2009-01-01

Agent Based Modeling in Computer Graphics and Games

Brian MacNamee
Dublin Institute of Technology, brian.macnamee@dit.ie

Recommended Citation

Mac Namee, B. (2009) Agent Based Modeling in Computer Graphics and Games. In R.A.Meyers (ed.) *Encyclopedia of Complexity and Systems Science*,Springer. doi:10.1007/978-1-4614-1800-9_39

Agent Based Modeling in Computer Graphics and Games

Brian Mac Namee
Dublin Institute of Technology, brian.macnamee@dit.ie

Agent-Based Modelling in Computer Graphics and Games

BRIAN MAC NAMEE
School of Computing, Dublin Institute of Technology, Dublin, Ireland

Article Outline

Glossary

Computer generated imagery (CGI) The use of computer generated images for special effects purposes in film production.

Intelligent agent A hardware or (more usually) software-based computer system that enjoys the properties autonomy, social ability, reactivity and pro-activeness.

Non-player character (NPC) A computer controlled character in a computer game – as opposed to a player controlled character.

Virtual character A computer generated character that populates a virtual world.

Virtual world A computer generated world in which places, objects and people are represented as graphical (typically three dimensional) models.

Definition of the Subject

As graphics technology has improved in recent years, more and more importance has been placed on the behavior of virtual characters in applications set in virtual worlds in areas such as games, movies and simulations. The behavior of virtual characters should be believable in order to create the illusion that these virtual worlds are populated with living characters. This has led to the application of agent-based modeling to the control of these virtual characters. There are a number of advantages of using agent-based modeling techniques which include the fact that they remove the requirement for hand controlling all agents in a virtual environment, and allow agents in games to respond to unexpected actions by players.

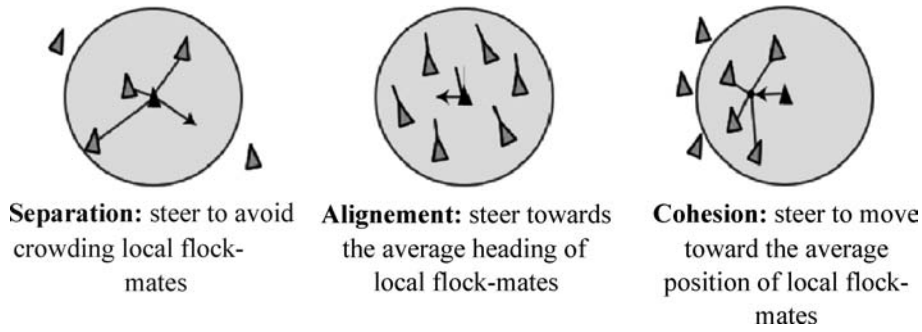
Introduction

Advances in computer graphics technology in recent years have allowed the creation of realistic and believable virtual worlds. However, as such virtual worlds have been developed for applications spanning games, education and movies it has become apparent that in order to achieve real believability, virtual worlds must be populated with life-like virtual characters. This is where the application of agent-based modeling has found a niche in the areas of computer graphics and, in a huge way, computer games. Agent-based modeling is a perfect solution to the problem of controlling the behaviors of the virtual characters that populate a virtual world. In fact, because virtual characters are embodied autonomous agents these applications require an even stronger notion of agency than many other areas in which agent-based modeling is employed.

Before proceeding any further, and because there are so many competing alternatives, it is worth explicitly stating the definition of an intelligent agent that will inform the remainder of this article. Taken from [83] an intelligent agent is defined as "… *a hardware or (more usually) software-based computer system that enjoys the following properties:*

- ***autonomy:*** *agents operate without the direct intervention of humans or others, and have some kind of control over their actions and internal state;*
- ***social ability:*** *agents interact with other agents (and possibly humans) via some kind of agent-communication language;*
- ***reactivity:*** *agents perceive their environment, (which may be the physical world, a user via a graphical user interface, a collection of other agents, the INTERNET, or perhaps all of these combined), and respond in a timely fashion to changes that occur in it;*
- ***pro-activeness:*** *agents do not simply act in response to their environment, they are able to exhibit goal-directed behavior by taking the initiative."*

Virtual characters implemented using agent-based modeling techniques satisfy all of these properties. The characters that populate virtual worlds should be fully autonomous and drive their own behaviors (albeit sometimes following the orders of a director or player). Virtual characters should be able to interact believably with other characters and human participants. This property is particularly strong in the case of virtual characters used in games which by their nature are particularly interactive. It is also imperative that virtual characters appear to perceive their environments and react to events that occur in that environment, especially the actions of other



Agent-Based Modelling in Computer Graphics and Games, Figure 1
The three rules used by Reynolds' original Boids system to simulate flocking behaviors

characters or human participants. Finally virtual characters should be pro-active in their behaviors and not always require prompting from a human participant in order to take action.

The remainder of this article will proceed as follows. Firstly, a broad overview of the use of agent-based modeling in computer graphics will be given, focusing in particular on the genesis of the field. Following on from this, the focus will switch to the use of agent-based modeling techniques in two particular application areas: computer generated imagery (CGI) for movies, and computer games. CGI has been used to astounding effect in movies for decades, and in recent times has become heavily reliant on agent-based modeling techniques in order to generate CGI scenes containing large numbers of computer generated extras. Computer games developers have also been using agent-based modeling techniques effectively for some time now for the control of non-player characters in games. There is a particularly fine match between the requirements of computer games and agent-based modeling due to the high levels of interactivity required.

Finally, the article will conclude with some suggestions for the future directions in which agent-based modeling technology in computer graphics and games is expected to move.

Agent-Based Modelling in Computer Graphics

The serious use of agent-based modeling in computer graphics first arose in the creation of autonomous groups and crowds – for example, crowds of people in a town square or hotel foyer, or flocks of birds in an outdoor scene. While initially this work was driven by visually unappealing simulation applications such as fire safety testing for buildings [75] focus soon turned to the creation of visually realistic and believable crowds for applications such as movies, games and architectural walk-throughs. Computer graphics researchers realized that creating scenes featuring large virtual crowds by hand (a task that was becoming important for the applications already mentioned) was laborious and time-consuming and that agent-based modeling techniques could remove some of the animator's burden. Rather than requiring that animators hand-craft all of the movements of a crowd, agent-based systems could be created in which each character in a crowd (or flock, or swarm) could drive its own behavior. In this way the behavior of a crowd would emerge from the individual actions of the members of that crowd.

Two of the earliest, and seminal, examples of such systems are Craig Reynolds' Boids system [64] and Tu & Terzopoulos' animations of virtual fish [76]. The Boids system simulates the flocking behaviors exhibited in nature by schools of fish, or flocks of birds. The system was first presented at the prestigious SIGGRAPH conference (www.siggraph.org) in 1987 and was accompanied by the short movie "Stanley and Stella in: Breaking the Ice". Taking influence from the area of artificial life (or aLife) [52], Reynolds postulated that the individual members of a flock would not be capable of complex reasoning, and so flocking behavior must emerge from simple decisions made by individual flock members. This notion of emergent behavior is one of the key characteristics of aLife systems.

In the original Boids system, each virtual agent (represented as a simple particle and known as a *boid*) used just three rules to control its movement. These were separation, alignment and cohesion and are illustrated in Fig. 1. Based on just these three simple rules extremely realistic flocking behaviors emerged. This freed animators from the laborious task of hand-scripting the behavior of each creature within the flock and perfectly demonstrates the advantage offered by agent-based modeling techniques for this kind of application.

The system created by Tu and Terzopoulos took a more complex approach in that they created complex models of biological fish. Their models took into account fish physiology, with a complex model of fish muscular structure, along with a perceptual model of fish vision. Using these they created sophisticated simulations in which properties such as schooling and predator avoidance were displayed. The advantage of this approach was that it was possible to create unique, unscripted, realistic simulations without the intervention of human animators. Terzopoulos has since gone on to apply similar techniques to the control of virtual humans [68].

Moving from animals to crowds of virtual humans, the Virtual Reality Lab at the Ecole Polytechnique Fédérale de Lausanne in Switzerland (vrlab.epfl.ch) led by Daniel Thalmann has been at the forefront of this work for many years. They group currently has a highly evolved system, VICrowd, for the animation of virtual crowds [62] which they model as a hierarchy which moves from individuals to groups to crowds. This hierarchy is used to avoid some of the complications which arise from trying to model large crowds in real time – one of the key gaols of ViCrowd.

Each of the levels in the ViCrowd hierarchy can be modeled as an agent and this is done based on *beliefs*, *desires* and *intentions*. The beliefs of an agent represent the information that the agent possesses about the world, including information about places, objects and other agents. An agent's desires represent the motivations of the agent regarding objectives it would like to achieve. Finally, the intentions of an agent represent the actions that an agent has chosen to pursue. The belief-desire-intention (BDI) model of agency was proposed by Rao and Georgeff [61] and has been used in many other application areas of agent-based modeling.

ViCrowd has been used in ambitious applications including the simulation of a virtual city comprised of, amongst other things, a train station a park and a theater [22]. In all of these environments the system was capable of driving the believable behaviors of large groups of characters in real-time.

It should be apparent to readers from the examples given thus that the use of agent-based modeling techniques to control virtual characters gives rise to a range of unique requirements when compared to the use of agent modeling in other application areas. The key to understanding these is to realize that the goal in designing agents for the control of virtual characters is typically not to design the most efficient or effective agent, but rather to design the most interesting or believable character. Outside of very practical applications such as evacuation simulations, when creating virtual characters, designers are concerned with maintaining what Disney, experts in this field, refer to the *illusion of life* [36].

This refers to the fact that the user of a system must believe that virtual characters are living, breathing creatures with goals, beliefs, desires, and, essentially, lives of their own. Thus, it is not so important for a virtual human to always choose the most efficient or cost effective option available to it, but rather to always choose reasonable actions and respond realistically to the success or failure of these actions. With this in mind, and following a similar discussion given in [32], some of the foremost researchers in virtual character research have the following to say about the requirements of agents as virtual characters.

Loyall writes [46] that "*Believable agents are personality-rich autonomous agents with the powerful properties of characters from the arts.*" Coming from a dramatic background it is not surprising that Loyall's requirements reflect this. Agents should have strong personality and be capable of showing emotion and engaging in meaningful social relationships.

According to Blumberg [11], "*… an autonomous animated creature is an animated object capable of goal-directed and time-varying behavior*". The work of Blumberg and his group is very much concerned with virtual creatures, rather than humans in particular, and his requirements reflect this. Creatures must appear to make choices which improve their situation and display sophisticated and individualistic movements.

Hayes–Roth and Doyle focus on the differences between "*animate characters*" and traditional agents [27]. With this in mind they indicate that agents' behaviors must be "*variable rather than reliable*", "*idiosyncratic instead of predictable*", "*appropriate rather than correct*", "*effective instead of complete*", "*interesting rather than efficient*", and "*distinctively individual as opposed to optimal*".

Perlin and Goldberg [59] concern themselves with building believable characters "*that respond to users and to each other in real-time, with consistent personalities, properly changing moods and without mechanical repetition, while always maintaining an author's goals and intentions*".

Finally, in characterizing believable agents, Bates [7] is quite forgiving requiring "*only that they not be clearly stupid or unreal*". Such broad, shallow agents must "*exhibit some signs of internal goals, reactivity, emotion, natural language ability, and knowledge of agents … as well as of the … micro-world*".

Considering these definitions [32] identifies the fact that the consistent themes which run through all of the requirements given above match the general goals of agency – virtual humans must display autonomy, reactivity, goal driven behavior and social ability – and again supports the use of agent-based modeling to drive the behavior of virtual characters.

The Spectrum of Agents

The differences between the systems mentioned in the previous discussion are captured particularly well on the *spectrum of agents* presented by Aylett and Luck [5]. This positions agent systems on a spectrum based on their capabilities, and serves as a useful tool in differentiating between the various systems available. One end of this spectrum focuses on *physical agents* which are mainly concerned with simulation of believable physical behavior, including sophisticated physiological models of muscle and skeleton systems, and of sensory systems. Interesting work at this end of the spectrum includes Terzopoulos' highly realistic simulation of fish [76] and his virtual stuntman project [21] which creates virtual actors capable of realistically synthesizing a broad repertoire of lifelike motor skills.

Cognitive agents inhabit the other end of the agent spectrum and are mainly concerned with issues such as reasoning, decision making, planning and learning. Systems at this end of the spectrum include Funge's cognitive modeling approach [26] which uses the situation calculus to control the behavior of virtual characters, and Nareyek's work on planning agents for simulation [55], both of which will be described later in this article.

While the systems mentioned so far sit comfortably at either end of the agent spectrum, many of the most effective inhabit the middle ground. Amongst these are *c4* [13], used to great effect to simulate a virtual sheep dog with the ability to learn new behaviors, *Improv* [59] which augments sophisticated physical human animation with scripted behaviors and the *ViCrowd* system [62] which sits on top of a realistic virtual human animation system and uses planning to control agents' behavior.

Virtual Fidelity

The fact that so many different agent-based modeling systems, all for the control of virtual humans exist gives rise to the question why? The answer to this lies in the notion of *virtual fidelity*, as described by Badler [6]. Virtual fidelity refers to the fact that virtual reality systems need only remain true to actual reality in so much as this is required by, and improves, the system.

In [47] the point is illustrated extremely effectively. The article explains that when game designers are architecting the environments in which games are set, the scale to which these environments are created is not kept true to reality. Rather, to ease players' movement in these worlds, areas are designed to a much larger scale, compared to character sizes, than in the real world. However, game players do not notice this digression from reality, and in fact have a negative response to environments that are designed to be more true to life finding them cramped. This is a perfect example of how, although designers stay true to reality for many aspects of environment design, the particular blend of virtual fidelity required by an application can dictate certain real world restrictions can be ignored in virtual worlds.

With regard to virtual characters, virtual fidelity dictates that the set of capabilities which these characters should display is determined by the application which they are to inhabit. So, the requirements of an agent-based modeling system for CGI in movies would be very different to those of a agent-based modeling system for controlling the behaviors of game characters.

Agent-Based Modelling in CGI for Movies

With the success of agent-based modeling techniques in graphics firmly established there was something of a search for application areas to which they could be applied. Fortunately, the success of agent-based modeling techniques in computer graphics was paralleled with an increase in the use of CGI in the movie industry, which offered the perfect opportunity. In many cases CGI techniques were being used to replace traditional methods for creating expensive, or difficult to film scenes. In particular, scenes involving large numbers of people or animals were deemed no longer financially viable when set in the real world. Creating these scenes using CGI involved painstaking hand animation of each character within a scene, which again was not financially viable.

The solution that agent-based modeling offers is to make each character within a scene an intelligent agent that drives its own behavior. In this way, as long as the initial situation is set up correctly scenes will play out without the intervention of animators. The facts that animating for movies does not need to be performed in real-time, and is in no way interactive (there are no human users involved in the scene), make the use of agent-based modeling a particularly fine match for this application area.

Craig Reynolds' Boids system [64] which simulates the flocking behaviors exhibited in nature by schools of fish, or flocks of birds and was discussed previously is one of the seminal early examples of agent-based modeling techniques being used in movie CGI. Reynold's approach was first used for CGI in the 1999 film "Batman Returns" [14] to simulate swarms of bats. Reynold's technologies have been used in "The Lion King" [4] and "From Dusk 'Till Dawn" [65] amongst others. Reynolds' approach was so successful, in fact, that he was awarded an Academy Award for his work in 1998.

Similar techniques to those utilized in the Boids system have been used in many other films to animate such diverse characters as ants, people and stampeding wildebeest. Two productions which were released in the same year, "Antz" [17] by Dreamworks and "A Bug's Life" [44] by Pixar took great steps in animating large crowds for CGI effects. For "Antz" systems were developed which allowed animators easily create scenes containing large numbers of virtual characters modeling each as an intelligent agent capable of obstacle avoidance, flocking and other behaviors. Similarly, the creators of "A Bug's Life" created tools which allowed animators easily combine predefined motions (known as alibis) to create behaviors which could easily be applied to individual agents in scenes composed of hundreds of virtual characters.

However, the largest jump in the use of agent-based modeling in movie CGI was made in the recent Lord of the Rings trilogy [33,34,35]. In these films the bar was raised markedly in terms of the sophistication of the virtual characters displayed and the sheer number of characters populating each scene. To achieve the special effects shots required by the makers of these films, the Massive software system was developed by Massive Software (www.massivesoftware.com). This system [2,39] uses agent-based modeling techniques, again inspired by aLife, to create virtual extras that control their own behaviors. This system was put to particularly good use in the large scale battle sequences that feature in all three of the Lord of the Rings films. Some of the sequences in the final film of the trilogy, the Return of the King, contain over 200,000 digital characters.

In order to create a large battle scene using the Massive software, each virtual extra is represented as an intelligent agent, making its own decisions about which actions it will perform based on its perceptions of the world around it. Agent control is achieved through the use of fuzzy logic based controllers in which the state of an agent's brain is represented as a series of motivations, and knowledge it has about the world – such as the state of the terrain it finds itself on, what kinds of other agents are around it and what these other agents are doing. This knowledge about the world is perceived through simple simulated visual, auditory and tactile senses. Based on the information they perceive agents decide on a best course of action. Designing the brains of these agents is made easier that it might seem at first by the fact that agents are developed for short sequences, and so a short range of possible tasks. So for example, separate agent models would be used for a fighting scene and celebration scene.

In order to create a large crown scene using Massive animators initially set up an environment populating it with an appropriate cast of virtual characters where the brains of each character are slight variations (based on physical and personality attribute) of a small number of archetypes. The scene will then play itself out with each character making it's own decisions. Therefore there is no need for any hand animation of virtual characters. However, directors can view the created scenes and by tweaking the parameters of the brains of the virtual characters have a scene play out in the exact way that they require.

Since being used to such impressive effect in the Lord of the Rings trilogy (the developers of the Massive system were awarded an academy award for their work), the Massive software system has been used in numerous other films such as "I, Robot" [60], "The Chronicles of Narnia: The Lion, the Witch and the Wardrobe" [1] and "Ratatouille" [10] along with numerous television commercials and music videos.

While the achievements of using agent-based modeling for movie CGI are extremely impressive, it is worth noting that none of these systems run in real-time. Rather, scenes are rendered by banks of high powered computers, a process that can take hours for relatively simple scenes. For example, the famous Prologue battle sequence in the "Lord of the Rings: The Fellowship of the Ring" took a week to render. When agent-based modeling is applied to the real-time world of computer games, things are very different.

Agent-Based Modelling in Games

Even more so than in movies, agent-based modeling techniques have been used to drive the behaviors of virtual characters in computer games. As games have become graphically more realistic (and in recent years they have become extremely so) game-players have come to expect that games are set in hugely realistic believable virtual worlds. This is particularly evident in the widespread use of realistic physics modeling which is now commonplace in games [67]. In games that make strong use of physics modeling objects in the game world topple over when pushed, float realistically when dropped in water and generally respond as one would expect them to. Players expect the same to be true of the virtual characters that populate virtual game worlds. This can be best achieved by modeling virtual characters as embodied virtual agents. However, there are a number of constraints which have a major influence on the use of agent-based modeling techniques in games.

The first of these constraints stems from the fact that modern games are so highly interactive. Players expect to be able to interact with all of the characters they encounter within a game world. These interactions can be as simple as having something to shoot at or having someone to race against; or involve much more sophisticated interactions in which a player is expected to converse with a virtual character to find out specific information or to cooperate with a virtual character in order to accomplish some task that is key to the plot of a game. Interactivity raises a massive challenge for practitioners as there is very little restriction in terms of what the player might do. Virtual characters should respond in a believable way at all times regardless of how bizarre and unexpected the actions of the player might be.

The second challenge comes from the fact that the vast majority of video games should run in real time. This means that the computational complexity must be kept to a reasonable level as there are only a finite number of processor cycles available for AI processing. This problem is magnified by the fact that an enormous amount of CPU power it usually dedicated to graphics processing. When compared to the techniques that can be used for controlling virtual characters in films some of the techniques used in games are rudimentary due to this real-time constraint.

Finally, modern games resemble films in the fact that creators go to create lengths to include intricate storylines and control the building of tension in much the way that film script writers do. This means that games are tested heavily in order to ensure that the game proceeds smoothly and that the level of difficulty is finely tuned so as to always hold the interest of a player. In fact, this testing of games has become something of a science in itself [77]. Using autonomous agents gives game characters the ability to do things that are unexpected by the game designers and so upset their well laid plans. This can often be a barrier to the use of sophisticated techniques such as learning.

Unfortunately there is also a barrier to the discussion of agent-based modeling techniques used in commercial games. Because of the very competitive nature of the games industry, game development houses often consider the details of how their games work as valuable trade secrets to be kept well guarded. This can make it difficult to uncover the details of how particularly interesting features of a game are implemented. While this situation is improving – more commercial game developers are speaking at games conferences about how their games are developed and the release of game systems development kits for the development of game modifications (or *mods*) allows researchers to plumb the depths of game code – it is still often impossible to find out the implementation details of very new games.

Game Genres

Before discussing the use of agent-based modeling in games any further, it is worth making a short clarification on the kinds of computer games that this article refers to. When discussing modern computer games, or video games, this article does not refer to computer implementations of traditional games such as chess, backgammon or card games such as solitaire. Although these games are of considerable research interest (chess in particular has been the subject of extremely successful research [23]) they are typically not approached using agent-based modeling techniques. Typically, artificial intelligence approaches to games such as these rely largely on sophisticated searching techniques which allow the computer player to search through a multitude of possible future situations dictated by the moves it will make and the moves it expects its opponent to make in response. Based on this search, and some clever heuristics that indicate what constitutes a good game position for the computer player, the best sequence of moves can be chosen. This searching technique relies on the fact that there are usually a relatively small number of moves that a player can make at any one time in a game. However, he fact that the ancient Chinese game of Go-Moku has not, to date, been mastered by computer players [80] illustrates the restrictions of such techniques.

The common thread linking together the kinds of games that this article focuses on is that they all contain computer controlled virtual characters that possess a strong notion of agency. Efforts are often made to separate the many different kinds of modern video games that are the focus of this article into a small set of descriptive genres. Unfortunately, much like in music, film and literature, no categorization can hope to perfectly capture the nuances of all of the available titles. However, a brief mention of some of the more important game genres is worth while (a more detailed description of game genres, and artificial intelligence requirements of each is given in [41]).

The most popular game genre is without doubt the *action game* in which the player must defeat waves of demented foes, typically (for increasingly bizarre motivations) bent upon global destruction. Illustrative examples of the genre include *Half-Life 2* (www.half-life2.com) and the *Halo* series (www.halo3.com). A screenshot of the upcoming action game *Rogue Warrior* (www.bethsoft.com) is shown in Fig. 2.

Agent-Based Modelling in Computer Graphics and Games, Figure 2
A screenshot of the upcoming action game Rogue Warrior from Bethesda Softworks (image courtesy of Bethesda Softworks)

Strategy games allow players to control large armies in battle with other people, or computer controlled opponents. Players do not have direct control over their armies, but rather issue orders which are carried out by agent-based artificial soldiers. Well regarded examples of the genre include the *Age of Empires* (www.ageofempires.com) and *Command & Conquer* (www.commandandconquer.com) series.

Role playing games (such as the *Elder Scrolls* (www.elderscrolls.com) series) place game players in expansive virtual worlds across which they must embark on fantastical quests which typically involve a mixture of solving puzzles, fighting opponents and interacting with non-player characters in order to gain information. Figure 3 shows a screenshot of the aforementioned role-playing game *The Elder Scrolls IV: Oblivion*.

Almost every sport imaginable has at this stage been turned into a computer based *sports game*. The challenges in developing these games are creating computer controlled opponents and team mates that play the games at a level suitable to the human player. *FIFA Soccer 08* (www.fifa08.ea.com) and *Forza Motorsport 2* (www.forzamotorsport.net).

Finally, many people expected that the rise of massively multi-player online games (MMOGs), in which hundreds of human players can play together in an online world, would sound the death knell for the use of virtual non-player characters in games. Examples of MMOGs include *World of Warcraft* (www.worldofwarcraft.com) and *Battlefield 2142* (www.battlefield.ea.com). However, this has not turned out to be the case as there are still large numbers of single player games being produced and even MMOGs need computer controlled characters for roles that players do not wish to play.

Of course there are many games that simply do not fit into any of these categorizations, but that are still relevant for a discussion of the use of agent-based techniques – for example *The Sims* (www.thesims.ea.com) and the *Microsoft Flight Simulator* series (www.microsoft.com/games/flightsimulatorx). However the categorization still serves to introduce those unfamiliar with the subject to the kinds of games up for discussion.

Implementing Agent-Based Modelling Techniques in Games

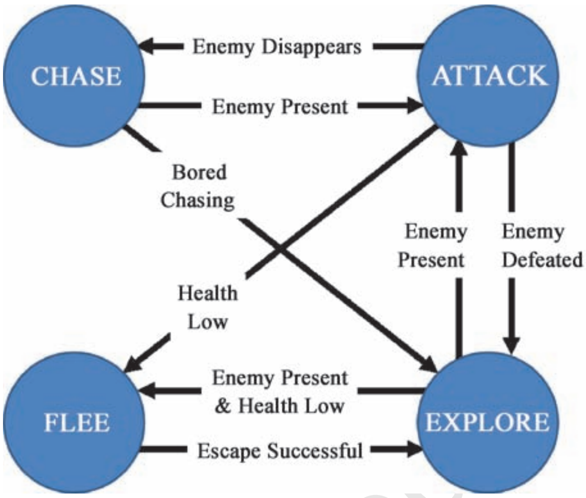
One of the earliest examples of using agent-based modelling techniques in video games was its application to path planning. The ability of non-player characters (NPCs) to manoeuvre around a game world is one of the most basic competencies required in games. While in very early games it was sufficient to have NPCs move along pre-scripted paths, this soon become unacceptable. Games programmers soon began to turn to AI techniques which might be applied to solve some of the problems that were arising. The A* path planning algorithm [74] was the first example of such a technique to find wide-spread use in the games industry. Using the A* algorithm NPCs can be given the ability to find their own way around an environment. This was put to particularly fine effect early on in real-time strategy games where the units controlled by players are semi-autonomous and are given orders rather

Agent-Based Modelling in Computer Graphics and Games, Figure 3
A screenshot from Bethesda Softwork's role playing game The Elder Scrolls IV: Oblivion (image courtesy of Bethesda Softworks)

than directly controlled. In order to use the A* algorithm a game world must be divided into a series of cells each of which is given a rating in terms of the effort that must be expended to cross it. The A* algorithm then performs a search across these cells in order to find the shortest path that will take a game agent from a start position to a goal.

Since becoming widely understood amongst the game development community many interesting additions have been made to the basic A* algorithm. It was not long before three dimensional versions of the algorithm became commonly used [71]. The basic notion of storing the energy required to cross a cell within a game world has also been extended to augment cells with a wide range of other useful information (such as the level of danger in crossing a cell) that can be used in the search process [63].

The next advance in the kind of techniques being used to achieve agent-based modeling in games was the finite state machine (FSM) [30]. An FSM is a simple system in which a finite number of *states* are connected in a directed graph by *transitions* between these states. When used for the control of NPCs, the nodes of an FSM indicate the possible actions within a game world that an agent can perform. Transitions indicate how changes in the state of the game world or the character's own attributes (such as health, tiredness etc) can move the agent from one state to another.



Agent-Based Modelling in Computer Graphics and Games, Figure 4
A simple finite state machine for a soldier NPC in an action game

Figure 4 shows a sample FSM for the control of an NPC in a typical action game. In this example the behaviors of the character are determined by just four states – CHASE, ATTACK, FLEE and EXPLORE. Each of these states provides an action that the agent should take. For example, when in the EXPLORE state the character should wander randomly around the world, or while in the FLEE state the character should determine a direction to move in that will take it away from its current enemy and move in that direction. The links between the states show how the behaviors of the character should move between the various available states. So, for example, if while in the ATTACK state the agent's health measure becomes low, they will move to the FLEE state and run away from their enemy.

FSMs are widely used because they are so simple, well understood and extremely efficient both in terms of processing cycles required and memory usage. There have also been a number of highly successful augmentations to the basic state machine model to make them more effective, such as the introduction of layers of parallel state machines [3], the use of fuzzy logic in finite state machines [19] and the implementation of cooperative group behaviors through state machines [72].

The action game *Halo 2* is recognized as having a particularly good implementation of state machine based NPC control [79]. At any time an agent could be in any one of the four states *Idle*, *Guard/Patrol*, *Attack/Defend*, and *Retreat*. Within each of these states a set of rules was used in order to select from a small set of appropriate actions for that state – for example a number of different ways to attack the player. The decisions made by NPCs were influenced by a number of character attributes including strength, speed and cowardliness. Transition between states was triggered by perceptions made by characters simulated senses of vision and hearing and by internal attributes such as health. The system implemented also allowed for group behaviors allowing NPCs to hold conversations and cooperate to drive vehicles.

However, FSMs are not without their drawbacks. When designing FSMs developers must envisage every possible situation that might confront an NPC over the course of a game. While this is quite possible for many games, if NPCs are required to move between many different situations this task can become overwhelming. Similarly, as more and more states are added to an FSM designing the links between these states can become a mammoth undertaking.

From [31] the definition of rule based systems states that they are "*… comprised of a database of associated rules. Rules are conditional program statements with consequent actions that are performed if the specified conditions are satisfied*". Rule based systems have been applied extensively to control NPCs in games [16], in particular for the control of NPCs in role-playing games. NPCs behaviors are scripted using a set of rules which typically indicate how an NPC should respond to particular events within the game world. Borrowed from [82], the listing below shows a snippet of the rules used to control a warrior character in the RPG *Baldur's Gate* (www.bioware.com).

```
IF
  // If my nearest enemy is not within 3
  !Range(NearestEnemyOf(Myself),3)
  // and is within 8
  Range(NearestEnemyOf(Myself),8)
THEN
  // 1/3 of the time
  RESPONSE #40
    // Equip my best melee weapon
    EquipMostDamagingMelee()
    // and attack my nearest enemy, checking every 60
    // ticks to make sure he is still the nearest
    AttackReevalutate(NearestEnemyOf (Myself),60)
  // 2/3 of the time
  RESPONSE #60
    // Equip a ranged weapon
    EquipRanged()
    // and attack my nearest enemy, checking every 30
    // ticks to make sure he is still the nearest
    AttackReevalutate(NearestEnemyOf (Myself), 30)
```

The implementation of an NPC using a rule-based system would consist of a large set of such rules, a small set of which would fire based on the conditions in the world at any given time. Rule based systems are favored by game developers as they are relatively simple to use and can be exhaustively tested. Rule based systems also have the advantage that rule sets can be written using simple proprietary scripting systems [9], rather than full programming languages, making them easy to implement. Development companies have also gone so far as to make these scripting languages available to the general public, enabling them to author there own rule sets.

Rule based systems, however, are not without their drawbacks. Authoring extensive rule sets is not a trivial task, and they are usually restricted to simple situations. Also, rule based systems can be restrictive in that they don't allow sophisticated interplay between NPCs motivations, and require that rule set authors foresee every situation that the NPC might find itself in.

Some of the disadvantages of simple rule based systems can be alleviated by using more sophisticated inference engines. One example uses Dempster Schafer theory [43] which allows rules to be evaluated by combining multiple sources of (often incomplete) evidence to determine actions. This goes some way towards supporting the use of rule based systems in situations where complete knowledge is not available.

ALife techniques have also been applied extensively in the control of game NPCs, as much as a philosophy as in any particular techniques. However, the outstanding example of this is *The Sims* (thesims.ea.com) a surprise hit of 2000 which has gone on to become the best selling PC game of all time. Created by games guru Will Wright the Sims puts the player in control of the lives of a virtual family in their virtual home. Inspired by aLife the characters in the game have a set of motivations, such as hunger, fatigue and boredom and seek out items within the game world that can satisfy these desires. Virtual characters also develop sophisticated social relationships with each other based on common interest, attraction and the amount of time spent together. The original system in the Sims has gone on to be improved in the sequel *The Sims 2* and a series of expansion packs.

Some of the more interesting work in developing techniques for the control of game characters (particularly in action games) has been focused on developing interesting sensing and memory models for game characters. Players expect when playing action games that computer controlled opponents should suffer from the same problems that they do when perceiving the world. So, for example, computer controlled characters should not be able to see through walls or from one floor to the next. Similarly, though, players expect computer controlled characters to be capable of perceiving events that occur in a world and so NPCs should respond appropriately to sound events or on seeing the player.

One particularly fine example of a sensing model was in the game *Thief: The Dark Project* where players are required to sneak around an environment without alerting guards to their presence [45]. The developers of *Thief 2* produced a relatively sophisticated sensing model that was used by non-player characters which modeled visual effects such as not being able to see the player if they were in shadows, and moving some way towards modeling acoustics so that non-player characters could respond reasonably to sound events.

2004's *Fable* (fable.lionhead.com) took the idea of adding memory to a game to new heights. In this adventure game the player took on the role of a hero from boyhood to manhood. However, every action the player took had an impact on the way in which the game world's population would react to him or her as they would remember every action the next time they met the player. This notion of long-term consequences added an extra layer of believability to the game-playing experience.

Serious Games & Academia

It will probably have become apparent to most readers of the previous section that much of the work done in implementing agent-based techniques for the control of NPCs in commercial games is relatively simplistic when compared to the application of these techniques in other areas of more academic focus, such as robotics [54]. The reasons for this have been discussed already and briefly relate to the lack of available processing resources and the requirements of commercial quality control. However, a large amount of very interesting work is taking place in the application of agent-based technologies in academic research, and in particular the field of serious games. This section will begin by introducing the area of serious games and then discuss interesting academic projects looking at agent-based technologies in games.

The term serious games [53] refers to games designed to do more than just entertain. Rather, serious games, while having many features in common with conventional games, have ulterior motives such as teaching, training, and marketing. Although games have been used for ends apart from entertainment, in particular education, for a long time, the modern serious games movement is set apart from these by the level of sophistication of the games it creates. The current generation of serious games is comparable with main-stream games in terms of the quality of production and sophistication of their design. Serious games offer particularly interesting opportunities for the use of agent-based modeling techniques due to the facts that they often do not have to live up to the rigorous testing of commercial games, can have the requirement of specialized hardware rather than being restricted to commercial games hardware and often, by the nature of their application domains, require more in-depth interactions between players and NPCs.

The modern serious games movement can be said to have begun with the release of *America's Army* (www.americasarmy.com) in 2002 [57]. Inspired by the realism of commercial games such as the *Rainbow 6* series (www.rainbow6.com), the United States military developed America's Army and released it free of charge in order to give potential recruits a flavor of army life. The game was hugely successful and is still being used today as both a recruitment tool and as an internal army training tool.

Spurred on by the success of America's Army the serious games movement began to grow, particularly within academia. A number of conferences sprung up and notably the Serious Games Summit became a part of the influential Game Developer's Conference (www.gdconf.com) in 2004.

Some other notable offerings in the serious games field include *Food Force* (www.food-force.com) [18], a game developed by the United Nations World Food Programme in order to promote awareness of the issues surrounding emergency food aid; *Hazmat Hotzone* [15], a game developed by the Entertainment Technology Centre at Carnegie Mellon University to train fire-fighters to deal with chemical and hazardous materials emergencies; *Yourself!Fitness* (www.yourselffitness.com) [53] an interactive virtual personal trainer developed for modern games consoles; and *Serious Gordon* (www.seriousgames.ie) [50] a game developed to aid in teaching food safety in kitchens. A screen shot of Serious Gordon is shown in Fig. 5.

Over the past decade, interest in academic research that is directly focused on artificial intelligence, and in particular agent-based, techniques and their application to games (as opposed to the general virtual character/computer graphics work discussed previously) has grown dramatically. One of the first major academic research projects into the area of Game-AI was lead by John Laird at the University of Michigan, in the United States. The SOAR architecture was developed in the early nineteen eighties in an attempt to *"develop and apply a unified theory of human and artificial intelligence"* [66]. SOAR is essentially a rule based inference system which takes the current state of a problem and matches this to production rules which lead to actions.

After initial applications into the kind of simple puzzle worlds which characterized early AI research [42], the SOAR architecture was applied to the task of controlling computer generated forces [37]. This work lead to an obvious transfer to the new research area of game-AI [40].

Initially the work of Laird's group focused on applying the SOAR architecture to the task of controlling NPC opponents in the action game *Quake* (www.idsoftware.com) [40]. This proved quite successful leading to opponents which could successfully play against human players, and even begin to plan based on anticipation of what the player was about to do. More recently Laird's group have focused on the development of a game which requires more involved interactions between the player and the NPCs. Named *Haunt 2*, this game casts the player in the role of a ghost that must attempt to influence the actions of a group of computer controlled characters inhabiting the ghost's haunted house [51]. The main issue that arises with the use the SOAR architecture is that it is enormously resource hungry, with the NPC controllers running on a separate machine to the actual game.

At Trinity College in Dublin in Ireland, the author of this article worked on an intelligent agent architecture, the Proactive Persistent Agent (PPA) architecture, for the control of background characters (or support characters) in character-centric games (games that focus on character interactions rather than action, e. g. role-playing games) [48,49]. The key contributions of this work was that it made possible the creation of NPCs that were capable of behaving believably in a wide range of situations and allowed for the creation of game environments which it appeared had an existence beyond their interactions with players. Agent behaviors in this work were based on models of personality, emotion, relationships to other characters and behavioral models that changed according to the current role of an agent. This system was used to develop a stand alone game and as apart of a simulation of parts of Trinity College. A screenshot of this second application is shown in Fig. 6.

At Northwestern University in Chicago the Interactive Entertainment group has also applied approaches from more traditional research areas to the problems facing game-AI. Ian Horswill has lead a team which are attempting to use architectures traditionally associated with robotics for the control of NPCs. In [29] Horswill and Zubek consider how perfectly matched the behavior based architectures often used in robotics are with the requirements of NPC control architectures. The group have demonstrated some of their ideas in a test-bed environment built on top of the game Half-Life [38]. The group also looks at issues around character interaction [85] and the many psychological issues associated with creating virtual characters asking how we can create virtual game agents that display all of the foibles that make us relate to characters in human stories [28].

Within the same research group a team led by Ken Forbus have extended research previously undertaken in conjunction with the military [24] and applied it to the problem of terrain analysis in computer strategy games [25]. Their goal is to create strategic opponents which are capable of performing sophisticated reasoning about the terrain in a game world and using this knowledge to identify complex features such as ambush points. This kind of high level reasoning would allow AI opponents play a much more realistic game, and even surprise human players from time to time, something that is sorely missing from current strategy games.

As well as this work which has spring-boarded from existing applications, a number of projects began expressly to tackle problems in game-AI. Two which particularly stand out are the Excalibur Project, led by Alexander Nareyek [55] and work by John Funge [26]. Both of these projects have attempted to applying sophisticated planning techniques to the control of game characters.

Agent-Based Modelling in Computer Graphics and Games, Figure 5
A screenshot of Serious Gordon a serious game developed to aid in the teaching of food safety in kitchens

Agent-Based Modelling in Computer Graphics and Games, Figure 6
Screenshots of the PPA system simulating parts of a college

Nareyek uses constraint based planning to allow game agents reason about their world. By using techniques such as local search Nareyek has attempted to allow these sophisticated agents perform resource intensive planning within the constraints of a typical computer game environment. Following on from this work, the term *anytime agent* was coined to describe the process by which agents actively refine original plans based on changing world conditions. In [56] Narayek describes the directions in which he intends to take this work in future.

Funge uses the situational calculus to allow agents reason about their world. Similarly to Nareyek he has addressed the problems of a dynamic, ever changing world, plan refining and incomplete information. Funge's work uses an extension to the situational calculus which allows the expression of uncertainty. Since completing this work Funge has gone on to be one of the founders of AiLive (www.ailive.net), a middleware company specializing in AI for games.

While the approaches of both of these projects have shown promise within the constrained environments to which they have been applied during research, and work continues on them it remains to be seen whether such techniques can be successfully applied to a commercial game environment and all of the resource constraints that such an environment entails.

One of the most interesting recent examples of agent-based work in the field of serious games is that undertaken by Barry Silverman and his group at the University of Pennsylvania in the United States [69,70]. Silverman models the protagonists in military simulations for use in training programmes and has taken a very interesting approach in his agent models are based on established cognitive science and behavioral science research. While Silverman admits that many of the models described in the cognitive science and behavioral science literature are not well quantified enough to be directly implemented, he has adapted a number of well respected models for his purposes. Silverman's work is an excellent example of the capabilities that can be explored in a serious games setting rather than a commercial game setting, and as such merits an in depth discussion. A high-level schematic diagram of Silverman's approach is shown in Fig. 7 showing the agent architecture used by Silverman's system, PMFserv.

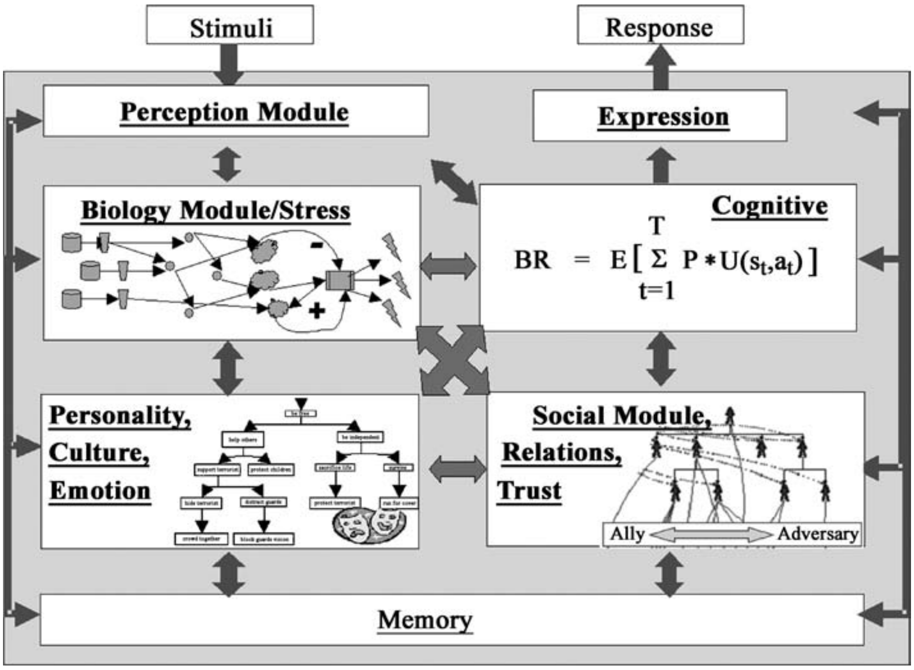
The first important component of the PMFserv system is the biology module which controls biological needs using a metaphor based on the flow of water through a system. Biological concepts such as hunger and fatigue are simulated using a series of reservoirs, tanks and valves which model the way in which resources are consumed by the system. This biological model is used in part to model stress which has an important impact on the way in which agents make decisions. To model the way in which agent performance changes under pressure Silverman uses *performance moderator functions* (PMFs). An example of one of the earliest PMFs used is the Yerkes–Dodson "inverted-u" curve [84] which illustrates that as a stimulus is increased performance initially improves, peaks and then trails off again. In PMFserv a range of PMFs are used to model the way in which behavior should change depending on stress levels and biological conditions.

The second important module of PMFserv attempts to model how personality culture and emotion affect the behavior of an agent. In keeping with the rest of their system PMFserv uses models inspired by cognitive science to model emotions. In this case the well known OCC model [58], which has been used in agent-based applications before [8], is used. The OCC model provides for 11 pairs of opposite emotions such as pride and shame, and hope and fear. The emotional state of an agent with regard to past, current and future actions heavily influences the decisions that the agent makes.

The second portion of the Personality, Culture, Emotion module uses a value tree in order to capture the values of an agent. These values are divided into a Preference Tree which captures long term desired states for the world, a Standards Tree which relates to the actions that an agent believes it can or cannot follow in order to achieve these desired states and a Goal Tree which captures short term goals.

The PMFserv also models the relationships between agents (Social Model, Relations, Trust in Fig. 7). The relationship of one agent to another is modeled in terms of three axes. The first is the degree to which the other agent is thought of as a human rather than an inanimate object relationship – locals tend to view American soldiers as objects rather than people. The second axis is the cognitive grouping (ally, foe etc) to which the other agent belongs and whether this is also a group to which the first agent has an affinity. Finally, the valence, or strength, of the relationship is stored. Relationships continually change based on actions that occur within the game world. Like the other modules of the system this model is also based on psychological research [58].

The final important module of the PMFserv architecture is the Cognitive module which is used to decide on particular actions that agents will undertake. This module uses inputs from all of the other modules to make these decisions and so the behavior of PMFserv agents is driven by their stress levels, relationships to other agents and objects within the game world, personality, culture and emotions. The details of the PMFserv cognitive process are beyond



Agent-Based Modelling in Computer Graphics and Games, Figure 7
A schematic diagram of the main components of the PMFserv system (with kind permission of Barry Silverman)

Agent-Based Modelling in Computer Graphics and Games, Figure 8
A screenshot of the PMFserv system being used to simulate the Black Hawk Down scenario (with kind permission of Barry Silverman)

the scope of this article, so it will suffice to say that action selection is based on a calculation of the utility of a particular action to an agent, with this calculation modified by the factors listed above.

The most highly developed example using the PMFserv model is a simulation of the 1993 event in Mogadishu, Somalia in which a United States military Black Hawk helicopter crashed, as made famous by the book and film "Black Hawk Down" [12]. In this example, which was developed as a military training aid as part of a larger project looking at agent implementations within such systems [78,81] the player took on the role of a US army ranger on a mission to secure the helicopter wreck in a modification (or "mod") of the game *Unreal Tournament* (www.unreal.com). A screenshot of this simulation is shown in Fig. 8.

The PMFserv system was used to control the behaviors of characters within the game world such as Somali militia, and Somali civilians. These characters were imbued with physical attributes, a value system and relationships with other characters and objects within the game environment. The sophistication of PMFserv was apparent in many of the behaviors of the simulations NPCs. One particularly good example was the fact that Somali women would offer themselves as human shields for militia fighters. This behavior was never directly programmed into the agents make-up, but rather emerged as a result of their values and assessment of their situation. PMFserv remains one of the most sophisticated current agent implementations and shows the possibilities when the shackles of commercial game constraints are thrown off.

Future Directions

There is no doubt that with the increase in the amount of work being focused on the use of agent-based modeling in computer graphics and games that there will be large developments in the near future. This final section will attempt to predict what some of these might be.

The main development that might be expected in all of the areas that have been discussed in this article is an increase in the depth of simulation. The primary driver of this increase in depth will be the development of more sophisticated agent models which can be used to drive ever more sophisticated agent behavior. The PMFserv system described earlier is one example of the kinds of deeper systems that are currently being developed. In general computer graphics applications this will allow for the creation of more interesting simulations including previously prohibitive features such as automatic realistic facial expressions and other physical expressions of agents' internal states. This would be particularly use in CGI for movies in which, although agent based modeling techniques are commonly used for crowd scenes and background characters, main characters are still animated almost entirely by hand.

In the area of computer games it can be expected that many of the techniques being used in movie CGI will filter over to real-time game applications as the processing power of game hardware increases – this is a pattern that has been evident for the past number of years. In terms of depth that might be added to the control of game characters one feature that has mainly been conspicuous by its absence in modern games is genuine learning by game agents. 2000's *Black & White* and its sequel *Black & White 2* (www.lionhead.com) featured some learning by one of the game's main characters that the player could teach in a reinforcement manner [20]. While this was particular successful in the game, such techniques have not been more widely applied. One interesting academic project in this area is NERO project (www.nerogame.org) which allows a player to train an evolving army of soldiers and have them battle the armies of other players [73]. It is expected that these kinds of capabilities will become more and more common in commercial games.

One new feature of the field of virtual character control in games is the emergence of specialized middleware. Middleware has had a massive impact in other areas of game development including character modeling (for example Maya available from www.autodesk.com) and physics modeling (for example Havok available from www.havok.com). AI focused middleware for games is now becoming more common with notable offerings including AI-Implant (www.ai-implant.com) and Kynogon (www.kynogon.com) which perform path finding and state machine based control of characters. It is expected that more sophisticated techniques will over time find their way into such software.

To conclude the great hope for the future is that more and more sophisticated agent-based modeling techniques from other application areas and other branches of AI will find their way into the control of virtual characters.

Bibliography

Primary Literature

1. Adamson A (Director) (2005) The Chronicles of Narnia: The Lion, the Witch and the Wardrobe. Motion Picture. http://adisney.go.com/disneypictures/narnia/lb_main.html TS2
2. Aitken M, Butler G, Lemmon D, Saindon E, Peters D, Williams G (2004) The Lord of the Rings: the visual effects that brought middle earth to the screen. International Conference on

Computer Graphics and Interactive Techniques (SIGGRAPH), Course Notes CE3

3. Alexander T (2003) Parallel-State Machines for Believable Characters. In: CE4 Massively Multiplayer Game Development. Charles River Media
4. Allers R, Minkoff R (Directors) (1994) The Lion King. Motion Picture. http://disney.go.com/disneyvideos/animatedfilms/lionking/ TS2
5. Aylett R, Luck M (2000) Applying Artificial Intelligence to Virtual Reality: Intelligent Virtual Environments. Appl Artif Intell 14(1):3–32
6. Badler N, Bindiganavale R, Bourne J, Allbeck J, Shi J, Palmer M (1999) Real Time Virtual Humans. In: Proceedings of the International conference on Digital Media Futures. CE3
7. Bates J (1992) The Nature of Characters in Interactive Worlds and the Oz Project. Technical Report CMU-CS-92–200. School of Computer Science, Carnegie Melon University
8. Bates J (1992) Virtual reality, art, and entertainment. Presence: J Teleoper Virtual Environ 1(1):133–138
9. Berger L (2002) Scripting: Overview and Code Generation. In: Rabin S (ed) AI Game Programming wisdom. Charles River Media CE5
10. Bird B, Pinkava J (Directors) (2007) Ratatouille. Motion Picture. http://disney.go.com/disneyvideos/animatedfilms/ratatouille/ TS2
11. Blumberg B (1996) Old Tricks, New Dogs: Ethology and Interactive Creatures. PhD Thesis, Media Lab, Massachusetts Institute of Technology CE5
12. Bowden M (2000) Black Hawk Down. Corgi Adult CE5
13. Burke R, Isla D, Downie M, Ivanov Y, Blumberg B (2002) Creature Smarts: The Art and Architecture of a Virtual Brain. In: Proceedings of Game-On 2002: the 3rd International Conference on Intelligent Games and Simulation, pp 89–93 CE3
14. Burton T (Director) (1992) Batman Returns. Motion Picture. http://www.warnervideo.com/batmanmoviesondvd/ TS2
15. Carless S (2005) Postcard From SGS 2005: Hazmat: Hotzone – First-Person First Responder Gaming. Retrieved October 2007, from Gamasutra: www.gamasutra.com/features/20051102/carless_01b.shtml
16. Christian M (2002) A Simple Inference Engine for a Rule Based Architecture. In: Rabin S (ed) AI Game Programming Wisdom. Charles River Media CE5
17. Darnell E, Johnson T (Directors) (1998) Antz. Motion Picture. http://www.dreamworksanimation.com/ TS2
18. DeMaria R (2005) Postcard from the Serious Games Summit: How the United Nations Fights Hunger with Food Force. Retrieved October 2007, from Gamasutra: www.gamasutra.com/features/20051104/demaria_01.shtml
19. Dybsand E (2001) A Generic Fuzzy State Machine in C++. In: Rabin S (ed) Game Programming Gems 2. Charles River Media CE5
20. Evans R (2002) Varieties of Learning. In: Rabin S (ed) AI Game Programming Wisdom. Charles River Media CE5
21. Faloutsos P, van de Panne M, Terzopoulos D (2001) The Virtual Stuntman: Dynamic Characters with a Repetoire of Autonomous Motor Skills. Comput Graph 25(6):933–953
22. Farenc N, Musse S, Schweiss E, Kallmann M, Aune O, Boulic R et al (2000) A Paradigm for Controlling Virtual Humans in Urban Environment Simulations. Appl Artif Intell J Special Issue Intell Virtual Environ 14(1):69–91
23. Feng-Hsiung H (2002) Behind Deep Blue: Building the Computer that Defeated the World Chess Champion. Princeton University Press CE5
24. Forbus K, Nielsen P, Faltings B (1991) Qualitative Spatial Reasoning: The CLOCK Project. Artif Intell 51:1–3
25. Forbus K, Mahoney J, Dill K (2001) How Qualitative Spatial Reasoning Can Improve Strategy Game AIs. In: Proceedings of the AAAI Spring Symposium on AI and Interactive Entertainment CE3
26. Funge J (1999) AI for Games and Animation: A Cognitive Modeling Approach. A.K. Peters CE5
27. Hayes-Roth B, Doyle P (1998) Animate Characters. Auton Agents Multi-Agent Syst 1(2):195–230
28. Horswill I (2007) Psychopathology, narrative, and cognitive architecture (or: why NPCs should be just as screwed up as we are). In: Proceedings of AAAI Fall Symposium on Intelligent Narrative Technologies, CE3
29. Horswill I, Zubek R (1999) Robot Architectures for Believable Game Agents. In: Proceedings of the 1999 AAAI Spring Symposium on Artificial Intelligence and Computer Games. CE3
30. Houlette R, Fu D (2003) The Ultimate Guide to FSMs in Games. In: Rabin S (ed) AI Game Programming Wisdom 2. Charles River Media CE5
31. IGDA (2003) Working Group on Rule-Based Systems Report. International Games Development Association CE5
32. Isbister K, Doyle P (2002) Design and Evaluation of Embodied Conversational Agents: A Proposed Taxonomy. In: Proceedings of the AA- MAS02 Workshop on Embodied Conversational Agents: Lets Specify and Compare Them! Bologna, Italy
33. Jackson P (Director) (2001) The Lord of the Rings: The Fellowship of the Ring. Motion Picture. http://www.lordoftherings.net/ TS2
34. Jackson P (Director) (2002) The Lord of the Rings: The Two Towers. Motion Picture. http://www.lordoftherings.net/ TS2
35. Jackson P (Director) (2003) The Lord of the Rings: The Return of the King. Motion Picture. http://www.lordoftherings.net/ TS2
36. Johnston O, Thomas F (1995) The Illusion of Life: Disney Animation. Disney Editions CE5
37. Jones R, Laird J, Neilsen P, Coulter K, Kenny P, Koss F (1999) Automated Intelligent Pilots for Combat Flight Simulation. AI Mag 20(1):27–42
38. Khoo A, Zubek R (2002) Applying Inexpensive AI Techniques to Computer Games. IEE Intell Syst Spec Issue Interact Entertain 17(4):48–53
39. Koeppel D (2002) Massive Attack. http://www.popsci.com/popsci/science/d726359b9fa84010vgnvcm1000004eecbccdrcrd.html. Accessed Oct 2007
40. Laird J (2000) An Exploration into Computer Games and Computer Generated Forces. The 8th Conference on Computer Generated Forces and Behavior Representation, CE3
41. Laird J, van Lent M (2000) Human-Level AI's Killer Application: Interactive Computer Games. In: Proceedings of the 17th National Conference on Artificial Intelligence, CE3
42. Laird J, Rosenbloom P, Newell A (1984) Towards Chunking as a General Learning Mechanism. The 1984 National Conference on Artificial Intelligence (AAAI), pp 188–192 CE3
43. Laramée F (2002) A Rule Based Architecture Using Dempster-Schafer theory. In: Rabin S (ed) AI Game Programming Wisdom. Charles River Media CE5
44. Lasseter J, Stanton A (Directors) (1998) A Bug's Life; Motion Picture. http://www.pixar.com/featurefilms/abl/ TS2

CE4 Please provide the editors.
CE5 Please provide the publisher location.

Editor's or typesetter's annotations (will be removed before the final TEX run)

45. Leonard T (2003) Building an AI Sensory System: Examining the Deign of Thief: The Dark Project. In: Proceedings of the 2003 Game Developers' Conference, San Jose
46. Loyall B (1997) Believable Agents: Building Interactive Personalities. PhD Thesis, Carnegie Melon University, CE5
47. Määta A (2002) Realistic Level Design for Max Payne. In: Proceedings of the 2002 Game Developer's conference, GDC 2002, CE3
48. Mac Namee B, Cunningham P (2003) Creating Socially Interactive Non Player Characters: The μ-SIC System. Int J Intell Games Simul 2(1) TS6
49. Mac Namee B, Dobbyn S, Cunningham P, O'Sullivan C (2003) Simulating Virtual Humans Across Diverse Situations. In: Proceedings of Intelligent Virtual Agents '03, pp 159–163 CE3
50. Mac Namee B, Rooney P, Lindstrom P, Ritchie A, Boylan F, Burke G (2006) Serious Gordon: Using Serious Games to Teach Food Safety in the Kitchen. The 9th International Conference on Computer Games: AI, Animation, Mobile, Educational & Serious Games CGAMES06, Dublin
51. Magerko B, Laird JE, Assanie M, Kerfoot A, Stokes D (2004) AI Characters and Directors for Interactive Computer Games. The 2004 Innovative Applications of Artificial Intelligence Conference. AAAI Press, San Jose
52. Thalmann MN, Thalmann D (1994) Artificial Life and Virtual Reality. Wiley CE5
53. Michael D, Chen S (2005) Serious Games: Games That Educate, Train, and Inform. Course Technology PTR CE5
54. Muller J (1996) The Design of Intelligent Agents: A Layered Approach. Springer CE5
55. Nareyek A (2001) Constraint Based Agents. Springer CE5
56. Nareyek A (2007) Game AI is Dead. Long Live Game AI! IEEE Intell Syst 22(1):9–11
57. Nieborg D (2004) America's Army: More Than a Game. Bridging the Gap;Transforming Knowledge into Action through Gaming and Simulation. Proceedings of the 35th Conference of the International Simulation and Gaming Association (ISAGA), Munich
58. Ortony A, Clore GL, Collins A (1988) The cognitive structure of emotions. Cambridge University Press, Cambridge
59. Perlin K, Goldberg A (1996) Improv: A System for Scripting Interactive Actors in Virtual Worlds. In: Proceedings of the ACM Computer Graphics Annual Conference, pp 205–216 CE3
60. Proyas A (Director) (2004) I, Robot. Motion Picture. http://www.irobotmovie.com TS2
61. Rao AS, Georgeff MP (1991) Modeling rational agents within a BDI-architecture. In: Proceedings of Knowledge Representation and Reasoning (KR&R-91). Morgan Kaufmann CE3, pp 473–484
62. Musse RS, Thalmann D (2001) A Behavioral Model for Real Time Simulation of Virtual Human Crowds. IEEE Trans Vis Comput Graph 7(2):152–164
63. Reed C, Geisler B (2003) Jumping, Climbing, and Tactical Reasoning: How to Get More Out of a Navigation System. In: Rabin S (ed) AI Game Programming Wisdom 2. Charles River Media CE5
64. Reynolds C (1987) Flocks, Herds and Schools: A Distributed Behavioral Model. Comput Graph 21(4):25–34
65. Rodriguez R (Director) (1996) From Dusk 'Till Dawn. Motion Picture
66. Rosenbloom P, Laird J, Newell A (1993) The SOAR Papers: Readings on Integrated Intelligence. MIT Press, CE5
67. Sánchez-Crespo D (2006) GDC: Physical Gameplay in Half-Life 2. Retrieved October 2007, from gamasutra.com: http://www.gamasutra.com/features/20060329/sanchez_01.shtml
68. Shao W, Terzopoulos D (2005) Autonomous Pedestrians. In: Proceedings of SIGGRAPH/EG Symposium on Computer Animation, SCA'05, pp 19–28 CE3
69. Silverman BG, Bharathy G, O'Brien K, Cornwell J (2006) Human Behavior Models for Agents in Simulators and Games: Part II: Gamebot Engineering with PMFserv. Presence Teleoper Virtual Worlds 15(2):163–185
70. Silverman BG, Johns M, Cornwell J, O'Brien K (2006) Human Behavior Models for Agents in Simulators and Games: Part I: Enabling Science with PMFserv. Presence Teleoper Virtual Environ 15(2):139–162
71. Smith P (2002) Polygon Soup for the Programmer's Soul: 3D Path Finding. In: Proceedings of the Game Developer's Conference 2002 , GDC2002, CE3
72. Snavely P (2002) Agent Cooperation in FSMs for Baseball. In: Rabin S (ed) AI Game Programming Wisdom. Charles River Media CE5
73. Stanley KO, Bryant BD, Karpov I, Miikkulainen R (2006) Real-Time Evolution of Neural Networks in the NERO Video Game. In: Proceedings of the Twenty-First National Conference on Artificial Intelligence, AAAI-2006. AAAI Press, pp 1671–1674 CE3
74. Stout B (1996) Smart Moves: Intelligent Path-Finding. Game Dev Mag Oct TS7
75. Takahashi TS (1992) Behavior Simulation by Network Model. Memoirs of Kougakuin University 73, pp 213–220 CE3
76. Terzopoulos D, Tu X, Grzeszczuk R (1994) Artificial Fishes with Autonomous Locomotion, Perception, Behavior and Learning, in a Physical World. In: Proceedings of the Artificial Life IV Workshop. MIT Press CE5
77. Thompson C (2007) Halo 3: How Microsoft Labs Invented a New Science of Play. Retrieved October 2007, from wired.com: http://www.wired.com/gaming/virtualworlds/magazine/15-09/ff_halo
78. Toth J, Graham N, van Lent M (2003) Leveraging gaming in DOD modelling and simulation: Integrating performance and behavior moderator functions into a general cognitive architecture of playing and non-playing characters. Twelfth Conference on Behavior Representation in Modeling and Simulation (BRIMS, formerly CGF), Scotsdale, Arizona
79. Valdes R (2004) In the Mind of the Enemy: The Artificial Intelligence of Halo 2. Retrieved October 2007, from HowStuffWorks.com: http://entertainment.howstuffworks.com/halo2-ai.htm
80. van der Werf E, Uiterwijk J, van den Herik J (2002) Programming a Computer to Play and Solve Ponnuki-Go. In: Proceedings of Game-On 2002: The 3rd International Conference on Intelligent Games and Simulation, pp 173–177 CE3
81. van Lent M, McAlinden R, Brobst P (2004) Enhancing the behavioral fidelity of synthetic entities with human behavior models. Thirteenth Conference on Behavior Representation in Modeling and Simulation (BRIMS) CE3
82. Woodcock S (2000) AI Roundtable Moderator's Report. In: Proceedings of the Game Developer's Conference 2000 (GDC2000), CE3
83. Wooldridge M, Jennings N (1995) Intelligent Agents: Theory and Practice. Know Eng Rev 10(2):115–152

TS6 Please provide page number.
TS7 Please provide page numbers

Editor's or typesetter's annotations (will be removed before the final TEX run)

84. Yerkes RW, Dodson JD (1908) The relation of strength of stimulus to rapidity of habit formation. J Comp Neurol Psychol 18:459–482
85. Zubek R, Horswill I (2005) Hierarchical Parallel Markov Models of Interaction. In: Proceedings of the Artificial Intelligence and Interactive Digital Entertainment Conference, AIIDE 2005, CE5

Books and Reviews

DeLoura M (ed) (2000) Game Programming Gems. Charles River Media CE5
DeLoura M (ed) (2001) Game Programming Gems 2. Charles River Media CE5
Dickheiser M (ed) (2006) Game Programming Gems 6. Charles River Media CE5
Kirmse A (ed) (2004) Game Programming Gems 4. Charles River Media CE5
Pallister K (ed) (2005) Game Programming Gems 5. Charles River Media CE5
Rabin S (ed) (2002) Game AI Wisdom. Charles River Media CE5
Rabin S (ed) (2003) Game AI Wisdom 2. Charles River Media CE5
Rabin S (ed) (2006) Game AI Wisdom 3. Charles River Media CE5
Russell S, Norvig P (2002) Artificial Intelligence: A Modern Approach. Prentice Hall CE5
Treglia D (ed) (2002) Game Programming Gems 3. Charles River Media CE5