The use of Fuzzy Logic for Artificial Intelligence in Games

Michele Pirovano[1,2]
[1]Department of Computer Science, University of Milano, Milano, Italy
[2]Dipartimento di Ingegneria Elettronica e Informazione, Politecnico di Milano, Italy

December 7, 2012

Abstract

Game artificial intelligence (game AI) is the branch of videogame development that is concerned with empowering games with the illusion of intelligence. Game AI borrows many techniques from the broader field of AI, from simple finite state machines to state-of-the-art evolutionary algorithms. Among these techniques, fuzzy logic is one of the tools that must be present in the arsenal of a good videogame AI developer, due to the simplicity of its formulation coupled with its expressive power. In this paper, we introduce the field of game AI and review the use of fuzzy logic in games, looking both at industry and research. We outline its benefits, address its shortcomings and present its practical uses.

1 Artificial Intelligence and Games

The field of game artificial intelligence encompasses all the techniques and methods for injecting intelligence into video games. To game AI pertain many different aspects of a videogame: animation control, steering, flocking, pathfinding, planning, procedural generation, tactical and strategic thinking, and learning [21, 6]. All these aspects share the same basis, they pose problems whose efficient solution requires AI algorithms. The purpose of game AI is to bring life to the non-playable characters (NPCs) present in the games, such as the enemy troops in a strategy game or the merchants in a peaceful AI-controlled village.

Game AI is just one of the branches of the broader field of artificial intelligence. Academic AI research always suffered from a difficulty to define artificial intelligence itself, although it can be typically identified in the study and recreation of human-like or human-level cognitive processes and in the capability of these processes to learn [16]. According to Alan Turing, who is considered the father of artificial intelligence, an agent is intelligent if its behavior cannot be distinguished from that of a human [33]. Game AI adds another piece to the puzzle of defining AI, as it is not concerned with the actual existence of intelligence, but just with the illusion of it [29]. For a game to be successful, we do not really need highly-intelligent, human-like, possibly unbeatable expert opponents, what is required is a compelling adversary, an AI agent that is fun to play against and that is at least not obviously stupid. Game AI, like all other aspects of a videogame, exists to propel the higher goal of a videogame: to be fun and entertain. In other words, the motto of the game AI society is often thought to be *If the player cannot see it, why do it?* [21]. The context is however not so simple nowadays, as partly due to successful products that have brought forward advanced AI, partly due to the ubiquitous online multi-player options that have changed the habits of players, players are now more demanding in regards to AI and more and more game developers have turned their attention from scripted and predictable agents to more human-like agents, capable of learning and thus, in the classical AI definition, intelligent. Another peculiarity of game AI lies in the fact that games are, for the most part, subject to real-time constraints and thus there is little space for slow, offline techniques. This fact shares similitudes to the AI used in robotics and control systems [25], and this is the reason why the same techniques, in one form or another, can be found in all these fields. At last, we must remember that video games are part of the entertainment industry, an industry well-known for its fast advancements, sudden turns of trends and tight development schedules. This leaves little room for highly innovative yet not throughly tested AI techniques that may hide nasty surprises and detour development. Due to these differences, the game AI field has been advancing in a different but parallel direction in respect to academic AI, borrowing ideas and methods from the latter while always keeping a practical approach in mind [16, 32].

In this paper, we discuss the application of a specific technique to videogame AI: fuzzy logic. In the next section, we examine the current state of the game AI field in order to provide a context for the discussion on fuzzy logic. We introduce the concepts of fuzzy logic and detail its applications for game AI in section 3. In section 5, existing uses of fuzzy logic in the industry and in research are reported and analyzed. In section 6, we draw conclusions on the current situation about the use of fuzzy logic in games.

2 The current state of Game AI

Considered as a last-minute addition in the eighties and nineties, game AI has instead been gaining attention from both academics and industry during the last decade. Traditionally, games have been pushing on their visual and audial compo-

nents in order to attract more consumers. Most CPU processing power had thus to be taken up by the graphics computation, often followed by sound processing and, of course, game logic. This left AI with very little time to perform its computations, which resulted in the ubiquitous stupid hordes of enemies of older games. The rush for better, more realistic and more gratifying graphics has however slowed down, if it has not come to an halt, in the recent years, to the point that some games today are almost indistinguishable from reality. Dedicated graphics processing units (GPUs) now sustain most of the graphics computation, leaving more room in the CPU for other processor intensive components, such as physics and AI. In this context, game AI has been earning its fame as a means to make a game stand out among the competition. Nowadays, more CPU time is given at each game frame to the AI, more than enough for most applications, and some programmers are specifically dedicated in a team to create a challenging, interesting and (seemingly) intelligent AI [37, 38].

On the other hand, academics have started to like to meddle with video games when experimenting with AI techniques, due to the fact that games present a low-cost, safe and simple environment in respect to reality. Instead of testing AI techniques on the field, researchers can safely test them in a virtual environment. As an example, researchers have been creating controllers for virtual cars in the TORCS Simulated Car Racing Competition [18] that sees researchers from all around the world competing to provide cutting-edge controllers for steering a car in a realistic racing simulation, with obvious benefits as a test-suite for automation without crashing a single vehicle.

2.1 Game AI techniques

As we have seen, game AI is the offspring of two, quite different, fields: academics AI and videogame development. On the one hand, it shares the enthusiasm for discovery typical of AI researchers. On the other hand, as with any other game development aspect, it tends to reuse tried and tested methods and to stop when something works *just enough*, due to the developers' struggle with tight availability of resources and strict time schedules imposed by the commercially-oriented entertainment industry. Despite the range of exotic techniques available to AI enthusiasts, game AI has thus always suffered from a reluctance to adopt them. Most games, even today, use very simple AI techniques, some of which would not even be considered AI in the first place from an academic point of view. Finite State Machines (FSMs) stand out above every other technique as the jack-of-all-trades for game AI, used everywhere in computer games due to their simplicity of implementation and to the power of their structure. Decision trees are also found in many games, due to their inherent simplicity. However, these trees are often hidden behind naive implementations consisting of nested if/else clauses. The use of such techniques is mainly due to the strict and short development schedules of games. Given the limited resources, game developers tend to go for the simple yet effective methods, easy to implement and, especially, to test.

Nonetheless, more advanced techniques have been spotted even in commercial games and the proof of their use is in the wealth of examples that can be found in game AI manuals [21, 6]. Bayesian networks are used in Real-Time Strategy games (RTS) for goal planning [21]. Neural networks appear in the Creatures game series[1] as well as in several real-time strategy games and in the award-winning game Black & White[2]. More recently, the use of evolutionary algorithms has been popularized by the Galactic Arms Race videogame [13]. Artificial life is another favorite of game AI developers, of which Creatures and most Maxis games[3] are the biggest exponents. Those games are however the exception to the rule and that whenever advanced AI techniques are inserted into a game, they are part of the main focus of the game itself. In other words, advanced techniques currently need a game built around them. There are three main reasons for this fact. First, those techniques are often difficult to implement in comparison to FSMs or decision trees. This is enough of a drawback for developers rushing for their game to ship. Second, some of the more advanced techniques, for instance genetic algorithms, usually bring with them either a high computational cost or a high memory cost. A last problem arising from the use of more advanced techniques lies in their non-determinism. Developers are reluctant to ship a game that has not been completely tested and the risk of an evolutionary algorithm learning something wrong is too high.

In the large spectrum of game AI techniques, somewhere between the simple techniques and the more advanced ones, we can find fuzzy logic.

3 Fuzzy Logic

Fuzzy logic is a superset of conventional logic that has been extended to handle the concept of partial-truth values between the boolean dichotomy of true and false. Fuzzy logic usually takes the form of a fuzzy reasoning system and its components are fuzzy variables, fuzzy rules and a fuzzy inference engine. Using fuzzy set theory, variables are fuzzified by selecting a set of membership functions on their possible range of values. The membership functions map the crisp value of a variable to a linguistic label with a degree of truth, usually in the range [0,1]. For example, the range of a weapon in a game can be divided into melee, ranged or out-of-range, as seen in Figure 1 (Top). Variables can be considered to be input, such as measures from sensors (even virtual sensors, as in games), or they can be considered to be output. Fuzzy rules can be created using the defined fuzzy variables and their sets. Those rules are usually in the form if-then, with a grammar similar to boolean logic. Rules determine the value of output variables given the current value of input variables. Operators such as AND, OR and NOT are also given a meaning in fuzzy logic. Once the rules have been created, a fuzzy inference engine is used to

[1] Creatures, Gameware Development, 1996
[2] Black&White, Lionhead Studios, 2001
[3] http://www.maxis.com/

infer conclusions from the current values of the variables and the rules that govern the system. Output variables are thus given a fuzzy value. Eventually, a procedure of defuzzification can be performed, transforming the output of the fuzzy engine to a crisp value.

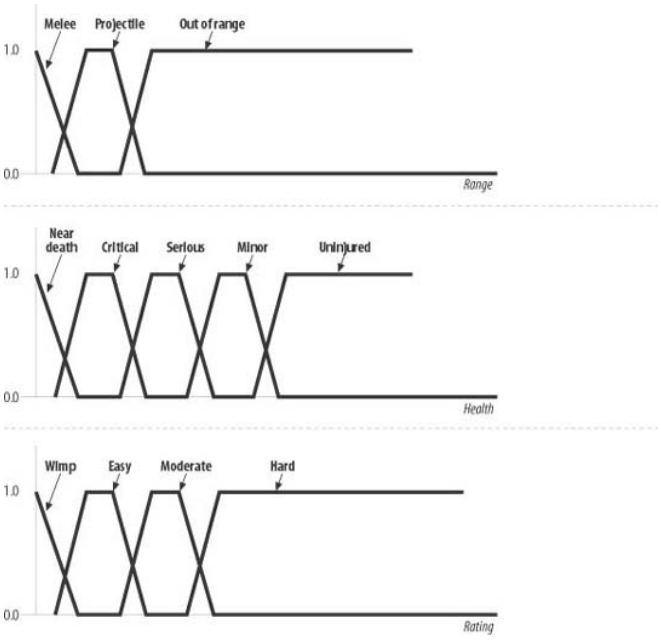

Figure 1: Fuzzy variables in a game. (Top) The fuzzy range of a weapon. (Middle) The fuzzy health of a NPC. (Bottom) The fuzzy ranking of a player. Image taken from [5].

Fuzzy set theory was introduced in 1965 by Lotfi A. Zadeh in the academic artificial intelligence field [40], but his ideas did not earn a following until Japanese researchers demonstrated the practical use of fuzzy control systems. Nowadays, many controllers use fuzzy logic, from dishwashers to electrical switches to hovering helicopters. Fuzzy logic is used with great success in the control systems field due to its similarity to human reasoning, allowing experts of the field, not necessarily programmers, to take part in the design process. Fuzzy logic is also used in conjunction with other AI techniques, such as evolutionary algorithms or neural networks, in learning and classification.

4 Fuzzy Logic in Games

As many other academic AI techniques, fuzzy logic has been tested in videogames pursuing the quest for simple design coupled with intelligent agents. Fuzzy logic fills these requirements as it is rather simple to reason with it due to its language-like nature, resulting in a fast and convenient design methodology [12].

Fuzzy logic has been officially introduced in game development in 1996 in the Game Developer Magazine by Larry O'Brien [22] and has been since explored and improved further by other authors. Fuzzy logic is listed as one of the useful techniques for game AI design by multiple sources. Books on game AI development devote entire chapters to it [21, 6, 5] with plenty of examples, while several articles can be found as an introduction to applying fuzzy logic to games, with [19] being one such example. Zarozinski suggests that fuzzy logic finds its way in almost every game [42].

4.1 Benefits of Fuzzy Logic

Fuzzy logic can be useful to game AI in several aspects. Among other uses, it can be used for NPC decision making such as item or weapon selection, for the control of units' movement similar to what happens with control systems, for enabling an AI opponent to assess threats and for classification, for example by ranking players and NPCs in terms of health or power using fuzzy variables like those in figure 1 (Middle, Bottom).

Fuzzy logic brings many benefits to the game AI developer. The basis of fuzzy logic are simple and there are no prerequisites to it apart from basic boolean logic, something that any AI developer will surely have mastered, making fuzzy logic a good candidate for introducing advanced AI in a game with relatively little effort. As a bonus, the fuzzy membership functions can be defined using the same response curves [2] that are usually used for tuning simpler behaviors in videogames.

Due to the linguistic nature of fuzzy logic, the formulation of its rules can be done by experts in the field domain and the fuzzy system can be then used to emulate the reasoning of the expert [19]. This is a great advantage in respect to other methods that require knowledge of the method itself to be tuned. In the videogame development field, this property comes in handy as the fuzzy rules can be created by the game designers without the need for a programmer to assist them, similar to what already happens in the game industry when using scripting languages to design gameplay sequences. A warning must however be cast as, in order to create challenging enemies, an expert player of the game should be in charge of defining the rule base, but prior to shipping this may not be possible. Due to the recent rise in user-generated content, this step may however be even done in a beta-phase of the game by giving the task to the player themselves.

Traditional decision making, when used to model agent behavior, can result in unnatural sudden switches from one decision to another, or from one state to another in a FSM. More gradual changes can be achieved using fuzzy logic. For example, an AI controlled car without fuzzy logic may brake or accelerate, but with fuzzy logic it may also decide how much to brake and how much to accelerate, achieving a more natural behavior. It is suggested that such gradual output may be over-kill for most current games [21], but nonetheless this is an advantage of fuzzy logic. This gradual behavior change is also done in some games without resorting to fuzzy logic, although fuzzy logic gives a better framework for working with such continuous values.

Fuzzy techniques allow an input-output relationship to be built upon the expert's knowledge, without the need of possi-

bly complex mathematical model that may be tedious or impossible to derive [12]. Due to the freedom on the shape of membership functions, this input-output relationship can be designed to be non-linear, effectively allowing non-linear behaviors to emerge in the game and thus increasing the unpredictability of NPCs.

Another benefit when designing the rule set for a fuzzy system lies in the non-sequential approach of traditional fuzzy logic. Since rules can be resolved in any order, it is far easier to remove or add new rules to a fuzzy system than would be to a typical if-then-else nested block.

Fuzzy logic can be used to model complex behavior with a low computational cost [39]. Due to the low resources available to the game AI developers and to the real-time constraints they are often required to abide to, this benefit is of high importance and is one of the reasons fuzzy is so favored in the gaming industry.

As a last benefit, fuzzy systems are good candidates to be used in learning, as shown by the examples of section 5.

4.2 Pitfalls of Fuzzy Logic

However, fuzzy systems present their own problems. Due to its knowledge-based nature, fuzzy logic requires a correct definition of input and output variables as well as of their relationships. If an expert of the field cannot be found, it will be hard to come up with the rules and there may need for a lot of tuning, resulting in increased development time.

As another drawback, the development of a fuzzy system for a game with many computer-controlled agents, if not carefully designed, may result in hundreds of rules to be checked at each time step, removing altogether the benefits of the low computational cost of the single checks. Although the simplest version of a fuzzy system appears in most games, improvements catered to game AI's nature have been suggested due to this drawback [3]. We mention as solutions single-state outputs to avoid unnecessary computations, hierarchical behaviors to resolve groups of rules at once and parallel and independent behavior layers with different evaluation frequencies. To further aid game developers in their implementation of a fast-performance fuzzy system, the Free Fuzzy Logic Library[4] (FFLL) has been created and its use is suggested in the literature [41].

One specific instance of the many-rules drawback in fuzzy logic is called combinatorial explosion [41]. In a typical videogame, there may be many input variables to an agent's behavior, each with a number of fuzzy sets. If we were aiming for completeness while creating the rules of the fuzzy system, we would end up with an exponential growth of rules in respect to fuzzy variables and sets. As an example, a fuzzy system with $n_V = 5$ input variables, each with $n_s = 5$ membership sets, would require $n_V^{n_s} = 3,125$ rules. This would make the fuzzy system too heavy for real-time computation. Game AI developers suggest to adopt the Combs method [8] that allows a linear growth of rules in regards to the number of variables and sets. Note, however, that the resulting system will be an approximation of the original one, giving slightly different results for the same input values. In addition, an existing fuzzy rule base cannot be easily transformed into a Combs notation, but instead the rules must be thought from scratch.

[4] http://ffll.sourceforge.net/

4.3 Fuzzy State Machines

Fuzzy logic often appears in games under the guise of fuzzy state machines (FuSMs), a technique that has been reported by several authors [10, 21, 28, 31]. The introduction of FuSMs has allowed game programmers to extend their method of choice, finite state machines, with the benefits of fuzzy logic. FSMs are known to explode in complexity as more states are added and it is not uncommon for many states to be needed for complex characters. According to [28], FuSMs allow the creation of subtle, less predicable behavior with fewer states and thus less complexity than basic FSMs. The result is that developers can use the well known state machine structure, with its simplicity of implementation and its structural power, paired with the unpredictability and dynamism that define fuzzy logic.

Existing FSMs can be easily transformed into FuSMs and this means that the development time does not suffer from the transformation, a benefit that game developers are all too pleased to encounter. There are two main methods for transforming a FSM into a FuSM. The first method employs fuzzy state transitions, transitions triggered by fuzzy reasoning, and is straightforward to implement. The second method employs fuzzy states, meaning that an agent may be in different states at the same time with a different degree of membership. In this last case, fuzzy transitions modify the membership degree for each state of the agent. This method can be useful to model the emotional state of the agent, who can be in a state of anger, sadness or happiness at different degrees [11]. A last very simple method to add fuzzy logic to a FSM, mentioned by [21], consists in adding a new state that uses fuzzy logic to decide what other state to move to, without altering the rest of the FSM.

5 Games using Fuzzy Logic

Reviewing the literature, we can find several mentions of the use of fuzzy logic in video-games. Due to their different context, we can separate the examples gathered from the industry from those found in the academic literature.

5.1 Fuzzy logic in the Videogame Industry

Given the fact that plenty of literature has been written on the use of fuzzy logic in games, it is surprising that little mention can be found about actual commercial games that specifically implement it. Especially, it is hard to find mention of it in more recent games. We think that fuzzy logic, due to the simple form that appears in video game AI and due to the high

interest in more exotic techniques, is sadly not deemed worthy anymore of fanfare when applied. Nonetheless, a few less recent examples can be found. A comprehensive list of games using interesting AI techniques can be found in [36].

Unreal[5] is one of the most famous first person shooter in videogame history and has been reported to use FuSMs to control the behavior of the enemy aliens. The game has been praised at release for its believable AI [37, 16].

BattleCruiser: 3000AD[6] is a space strategy game, with a controversial development history, that was supposed to use fuzzy logic alongside neural networks to control the non-player characters in the game [31].

S.W.A.T. 2[7], is a real-time tactics game that has been reported to make extensive use of fuzzy logic to enable the non-player characters to behave spontaneously based on their defined personalities and abilities [16, 31].

Civilization: Call to Power[8], a turn-based strategy game that is a spin-off of a very well known franchise, uses FuSMs to set priorities for strategic level AI, allowing personality traits to be defined for the different civilization leaders [16].

Close Combat[9] and its sequel Close Combat 2[10] use a FuSM that weights hundreds of variables through many formulas to determine the probability to take a particular action [31, 20].

Enemy Nations[11] features AI-controlled enemies that employ finite and fuzzy state systems and a database of goals and tasks [36].

The worldwide top-selling game The Sims[12] uses FuSMs to determine what objects a Sim character can interact with based on their properties and the Sim's personality traits, coupled with the *smart terrain* engine created by the famous designer Will Wright [20].

The Chronicles of Jaruu Tenk[13] employs A-Life technologies alongside heavy use of cascaded fuzzy state machines to provide behavior for its critters [36].

From the examples provided, we see that fuzzy logic in games has been used successfully into the industry, although developers tend to not go further than simple inference engines or FuSMs.

5.2 Research on Games and Fuzzy Logic

We instead find a different situation in the research field, where diverse applications of fuzzy logic and systems can be found, although with less frequency than the more popular techniques of evolutionary algorithms and neural networks. Fuzzy in game AI research is also often used alongside other advanced techniques.

Often, due to the impossibility to acquire the source code of commercial games, researchers try to apply their techniques to games created specifically for research or to clones of known games.

Just as in the industry, fuzzy logic has been used in research for modeling the behavior of computer-controlled agents and NPCs. Fuzzy logic is used for the design of the behavior of the enemy ghosts in a Pac-Man clone[14], although with heavy tuning needed for achieving a reasonable behavior [30]. Fuzzy Q-learning, borrowed from the fields of robotics, is used in a Ms. Pac-Man clone[15] [9]. Li et al. [39] mention fuzzy control as a practical method for generating subtle behavior and use it in a Belief-Desire-Intention (BDI) framework as part of decision making for a BattleCity[16] clone game. Ho et al. use a context-dependent fuzzy controller for maneuvering a car in the simulated environments of TORCS with empirically chosen fuzzy sets [14]. Perez et al. use evolutionary algorithms to evolve the parameters of the same fuzzy system from a starting guess [24]. Cardemone et al. employ an expert who is in charge of designing the rules of a similar fuzzy controller [7]. To this date, the tuned fuzzy controller still outperforms other controllers.

Of high interest are applications of fuzzy logic alongside or in place of the AI of the original commercial games. Since the AI is applied to a well known game, it is easier to find expert players to test it against. In addition, a powerful AI applied to a commercial success is more likely to influence the industry. This requires the game AI to be extensible, a feature that is found in only a handful of games. The code of Quake III Arena[17], a well known shooter, is available as open source and as such the game has been the target of much research. Fuzzy-logic controlled bots have been released for the game [34], with weapon and item selection controlled by fuzzy decision making. These controllers are taken as basis for comparisons when more advanced techniques are used [35, 27]. Pinto models fuzzy sensors as input to extended behavior networks in Unreal Tournament[18], another famous first person shooter [26], while Acampora uses timed automata and fuzzy controllers to model emotions for bots in a Unreal Tournament 2004[19] match [1].

Additional uses of fuzzy logic in games regard the classification of player feedback and learning from the player. In these contexts, fuzzy logic is usually chosen to model the player (or of a virtual opponent) due to its power as a method to model emotion. El-Nasr reports that fuzzy logic has provided a better means of modeling emotions due to its qualitative and quantitative expressiveness [11] . He applies fuzzy logic to define emotions for a virtual pet, then introduces reinforcement learning to adapt the pet's behavior to the specific player. Levillain et al. use fuzzy decision trees to categorize the emotional feedback of players during gameplay[17].

[5] Unreal, Epic Games, 1998
[6] Battlecruiser 3000AD, 3000AD Inc., 1995
[7] Police Quest: S.W.A.T. 2, Yosemite Entertainment, 1998
[8] Civilization: Call to Power, Activision 1999
[9] Close Combat, Atomic Games, 1996
[10] Close Combat: A Bridge Too Far, Atomic Games,1998
[11] Enemy Nations, Windward Studios, 1997
[12] The Sims, Maxis, 2000
[13] The Chronicles of Jaruu Tenk, Gee Whiz! Entertainment, 1999
[14] Pac-Man, Namco, 1980
[15] Ms. Pac-Man, Midway, 1982
[16] BattleCity, Namco, 1995
[17] Quake III Arena, Id Software, 1999
[18] Unreal Tournament, Epic, 1999
[19] Unreal Tournament 2004, Epic, 2003

Wong et al. use fuzzy control to manage the complexity of a scene, using fuzzy classification to determine the level-of-detail for 3D models [12]. Ohsone et al. use fuzzy decision trees to determine the best response of their virtual opponent [23].

Learning of membership sets, or more commonly of fuzzy rules, has also been explored. Due to their nature, fuzzy systems are good candidates for learning and are thus paired with different techniques to enable it. For example, fuzzy logic is used to model rules that are then evolved using the player's gameplay as input to the evolutionary algorithm [4]. In another work, the authors extract data from the iterative execution of games and then learn fuzzy rules for the classifications of player actions [15].

Figure 2: Games that make use of fuzzy logic. Top left: Call to Power. Top right: The Sims. Bottom left: Close Combat 2. Bottom right: S.W.A.T. 2

6 Conclusions

Fuzzy logic has been explored both in research and in the industry for use in videogames and, specifically, to model agent behaviors. It is nowadays a well-known technique, one that every AI game developer is required to master.

Fuzzy logic brings many benefits to the modeling of intelligent game agents. Its main benefit is the simplicity of its formulation, which paired with its input-output nature allows developers to integrate it into many games with ease, a huge benefit if we consider the tight schedules of game developers.

Even in its simplicity, fuzzy logic can be a powerful AI technique, especially due to the possibilities to model nonlinearities, to achieve complex behavior with simple rules, to be a candidate for learning and to mimic human reasoning and emotions.

Games that introduced fuzzy logic have been for the most part commercial successes, with The Sims still being the best selling PC game in history (16 million copies sold[20]). It is safe to assume that fuzzy logic had some role in this success.

We think that the potential of the technique, however, has not been fully expressed, especially in commercial games, where even today it is common to find seemingly inhuman and outright unintelligent computer-controlled agent. Fuzzy logic would be a simple way to elevate those agents to a higher standard with little effort.

In order for this to happen, developers should dare to separate for a moment from the simplest AI techniques, gaining huge benefits on the long run.

References

[1] Giovanni Acampora. Synthesizing bots emotional behaviors through fuzzy cognitive processes. In *Proceedings of the 6th international conference on Computational Intelligence and Games*, CIG'10, 2010.

[2] Bob Alexander. The beauty of response curves. *AI Game Programming Wisdom*, 2002.

[3] Thor Alexander. An optimized fuzzy logic architecture for decision-making. *AI Game Programming Wisdom*, 2002.

[4] Philippa Avery and Zbigniew Michalewicz. Adapting to human game play. In *Proceedings of the 4th international conference on Computational Intelligence and Games*, CIG'08, 2008.

[5] David M. Bourg and Glenn Seemann. *AI for Game Developers*. O'Reilly Media, 2004.

[6] Mat Buckland. *Programming Game AI by Example*. Jones & Bartlett Publishers, 1 edition, September 2004.

[7] L. Cardamone, D. Loiacono, and P. L. Lanzi. Overtaking opponents with blocking strategies using fuzzy logic. In *Proceedings of the 6th international conference on Computational Intelligence and Games*, CIG'10, 2010.

[8] William E. Combs. *The Fuzzy Systems Handbook 2nd Ed, Academic*. 1999.

[9] Lori L. DeLooze and Wesley R. Viner. Fuzzy q-learning in a nondeterministic environment: developing an intelligent ms. pac-man agent. In *Proceedings of the 5th international conference on Computational Intelligence and Games*, CIG'09, pages 162–169, Piscataway, NJ, USA, 2009. IEEE Press.

[10] Eric Dybsand. A generic fuzzy state machine in c++. *Game Programming Gems 2*, 2001.

[11] Magy Seif El-Nasr, John Yen, and Thomas R. Ioerger. Flamefuzzy logic adaptive model of emotions. *Autonomous Agents and Multi-Agent Systems*, 3(3):219–257, September 2000.

[20] http://callcenterinfo.tmcnet.com/news/2005/feb/1114806.htm

[12] Jiljang Wang Gabriyel Wong. A fuzzy-control approach to managing scene complexity. *Game Programming Gems 6*, 2006.

[13] E. Hastings, R. Guha, and K. Stanley. Automatic content generation in the galactic arms race video game. In *Proceedings of the 5th international conference on Computational Intelligence and Games*, CIG'09, 2009.

[14] Duc Thang Ho and Jonathan M. Garibaldi. A fuzzy approach for the 2007 cig simulated car racing competition. In *Proceedings of the 4th international conference on Computational Intelligence and Games*, CIG'08, 2008.

[15] Hisao Ishibuchi, Ryoji Sakamoto, and Tomoharu Nakashima. Learning fuzzy rules from iterative execution of games. *Fuzzy Sets and Systems*, 2003.

[16] Daniel Johnson and Janet Wiles. Computer games with intelligence. In *In Procs. 10th IEEE Intl Conf. on Fuzzy Systems*, pages 61–68. IEEE, 2001.

[17] F. Levillain, J. Orero, and M. Rifqi. Characterizing players experience from physiological signals using fuzzy decision trees. In *Proceedings of the 6th international conference on Computational Intelligence and Games*, CIG'10, 2010.

[18] Daniele Loiacono, Julian Togelius, Pier Luca Lanzi, Leonard Kinnaird-heether, Simon M. Lucas, Matt Simmerson, Diego Perez, Robert G. Reynolds, and Yago Saez. The wcci 2008 simulated car racing competition, 2008.

[19] Mason McCuskey. Fuzzy logic for video games. *Game Programming Gems 1*, 2000.

[20] Kathryn E. Merrick and Mary Lou Maher. *Motivated Reinforcement Learning: Curious Characters for Multiuser Games*. Springer, 2009.

[21] Ian Millington. *Artificial Intelligence for Games (The Morgan Kaufmann Series in Interactive 3D Technology)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2006.

[22] Larry O'Brien. Fuzzy logic in games. *Game Developer Magazine*, 1996.

[23] K. Ohsone and T. Onisawa. Friendly partner system of poker game with facial expressions. In *Proceedings of the 4th international conference on Computational Intelligence and Games*, CIG'08, 2008.

[24] Diego Perez, Gustavo Recio, Yago Saez, and Pedro Isasi. Evolving a fuzzy controller for a car racing competition. In *Proceedings of the 5th international conference on Computational Intelligence and Games*, CIG'09, pages 263–270, Piscataway, NJ, USA, 2009. IEEE Press.

[25] Hugo Pinto and Luis Otavio Alvares. Behavior-baed robotic architectures for games. *Game Programming Gems 6*, 2006.

[26] Hugo Pinto and Luis Otavio Alvares. Costructing a goal-oriented robot for unreal tournament using fuzzy sensors, finite-state machines, and extended behavior networks. *Game Programming Gems 6*, 2006.

[27] Armand Prieditis and Mukesh Dalal. Applying model-based decision-making methods to games: Applying the locus ai engine to quake iii. *Game Programming Gems 6*, 2006.

[28] Sbastien Schertenleib. Designing a multilayer, pluggable ai engine. *Game Programming Gems 6*, 2006.

[29] Bob Scott. The illusion of intelligence. *AI Game Programming Wisdom*, 2002.

[30] Adnan Shaout, Brady W. King, and Luke A. Reisner. Real-time game design of pac-man using fuzzy logic. *Int. Arab J. Inf. Technol.*, 3(4):315–325, 2006.

[31] Penelope Sweetser and Janet Wiles. Current ai in games: a review. *Australian Journal of Intelligent Information Processing Systems*, 8(1), 2002.

[32] Paul Tozour. Evolution of game ai. *AI Game Programming Wisdom*, 2002.

[33] Alan Turing. Computing machinery and intelligence. *Mind LIX*, 1950.

[34] J. M. P. van Waveren. The quake iii arena bot. Master's thesis, University of Technology Delft, 2001.

[35] Joost Westra and Frank Dignum. Evolutionary neural networks for non-player characters in quake iii. In *Proceedings of the 5th international conference on Computational Intelligence and Games*, CIG'09, pages 302–309, Piscataway, NJ, USA, 2009. IEEE Press.

[36] Steven Woodcock. Games making interesting use of artificial intelligence techniques. *Game AI*, 1995-2000.

[37] Steven Woodcock. Game ai: the state of the industry. 1999.

[38] Steven Woodcock. Game ai: the state of the industry. 2000.

[39] Petr Musilek Yifan Li and Loren Wyard-Scott. Fuzzy logic in agent-based game design. *Fuzzy Information, 2004. Processing NAFIPS '04. IEEE Annual Meeting of the*, 2004.

[40] Lofti A. Zadeh. Fuzzy sets and systems. *System Theory*, 1965.

[41] Michael Zarozinski. Imploding combinatorial explosion in a fuzzy system. *Game Programming Gems 2*, 2001.

[42] Michael Zarozinski. An open source fuzzy library. *AI Game Programming Wisdom*, 2002.