind of agent-to-agent communication, cooperation among agents can be achieved.

4 LOD AGENTS

Transforming the city into a multi-agent system, as described in the previous section, would require a huge agent count to be active all the time, which would produce a big overhead on performance.

AI, as any other subsystem of a game, would need to fit in the resource slot it has been assigned, so performance must become as priority as obtaining human-like behaviours.

LOD (Level of detail) is a concept mainly used in computer generated graphics. In a 3D environment (with perspective) objects will become smaller as the are taken further back on the scene. Thus, at a certain distance, using a worse quality model or texture will not affect the final result, and will save resources (as memory).

Extending this idea to AI, multi-agent systems, active agent count can be reduced, so they become less resource-intensive. Agents must be classified depending on their relevance (most of the times, their distance to players/cameras). Each of these groups represent a level of detail, would be ruled by its own logic and would, probably, be updated at a different rate, so those entities in lower levels are stepped less frequently [16] [17] [18].

4.1 Architecture

Once the different levels have been decided, a suitable architecture shall be chosen. This paper proposes a hierarchical model using agent controllers or *overlords* and the *mastermind*.

In this model, the highest LOD will contain fully functional agents. They will behave as regular agents and will be updated at the maximum available rate. Once the system decides an agent has become less important, it will be downgraded.

Agents in the second level of detail would lose their identity as individuals, and would be controlled by an *overlord*. This overlord is, in fact, a new type of agent, a controller one, which will act as a proxy between the environment and the agents it is managing, deciding what information is relevant for each of its minions and what actions shall affect the environment.

A *mastermind agent* forms the lowest LOD. It will act as a ruler for the *overlords* and decide when and how to update them.

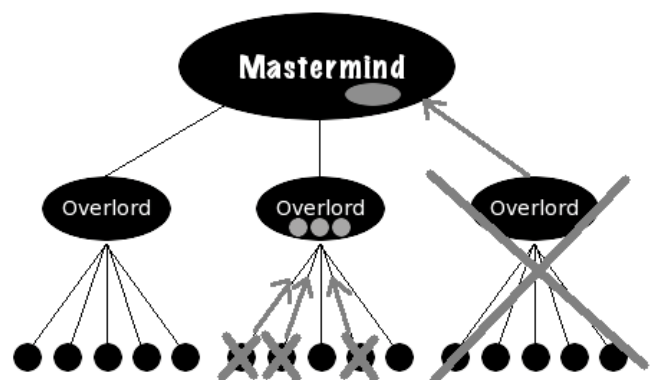


Figure 2. Overlord/mastermind architecture. Individual agents would become part of overlords as they are downgraded. The same would happen to overlords, which eventually would become controlled by the mastermind. This way, the number of active agents would be reduced.

So, contrary to what happens in graphics LODs, an agent would not be independent during all the states of its life, but would become part of a group at any time its behaviour is not directly noticeable by players. This way, the agent count would be dramatically decreased.

5 CONCLUSION

Videogames are a great challenge for Artificial Intelligence systems and represent a whole research field in themselves, trying to achieve a type of intelligence that is not looking for perfection, but to improve the game experience.

Videogames tend to present rich, full of life worlds, getting close to things that, so far, were only within the reach of films, although they are still at an early stage of development, and not always succeed in showing complex realistic environments.

Situations handled by an AIs are highly dependent on the type of game and its design requirements, and it is possible that a solution used in one videogame cannot be applicable in any other, which would lead to the development of specific techniques. Therefore, this is a field that offers a great potential in terms of researching and development of new algorithms.

The proposal of this work is to start a new research aimed at achieving more realistic sandbox games, building a city full of life, using, improving or creating specific algorithms or techniques built upon existing work relating multiagent systems.

Specifically, the research lines that could be followed are:

- Mastermind/overlord LOD architecture.
- Traffic control system, with intelligent agent-driven vehicles.
- Command hierarchy to manage conflicts.
- Conflict-solving planning.
- Personality and agent artificial emotions.
- Adapt AI behaviours to players style.
- Study a new methodology to implement agent-based AI systems.

REFERENCES

- [1] Growth of gaming in 2007 far outpaces movies, music. <http://arstechnica.com/news.ars/post/20080124-growth-of-gaming-in-2007-far-outpaces-movies-music.html>
- [2] Video game sales outstripped sales of DVDs in 2008, say analysts. <http://www.telegraph.co.uk/scienceandtechnology/technology/technologyreviews/videogamereviewsandpreviews/4357957/Video-game-sales-outstripped-sales-of-DVDs-in-2008-say-analysts.html>
- [3] Bryant, B. D. Evolving Visibly Intelligent Behavior for Embedded Game Agents. University of Texas at Austin. (2006).
- [4] Miikkulainen, R., Bryant, B. D., Cornelius, R., Karpov, I. V., Stanley, K. O., and Yong, C. H., Computational Intelligence in Games. In Yen, G. Y. and Fogel, D. B. (editors), Computational Intelligence: Principles and Practice, pp. 155-191. IEEE Computational Intelligence Society. (2006).
- [5] Stanley, K. O., Bryant, B. D., Miikkulainen, R., Evolving Neural Network Agents in the NERO Video Game. Proceedings of the IEEE 2005 Symposium on Computational Intelligence and Games (CIG05). (2005).
- [6] Nareyek, A. AI in computer games. Queue archive. Volume 1, Issue 10 (February 2004), pp. 58-65. (2004).
- [7] King, J.A. Intelligent Agents: Bringing Good Things To Life. AI Expert, pp. 17-19. (1995).
- [8] Russell, S. y Norvig, P. Inteligencia Artificial: un Enfoque Moderno. Prentice-Hall. (1997).
- [9] Wooldridge, M. Intelligent Agents: Introduction. 2nd European Agent Systems Summer School, EASSS2000. August 14th-18th. Saarbrücken, Germany. (2000).
- [10] Demazeau, Y. Foundations of Multi Agent Systems. 2nd European Agent Systems Summer School, EASSS2000, August 14th-18th, Saarbrücken, Germany. (2000).
- [11] Reynolds, J., Tactical Team AI Using a Command Hierarchy. AI Programming Wisdom. Charles River Media, Inc. (2002).
- [12] Ellinger, B., Artificial Personality: A Personal Approach to AI. AI Programming Wisdom 4. Course Technology. (2008).
- [13] Abaci, T., Ciger, J., Thalmann, D., Action Semantics in Smart Objects. Proceedings of the Workshop towards Semantic Virtual Environments, pp. 121-126. (2005).
- [14] Sequeira, P., Vala, M., Paiva, A., "What can I do with this?" Finding Possible Interactions Between Characters and Objects. Proceedings of the Sixth Intl. Joint Conf. on Autonomous Agents and Multiagent Systems. The International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS). (2007).
- [15] Rabin, S., Enhancing a State Machine Language through Messaging. AI Programming Wisdom. Charles River Media, Inc. (2002).
- [16] Kyo-Hyeon, P., Dong-Moon, K., Tae-Bok, Y., and Jee-Hyong, L. Real-Time Crowd Control using Fuzzy Steering Behavior. ISIS 2007. Proceedings of the 8th Symposium on Advanced Intelligent Systems. (2007).
- [17] Brom, C., Lukavsk, J., er, O., Poch, T., and afrata, P. Affordances and level-of-detail AI for virtual humans. Proceedings of Game Set and Match 2, Delft. (2006).
- [18] MacNamee, B., Dobbyn, S., Cunningham, P., and OSullivan, C.. Men Behaving Appropriately: Integrating the Role Passing Technique into the ALOHA System. Proceedings of the AISB02 symposium: Animating Expressive Characters for Social Interactions (short paper), pp. 59-62. (2002).