Believable and Effective AI Agents in Virtual Worlds: Current State and Future Perspectives

Iskander Umarov
TruSoft Int'l Inc.
204 37th Ave. #133, St. Petersburg, FL 33704, USA
umarov@trusoft.com

Maxim Mozgovoy[*]
University of Aizu
Aizu-Wakamatsu, Fukushima-Ken, 965-8580 JAPAN
mozgovoy@u-aizu.ac.jp

P. Clint Rogers
University of Eastern Finland,
P.O. Box 111, FI-80101 Joensuu, FINLAND
clint.rogers2008@gmail.com

ABSTRACT
The rapid development of complex virtual worlds (most notably, in 3D computer and video games) introduces new challenges for the creation of virtual agents, controlled by artificial intelligence (AI) systems. Two important subproblems in this topic area which need to be addressed are (a) believability and (b) effectiveness of agents' behavior, i.e. human-likeness of the characters and high ability to achieving their own goals. In this paper, we study current approaches to believability and effectiveness of AI behavior in virtual worlds. We examine the concepts of believability and effectiveness, and analyze several successful attempts to address these challenges. In conclusion, we suggest that believable and effective behavior can be achieved through learning behavioral patterns from observation with subsequent automatic selection of winning acting strategies.

Keywords*:* AI for games, learning by observation, believability, behavior capture.

INTRODUCTION

What is a Virtual World?
Virtual worlds provide a basis for many popular video and computer games (e.g. Half-Life, The Sims), "life simulators" (e.g. Second Life, Entropia Universe), and virtual military training grounds (e.g. Virtual Battle Space). The use of such worlds becomes more and more widespread, as typical hardware is now able to perform realistic 3D real-time rendering, and the coverage of broadband Internet connection constantly increases. Furthermore, virtual worlds get new applications, most notably in the area of education and "serious games". One can note, for instance, the existence of "virtual classrooms" in Second Life.

[*] The corresponding author; supported in part by the Fukushima Pref. Foundation, Project F-23-1, FY2011.

It is interesting to note that while the term "virtual world" is commonly used, only few authors provide its precise definition. Bell (2008) tried to combine previous definitions, found in the literature, into the following formula: "a synchronous, persistent network of people, represented as avatars, facilitated by networked computers". This definition, though, ignores an important observation: modern virtual worlds can be inhabited not only with human-controlled characters, but also with AI-based virtual agents, serving as non-participating world's "native population", hostile opponents or friendly teammates. Furthermore, it is still unclear what kinds of computer-generated environments qualify for the name of "virtual worlds". The work by Mitchell (1995) "From MUDs To Virtual Worlds" suggests a clear distinction between MUDs (Multi-User Dungeons, text-based multiplayer games) and virtual worlds. Mitchell notes the lack of immersion, caused by limited visual capabilities of MUDs, and suggests that virtual worlds should combine MUD-styled gameplay with vivid 3D graphics. Though the discussion of features that turn a virtual simulation into a virtual world are beyond the scope of this paper, it is natural to presume that "a world" should possess certain high degree of complexity and audiovisual interaction, not found in simpler computer-simulated environments.

Virtual World's AI-controlled Inhabitants

As noted by Bell, the primary population of a virtual world is constituted by real people, represented as human-controlled agents or *avatars*. A world can be also inhabited by computer-controlled agents, also known as *NPCs* (non-player characters). Their role depends on the nature of the given world: serving as elements of the world's setting (Milward, 2009), being used for research purposes (Friedman, Steed, & Slater, 2007) or even being completely prohibited as violating the intended world's configuration (Blizzard, 2010).

The case of our interest is AI systems that can substitute human players or even *pretend* to be humans. This scenario is very common in the domain of computer games: if you play a computer chess match, fighting game, or a soccer tournament, you naturally assume that it is possible to choose either human- or computer-controlled opponent. Complex computer games and virtual worlds demand high-quality AI systems, able to control characters satisfactorily. The success of such AI is determined by the (human) users of a virtual world or a computer game. The factors that contribute to the overall user satisfaction are not obvious: successful AI systems are not necessarily the strongest game characters.

Consider the simplest example: in the classic Pac-Man game a human player has to eat all the "pills" located on the level, avoiding computer-controlled ghosts. It is not hard to program "optimal ghosts" that easily capture the protagonist; however, a player will be doomed to defeat in such a game, making the whole game project unejoyable and thus unsuccessful. The same problem arises in chess, since the best AI systems are able to play at grandmaster level, while not all human participants are eager to compete with grandmasters all the time.

With games being the way in which many people around the world are learning how to use computers, and one of the primary reasons people spend time on computers, questions regarding how to improve the quality of the virtual worlds (and the AI agents in them) is an interesting challenge and opportunity for our field. This paper is focused specifically on the analysis of the current state of affairs and possible further directions in AI agents' development for virtual worlds.

This paper is organized as follows. We begin with an overview of the research regarding what makes games fun and engaging, or "fun factors", then looking specifically at how AI Agents in Virtual worlds can contribute to or distract from these factors, and the overall success of that game. We then examine what it is that makes AI agents "believable" and "effective", and address what are the practical challenges that exist in AI development of these characteristics? We outline some of the potentially promising experimental approaches to AI development (including what testbeds can be used, how to test believability, what research projects have contributed to creating believable behavior, and what research projects have contributed to creating effectiveness). Using the background of the existing research, we then look at a specific case study of the same: a 3D boxing video game environment. Through this case study, we experiment with and research the practical application of creating AI agents that: (a) increase "believable" behavior through learning by observation and case-based reasoning; (b) optimize "effectiveness" behavior by means of reinforcement learning; and thus (c) outperform built-in handcrafted AI agents in evaluation with Turing tests and automated schemes (Mozgovoy & Umarov, 2010a,b). We conclude with discussion, conclusions, and some suggestions for future research.

FUN FACTORS AND REALISM

The Pac-Man and chess examples show that the quality of artificial agent's decision making, in terms of achieving own goals in the game, is not the only criterion to measure the quality of a computer-controlled opponent. A game should be *enjoyable*, and virtual training environment has to be *realistic*, and AI features should be consistent with these goals. As Paul Tozour (AI Programmer for Deus Ex 2 game) notes, "The whole point is to entertain the audience, so no matter what you do, you need to make sure the AI makes the game more fun. If a game's AI doesn't make the game a better experience, any notions of 'intelligence' are irrelevant" (Eyewitness: Complete AI Interviews. 2002).

General Game Fun Factors

While the concept of *realism* leaves little ambiguity[1], the factors that contribute to the overall game enjoyability are not easy to reveal. These factors (often called "fun factors") serve as a subject of research activities for decades. Perhaps, the earliest attempts to analyze them are found in Malone (1981). Malone lists the following factors as the most important: (a) the existence of clear goals; (b) the presence of hi-score table; (c) good audiovisual effects; (d) the presence of randomness in the gameplay.

More recent studies have shown the differences in perceived importance of particular fun factors between game developers and players, and across game genres (Choi, Kim, & Kim, 1999). Furthermore, the conclusions often depend on the used research basis and methods. Many researchers, for example, rely on the Flow Theory (Csikszentmihalyi, 1991). As shown

[1] According to Encyclopedia Britannica, realism is "the accurate, detailed, unembellished depiction of nature or of contemporary life".

by Csikszentmihalyi, most Flow (i.e. deep enjoyment of a certain activity) experiences are caused by goal-directed, rule-bounded activities that require appropriate skills and mental energy. These observations were applied to the domain of game worlds by Sweetser and Wyeth (2005), who list the following factors, affecting user enjoyment:

1. Concentration: games should require concentration, and the player should be able to concentrate on the game.
2. Challenge: games should be sufficiently challenging and match the player's skill level.
3. Player skills: games must support player skill development and mastery.
4. Control: players should feel a sense of control over their actions in the game.
5. Clear goals: games should provide the player with clear goals at appropriate times.
6. Feedback: players must receive appropriate feedback at appropriate times.
7. Immersion: players should experience deep but effortless involvement in the game.
8. Social interaction: games should support and create opportunities for social interaction.

However, systematic study of game reviews performed by Wang, Shen, and Ritterfeld (2009) shows that professional reviewers mention other game attributes as key fun factors:

1. Technological capacity (various technological aspects of the game).
2. Game design.
3. Aesthetic presentation (audiovisual experience and game style).
4. Entertainment gameplay experience.
5. Narrativity / storyline.

Certain fun factors can be identified even for very simple games, such as Pac-Man: (a) the game should be neither too hard nor too easy; (b) there should be a diversity in ghosts' behavior over a series of games; (c) the ghosts' behavior should be aggressive rather that static (Yannakakis & Hallam, 2004).

AI as Fun Factor

So how does this relate to AI agents in virtual worlds? The above lists of fun factors may not provide direct guidance on the role of computer-controlled characters in overall success of a computer game or virtual world, but can help frame the discussion. Additionally, it is clear that the importance of high-quality AI varies from genre to genre: in computer chess, AI is the most valuable game subsystem; in Tanki Online[2] AI-controlled characters are absent, so game enjoyability is completely defined by other factors.

If we presume that a certain virtual world is inhabited with AI-controlled characters, it is important to examine the impact of AI quality on the overall user experience. Technically we can connect AI with the factors 1 to 6 in Sweetser and Wyeth's list, and with the factors 1, 2, & 4 in the list composed by Wang et al. Indeed, Sweetser and Wyeth mention AI as an important source of game challenge. Explicit views on this issue are expressed in the works

[2] Available at http://tankionline.com/en/game/.

by Yannakakis and Hallam (2004, 2005). The authors show how AI quality affects player's enjoyment in the games that involve interaction with computer-controlled opponents. They conclude that more enjoyment comes with non-trivial adaptive opponents, which means they exhibit a variety of different behavioral styles and are able to respond to changing human strategies.

What other features, besides adaptivity, help make AI an enjoyable opponent? Numerous researchers, e.g. (Choi, Konik, Nejati, Park, & Langley, 2007; Glende, 2004; Taatgen, van Oploo, Braaksma, & Niemantsverdriet, 2003), mention *human-likeness* of computer-controlled opponents as one of the primary attributes of a successful AI system. To quote Taatgen et al., "a computer opponent becomes more interesting and enjoyable to play against as it behaves more like a person"[3]. In case of virtual worlds, inhabited with human-like creatures, the human-likeness of their behavior (fulfilled by an AI system) is essential, since it contributes to the overall realism of the world. Players may expect human-like behavior from human-like characters even in simpler game environments, not necessarily qualifying for a status of full-fledged virtual world. But what does "human-like" mean exactly?

Since *game challenge* is a widely recognized fun factor, we can identify the following attributes of a successful AI system: realism / human-likeness, and high skill level. In the subsequent sections, we will have a more detailed look at these attributes and examine, what is it that makes AI agents "believable" and "effective".

BELIEVABLE AND EFFECTIVE AI AGENTS

Realism and human-likeness of AI-controlled agents are related to the general principle of *believability*. Following the concept of "believable character" in the Arts, the term "believable agent" is used in Computer Science literature to denote realistic interactive characters, inhabiting virtual worlds. Defined in (Bates, 1994) as the "one that provides the illusion of life, and thus permits the audience's suspension of disbelief", a believable agent is usually characterized with human-like peculiarities, such as capabilities to learn, to "think" for a while between the decisions, to make mistakes, to adjust own strategy in response to opponent's actions. Furthermore, a believable agent should not be given "god abilities", such as seeing hidden objects, having extra-fast reaction or controlling several army squads simultaneously.

Identifying individual AI features, leading to believability, is a worthwhile research topic. Jones et al. (1999) propose the following basic principles for building a believable agent: (a) it should have roughly the same basic data processing structure as human players; (b) it should have the same basic sensory and motor systems as human players; (c) it should have the same knowledge as human players.

[3] However, Yannakakis and Hallam (2007) note the lack of evidence that human-like behavior should generate more enjoyment; instead, they connect user satisfaction with individual AI features, not necessarily ensuring human-likeness: high skill level, diversity and adaptivity of behavior. Although, a more recent study by Soni and Hingston (2008) shows that AI's human-likeness is indeed a notable fun factor.

The acceptance of these general principles still leaves enough room for their interpretation and implementation in actual virtual worlds. For example, the work by Choi et al. (2007) adds three more features for a human-like agent: (a) hierarchical organization of knowledge; (b) goal-directed, but reactive behavior; (c) incremental learning schemes.

The developers of computer games are well-aware of the positive impact which believability of agents makes on the user's gaming experience. For example, the authors of the official Counter-Strike bot specifically mention human-like AI features, such as "behave in a believably human manner", "communicate with teammates", respect "player perception of a 'fair fight'", and "create behavior variation among bots" in terms of aggression, skill, teamwork, reaction times, and morale (Booth, 2004). The same work mentions non-technical issues, but important from the standpoint of user perception AI features, such as treating a human player as the head (bots refer to human players "Sir" or "Commander") and congratulating players occasionally by sending a message "nice shot, sir" to a human player on a successful attack.

The same believability-increasing trend is noted by Orkin (2006): "in the early generations of shooters, such as *Shogo* (1998) players were happy if the A.I. noticed them at all and started attacking. … Today, players expect more realism, to complement the realism of the physics and lighting in the environments."

The contribution of such human-like AI-controlled characters is likely to be especially valuable in case of believable cooperating opponents, as noted by Yannakakis and Hallam (2007). So we can analyze not only individual believable agents, but also whole teams of believable agents, which are a topic of special interest in the case of first-person shooters and military training virtual 3D worlds.

However, believability is not the only feature that makes AI-controlled characters fun to play with. A game should be challenging, so AI should possess reasonable skills or, in other words, to be *effective*.[4] Effectiveness of decision-making is a classical aim for AI design: developing virtual characters that can compete with human opponents is a demanding task for high variety of computer games. As already stated, the goals of believability and effectiveness are not always synonymous: a highly skillful agent is not always believable; a believable character can be a weak opponent.

A good example of a virtual environment where successful AI is completely identified with effective AI is RoboCup (Kitano et al., 1998). Within this project, the participants create competing teams of virtual soccer players. A winning team should exhibit effective behavior, which does not have to be similar to strategies of human players in a real soccer match. So the question now is, what practical challenges exist in AI development of these characteristics of believability and effectiveness?

[4] Effectiveness is important not only for AI-controlled opponents. Human players might expect reasonably skillful behavior also from their teammates and neutral characters.

PRACTICAL CHALLENGES IN AI DEVELOPMENT

From the technical point of view, the approaches used to build virtual characters in practice are generally conservative. For example, Counter-Strike bots rely on preprogrammed rule-based scenarios (Booth, 2004), and Halo 2 AI is built on top of a hierarchical finite state machine (Isla, 2005). More advanced AI techniques, such as neural networks are primarily used when behavior learning is a significant part of gameplay, as in the case of Black & White and Creatures (Champandard, 2007).

During the last years, game AI technologies experienced significant evolutionary development. Hierarchical finite state machines have evolved into behavior trees (Knafla, 2011), successfully used in such games as Crysis (Pillosu, 2009) and Spore (Hecker, 2011). Existing technologies have matured, industry has developed specialized tools for AI authoring and debugging (Champandard, 2011); although, most game AI systems are based on handcrafted behavioral scripts.

The main problem with manual design of AI behavior lies in inability to deal with growing complexity efficiently. The more complex agent's behavior becomes, the harder it is to create the corresponding data structures and algorithms. The solutions tend to be difficult for scalability, and the consistency of rules is difficult to maintain. Some of these complexity issues can be overcome by following clear and consistent design principles (Isla, 2005)[5]. However, these recommendations seem to be tactical rather that strategic. Truly complex issues such as above-mentioned real-time adaptive learning or complex team-based behavior remain a serious challenge for well-established game AI methods in the game development industry.

A well-recognized alternative is to employ some variation of machine learning to create AI-controlled characters and to adjust their behavior patterns. This approach is not popular in today's commercial games due to various reasons, including (a) hard predictability of obtained behavior, which makes AI harder to control by game designers; (b) significant time that may be required to train an effective AI agent; (c) possible unreliability of trained AI, as it can get stuck in certain cases (Sánchez-Crespo Dalmau, 2005). Furthermore, machine learning often does not optimize AI in terms of skill level, so the developers are afraid that agents might learn inferior behavioral patterns (Spronck, Ponsen, Sprinkhuizen-Kuyper, & Postma, 2006).

In comparison to research projects, commercial video and computer games are usually developed under much stricter bounds. The code has to be highly efficient, time and memory constraints are challenging, and the development should be finished within a given time frame, under a specified budget. Thus, game developers do not always have a chance to experiment with cutting-edge AI research. Furthermore, AI agents in commercial games have to be robust and reliable, which normally makes the use of research technologies intolerably risky. Though, according to observations made by Champandard (2011), recent attempts to bridge the gap between academia and industry in the field of AI research are very promising.

[5] The author lists the following features of a reasonable AI architecture: customizability, simple and explicit ways to add hacks into the code, the capability to build new features on the basis of existing features.

In this next section we address some of the potentially promising experimental approaches to AI development.

EXPERIMENTAL APPROACHES TO AI DEVELOPMENT

The development of believable and effective AI-controlled characters is a subject of numerous research projects. Many of them deal with isolated AI attributes, and the obtained systems are not always generalizable to full-scaled virtual worlds; however, these experiments provide greater diversity in approaches to development of believable agents, and usually implement more advanced features, not yet available in released game titles. It should be mentioned that several recent projects are devoted to comprehensive AI research in currently available virtual worlds, thus bridging "academia-industry" gap, as Champandard (2011) noted. However, this needs to be balanced with the reality that experimental methods are still often not mature enough to form a solid ground for a commercial game's AI, as their robustness and reliability are not always guaranteed.

Testbeds

In order to develop AI-controlled characters inhabiting a virtual world, one first needs a virtual world. While certain researchers work with specialized worlds, designed for the experiment, many projects rely on existing virtual environments. By utilizing ready-made engines, researchers can concentrate on pure AI development, and prevent possible criticism of being a "toy example", not scalable for the needs of real systems. Naturally, the popularity of a certain testbed is determined by a variety of both technical and non-technical factors, such as complexity of the environment (and thus the possibility to test complex behavior patterns), ease of interface and programming, quality of documentation, availability of the virtual world (as an open source or for a reasonable price), general widespreadness of the environment and the presence of a community around it.

A good example of a testbed popular in the past is Quake 2. It was used in early experiments to experiment with learning schemes of human-like behavior (Thurau, Bauckhage, & Sagerer, 2004) and to create believable bots (Laird, 2002). Quake 2 implements a special interface that allows third-party developers to access internal data structures and thus program engine-agent interaction. However, this environment was characterized as being too restricted for studying complex behavior (Laird, 2002). So the world of Quake 2 allows obtaining relevant results, but its limits are already reached with the current AI technologies.

Naturally, particular types of virtual environments are best suitable for different kinds of AI research. While military-style first-person shooters are best for developing arcade behavioral elements (shooting and dodging) and spatial reasoning, role-playing environments are best for computer-simulated social interactions, and real-time strategies are popular for creating team behavior and long-term goal-based planning modules. The most used testbeds of the present day include (but not limited to):

- **Quake 3 Arena**. Referenced in (Choi et al., 2007) as a testbed for a believable agent, built with the help of general-purpose cognitive architecture called ICARUS. Also used in experiments with neural network-based AI by Westra and Dignum (2009). Since the engine's source code is freely available now under GNU license, some researchers implement new algorithms directly in the code of existing game bots (El Rhalibi & Merabti, 2008).

- **Unreal Tournament.** Used as a testbed for creating human-like bots (Hirono & Thawonmas, 2009), to learn winning AI policies (Smith, Lee-Urban, & Munoz-Avila, 2007), and as a user-end system for a distributed interactive military simulator (Manojlovich, Prasithsangaree, Hughes, Chen, & Lewis, 2003). Unreal Tournament also lies in the foundation of a well-known America's Army recruiting simulation project (Zyda et al., 2003). This engine is known for an extensive software development kit (shipped with the Unreal Tournament game) and a built-in programming language UnrealScript, allowing to design own virtual worlds (Laird, 2002). Unreal engine's extension Gamebots (Adobbati et al., 2001) is specifically developed for multi-agent research in 3D worlds. Currently Gamebots serves as a framework for several AI research projects (Hingston, 2009; Lee & Gamard, 2003; Robertson & Good, 2005). Unreal engine is free for noncommercial use.
- **Second Life**. Since Second Life is an online virtual world, mostly inhabited by real humans, AI-controlled characters are often used to analyze human population, and to establish contacts with humans. For example, research bot by Friedman et al. (2009) collect age data of avatars; Ijaz et al. (2011) develop a conversational agent with improved abilities to understand its environment. Designing AI with advanced spatial reasoning is still a new topic for Second Life, but there are research projects aimed at integration of existing cognitive architectures into Second Life's virtual world (Ranathunga et al., 2010; Ranathunga et al., 2011). AI-controlled characters in Second Life are programmed with built-in Linden Scripting Language. Basic Second Life accounts are free; additional capabilities are available for subscribed users.
- **Open Real-Time Strategy (ORTS) and Stratagus**. ORTS and Stratagus are open source engines, suitable for studying AI problems, applied to the domain of real-time strategy games. Real-time strategies are especially relevant for analyzing agent coordination (Lichocki, Krawiec, & Jaśkowski, 2009; van der Heijden, Bakkes, & Spronck, 2008) and strategic planning algorithms (Balla & Fern, 2009). Other directions include applying general AI methods (Hagelbäck & Johansson, 2009) and existing cognitive architectures (Wintermute, Xu, & Irizarry, 2007) to the domain of real-time strategies. Furthermore, real-time strategies serve as appropriate testbeds for developing and testing utility AI algorithms, e.g., pathfinding (Naveed, Kitchin, & Crampton, 2010).
- **StarCraft**. A popular game StarCraft became a notable testbed for evaluation of AI agents thanks to StarCraft AI Competition, first hosted by AIIDE Conference in 2010[6]. Participating AI systems are engaged in full-scaled StarCraft matches. The competition is, however, not aimed at creation of believable or fun characters; in this point it thus can be compared to RoboCup.

[6] http://eis.ucsc.edu/StarCraftAICompetition

Believability Criteria

As discussed above, believability is an important property of AI-controlled characters. Computer-generated behavior should be reasonably "human-like" to provide feeling of realism, which contributes to general enjoyability of a game. In case of simulation and training applications (e.g., military training games), realism of the constructed virtual environment is one of the primary explicit design goals of simulation software.

Turing Test

Evaluating believability is not an easy task. One of the possible approaches to the problem is to adapt a well-known Turing test (Turing, 1950) to intelligent agents, behaving in a virtual world. Hingston (2009) describes this modified test as follows. A human player (judge) is engaged in a game against two opponents, taking place in a virtual world. One of these opponents is controlled by another human (confederate), while another is controlled by an AI system. The task of the judge is to identify the human among the opponents. The task of the AI system is to pretend to be human thus deceiving the judge. Hingston notes the following differences from classic Turing test:

- All participants are independently playing against each other.
- The confederate simply tries to reach own goal (to win the game), and does not assist/impede the judge.
- No natural language-related difficulties are involved.
- Human players and bots are supplied with different kinds of information: vision and sound for humans, data and events for bots.

A simplified version of this test suggests that the judge has to watch a game between two players, both of whose can be controllable by a human or an AI system. The task of the judge is to identify game participants. Such a test can be relatively easily passed by a conventional AI agent in case of simple games, due to the overall simplicity of gaming process. For example, a believable agent for Pong[7] is discussed in (Livingstone, 2006).

Complex game worlds demand more from AI-controlled players. Glende (2004) lists the following features that agents should possess in order to pass Turing test:

- a reasonable combination of predictability and unpredictability in agent's decisions,
- creativity in problem-solving,
- clear personality,
- internal intensions and autonomy in acting,
- the capability to plan and improvise,
- the ability to learn.

An experiment that involves evaluation of Quake 2 agents has been performed by Gorman, Thurau, Bauckhage, and Humphrys (2006). The idea of the study was to show a series of Quake 2 video clips (as seen by the player's first-person camera) to a number of people and

[7] Pong is the simplest variation of 2D tennis, where each player controls a rectangular bat, moveable in vertical direction.

to ask, whether they do believe that the active player is a human. In different clips, the game character was controlled by: (a) a real human player; (b) a popular Quake rule-based agent; (c) a specifically designed “imitation agent” that tried to reproduce human behavior using Bayesian motion modeling.

The results were positive for the imitation agent: it was misidentified as human in 69% of cases, while the conventional AI was mistaken as human only 36% of the time (the average experience level of evaluators was between “played first-person shooters monthly” and “played first-person shooters weekly”). Sample evaluators’ comments, quoted in (Gorman et al., 2006), indicate that quite simple clues were used to guess human players (“fires gun for no reason, so must be human”, “stand and wait, AI wouldn’t do this”, “unnecessary jumping”).

One should note that Quake 2 world is relatively uncomplicated (Laird, 2002). So we can expect that a more complex world would require much stronger AI capabilities to pass Turing test. Indeed, an open “believability competition” 2K BotPrize among Unreal Tournament bots, hosted by IEEE CIG Symposium in 2008, declared no winner. None of five competing systems was unable to deceive human judges[8] (Hingston, 2009). Moreover, all five participating human players were ranked by the judges as “more human” than any of the bots. Still, after analyzing judges’ comments, Hingston concludes with an optimistic prognosis: “the task is just a little beyond the current state of the art: neither so easy that we need not stretch ourselves to succeed, nor so hard that success cannot be imagined”.

Automated Believability Tests

Direct Turing test involving human judges is not the only way to measure agents’ believability. An alternative approach suggests employing automated analysis algorithms: one can store action sequences, performed by human-controlled and AI-controlled characters in certain data structures, and analyze them for “believability attributes”.

While this idea is certainly appealing, as it implies the use of objective number-based evaluation algorithms, automated analysis has notable drawbacks, contributing to higher popularity of Turing test-based schemes. First, game / virtual worlds are different, and thus may require designing distinct evaluation algorithms. Second, one can argue that automated tests may serve as indirect proofs of presence or absence of believable behavior, but the final decision can be made only by human judges.

Nevertheless, automated tests are used in some research projects, at least, as additional support for the claims of believability. Tencé and Buche (2008) describe a method to measure similarities between human and AI behavior. First, an experimenter has to specify the most valuable attributes found in the complete description of a player’s individual action. Next, the system encodes player’s action sequences as vectors of numeric data. The vectors that correspond to different players are further compared using dot product: similar behavioral patterns should yield dot product close to one. The authors evaluated Unreal Tournament’s agents using two different kinds of simplified behavioral vectors: (1) vector of frequencies of

[8] The competition involved five judges; four of them were experienced computer game players.

velocity direction angle changes between two consecutive observation points; (2) vector of frequencies of angles between player direction and velocity direction at given consecutive observation points (new observation point arrives every 125 milliseconds). The experiments showed a clear separation between the groups of human players and AI agents for both versions of behavioral vectors: each human-generated vector was closer to any other human-generated vector than to any of AI-generated vectors.

Kemmerling et al. (2009) used automated believability measure to improve the believability of existing AI-controlled player. Through a number of surveys, they identified sets of human-like and non-human-like actions for the strategy game Diplomacy. A "believability calculator" that measures the believability of given action sequence was set up. Then an additional functionality was added to the AI system to encourage human-like and discourage non-human-like actions. However, this method can be characterized as indirect: while the calculator shows the share of believable actions in AI behavior, it does not evaluate the overall believability of bot's playing style.

Riedl and Young (2005) proposed a semi-automatic believability evaluation procedure for a completely different domain of computer-aided story generation, where the task is to measure believability of characters that participate in the generated story. The evaluation procedure analyzes a tree-like QUEST model (Graesser, Lang, & Roberts, 1991) of the given story that incorporates story events and character goals as nodes, connected with links labeled with relationship types (five types are supported). Then specialized algorithms are used to generate random pairs of questions and answers, related to the story, and to rate the quality (coherence) of answers. Human experts are also asked to provide their rating of answers' quality. The idea is that for a well-structured, believable story human-supplied answer ratings should be close to machine-made guesses.

An interesting application of automated believability testing is discussed in the paper by Pao, Chen, and Chang (2010). The authors develop an algorithm that compares trajectories of Quake 2 characters with pre-recorded trajectories of human players and bots. The goal is to classify any given player as a human or an AI system, thus revealing game bots. Since game bots are prohibited in a number of online games (their use is treated as cheating), and their revelation requires efforts, the proposed automated procedure may have a commercial value.

Creating Believable Behavior

Most research projects follow a natural way of constructing human-like believable behavior by analyzing actual human behavior patterns and subsequently implementing them in AI system. Even hand-coded algorithms can implement specific behavioral patterns, considered as "human-like" by their creators (Booth, 2004). However, a greater interest is evoked by the methods that can automatically update/fill-out agents' knowledge by observing behavior of human players. Below we briefly introduce several projects devoted to believable AI creation (with the intention to illustrate typical ideas of how to obtain believable behavior, rather than to provide a thorough survey; furthermore, we do not consider a related but separate task of agent coordination, leading to believable teams of AI-controlled characters).

Choi et al. (2007) apply a general-purpose cognitive architecture ICARUS (Langley, Choi, & Rogers, 2005) to train an agent for Urban Combat — a modification of Quake 3 Arena game. ICARUS is specifically designed to assist reactive execution in physical environments and implements the following principles: (1) primacy of action and perception over cognition; (2) separation of categories from skills; (3) hierarchical structure of long-term memory; (4) correspondence between long-term and short-term structures (ICARUS, 2007). Initially the agent is given hand-coded basic knowledge about the objectives and acting capabilities. Then the automated learning algorithm is executed every time whenever the agent has achieved a certain goal. Then this algorithm generates new behavioral skills that help to achieve the same goal with less effort, therefore, improving agent's performance. The experiments were conducted as a series of capture-the-flag games, where an agent has to find and pick up a flag, blocked by various obstacles. The authors make a conclusion that the agent exhibits reasonably human-like behavior by exploring the area, learning how to avoid obstacles, and utilizing obtained knowledge in other environments; however, no formal testing of believability (via Turing test or automated measurements) was performed.

Schrum, Karpov, and Miikkulainen (2010) describe one of three top AI agents in recent 2K BotPrize competitions among bots, acting in Unreal Tournament environment. Their AI system is based on a combination of evolved neural network and a database of traces of human behavior. Neural network is used for decision-making in battles (whether to chase opponent, retreat, strafe, stand still or pickup an item). It is first trained in a series of matches with Unreal Tournament built-in bots. Only three objectives were set during training: maximize opponent's damage, minimize own damage, and minimize collisions with level geometry. The database of human replays has two applications. First, it is used to unstuck the bot in a human-like manner, if it cannot escape from the current level location. Second, it supports human-like navigation between the given level points: if the bot needs to proceed to a certain location, and the database contains a human-generated path to that point, the bot selects this ready path.

Interestingly, two other top AI agents are not based on learning by observation. The bot by Hirono and Thawonmas (2009) is based on handcrafted finite-state machine with manually implemented "human-like" features, and the bot by Arrabales and Muñoz (2010) relies on the functions of cognitive architecture CERA-CRANIUM (Arrabales, Ledezma, & Sanchis , 2009). This architecture can be viewed as a complex rule-based system, where individual rules "compete" in order to be applied in the current game context. Why does the latter approach produce believable behavior? The authors see the source of believability in human-likeness of decision making process: the rules are selected on the basis of bot's perceptions and understanding of the current game situation, supported by its long-term memory mechanisms.

An earlier description of learning by observation-based Unreal Tournament bot is provided by Le Hy, Arrigoni, Bessière, and Lebeltel (2004). The decision-making system is represented in the following way. A bot's internal state is encoded with a vector of parameters. Each possible state has a number of outgoing transitions to the future probable states. Every outgoing transition (representing a bot's action) has an associated probability.

Altogether, these probabilities form the behavioral pattern of the bot. Authors study both manual probability distribution specification (to obtain clearly shaped behavior — e.g., aggressive or cautious) and distributions obtained through learning by observation. The experiments prove the approach to be suitable for creating bots with good level of skill: a character, trained by aggressive human player, demonstrates an aggressive behavior and beats average-leveled built-in AI agent. While the authors consider their approach as the way to construct believable bots, no believability testing was performed. However, they note that their method allows the game designer to translate own expertise easily to the AI system, contributing to believability of the bot.

The application of case-based planning methods to the real-time strategy game WARGUS[9] is studied in (Ontañón, Mishra, Sugandh, & Ram, 2007). The database of cases is created by observing human expert behavior. The system watches human's atomic actions (such as "move", "build" or "attack") and allow the user to annotate behavior patterns by specifying intended goal of the pattern (chain of atomic actions), its preconditions and "alive conditions" that have to be satisfied while the pattern is being executed (in order to have a chance to be successful). Meanwhile, the system records the world's state at different time moments. This information is used for further action retrieval. At the end, the system builds a "plan tree" — a hierarchy of goals and subgoals that should be accomplished to win the game. During the run-time, AI agent constantly examines the current world's state to determine which subgoals can be achieved, and starts executing the corresponding behavioral patterns. The algorithm also implements certain techniques to adapt extracted behavioral sequence to the current world's state and to search for alternative ways of achieving failed subgoals. The experiments showed that case-based AI performed significantly better than WARGUS' built-in AI agent. The authors evaluated two different human-supplied strategies independently, and then combined these strategies into a single knowledgebase. As expected, the later strategy was the best, as it was able to try a different approach to achieving the next subgoal, if the previous attempt has failed. Currently, the authors of this research project are working on automatic behavior optimization in order to improve AI's skill level (Mehta, Ontañón, & Ram, 2010). It should be mentioned that they also emphasize possible economic advantages of case-based reasoning approach, since behavior can be demonstrated rather than programmed.

We believe that these examples illustrate the following trend. Purely manual methods of behavior creation, primarily based on finite-state machines and behavior trees, are still popular, but they are often characterized as requiring too much effort, while obtained computer-controlled agents still can be distinguished from humans by the observers. Behavior authoring is made easier by cognitive architectures, providing built-in reasoning mechanisms to support high-level decision making. This method, however, still needs manmade description of agent acting principles. Some of these principles often can be formulated in rather abstract terms, such as AI's goals and beliefs. This capability of cognitive architectures provides a natural way to describe AI designer's intentions, thus contributing to complex, human-like behavior of obtained agents. Behavior authoring is

[9] WARGUS is a Stratagus engine-based modification of Blizzard's commercial game Warcraft 2.

minimized in learning by observation-based AI systems: agent's acting patterns are discovered automatically via machine learning methods. Currently, there are no generally preferred knowledge representation data structures and machine learning algorithms for the task of creating believable behavior.

Behavior Optimization

The projects surveyed in the last section primarily concentrate on believability in AI agents, while the problem of effectiveness is not considered as a primary aim. Since believability and effectiveness are not always connected, effective agents are often based on different principles than believable agents. In this context, it is interesting to compare 2K BotPrize believability competition among Unreal Tournament agents with StarCraft AI Competition that makes emphasis on effectiveness. As already mentioned, all three current top 2K BotPrize bots are based on different principles: (a) finite-state machine; (b) rule-based cognitive architecture; (c) neural network, trained on human-made behavioral traces. In contrast, the best current StarCraft bots rely on human-designed strategies, most often implemented as finite-state machines, and on special *ad hoc* methods, such as "mimic opponent's behavior" (EIS, 2010).

Thus, a reasonable research goal would be to combine believability and effectiveness in order to obtain AI-controlled characters, yielding higher user satisfaction. One of possible approaches to tackle this problem is to optimize the behavior of an existing believable AI-controlled character, making it more effective in terms of skill level. A number of experimental projects are devoted to such behavior optimization. Though it should be mentioned that the task of optimizing *believable* agents is rarely addressed; more often researchers try to optimize a certain AI system without discussing its believability. Usually such experiments start with an agent having a hardcoded decision-making system that reflects the authors' abilities to design a winning strategy. Next, the agent's behavior is adjusted real-time or offline using automatic optimization methods. For example, having a number of applicable behavior patterns, it is possible to encourage the use of winning patterns, meanwhile discouraging inferior action sequences (often through reinforcement learning methods (Kaelbling, Littman, & Moore, 1996)). Since AI real-time adaptation and learning optimizes effectiveness rather than believability, the result is evaluated in terms of achieving game goals rather than in being "human-like".

Bonse, Kockelkorn, Smelik, Veelders, and Moerman (2004) extended the standard Quake 3 bot with neural network that maps current game state parameters into vector of weights of possible actions, and applied Q-learning algorithm to adjust these weights. The network was trained on a large number (100 000 – 200 000) of one-on-one games (the game continues until one of the opponents is killed). The results showed that the improved bot has a 65% chance to win in a match with the original Quake 3 AI agent. The project by Bonacina, Lanzi, and Loiacono (2008) continues this research by learning better dodging (i.e. hit-avoiding) behavior. In their experiments, a neural network-enabled Quake 3 bot has to survive as long as possible in an empty room with an enemy computer-controlled bot equipped with a rocket launcher. Experiments show that the bot quickly learns dodging, which allows it to survive for more than 150 game tics, while the original bot has an average lifetime of 100 tics.

Cole, Louis, and Miles (2004) successfully applied genetic programming to the task of optimizing Counter-Strike built-in bot's behavior. A bot's playing style is defined with a set of parameters, such as adjusting weapon selection preferences and agressivity. The selection of parameters is not easy, as many non-obvious combinations can result in good gameplay, while a slight change in one of the parameters can have a noticeable negative impact on bot's effectiveness. The idea of the experiment was to employ genetic programming to automate parameter adjustment process. The authors have demonstrated that their approach can provide bots as effective as the ones designed by human experts. Moreover, genetic optimization algorithm found several different winning sets of parameters, and therefore produced a number of highly effective bots with distinct playing styles.

Research performed by Spronck et al. (2006) shows the possibility of applying a variation of reinforcement learning, called dynamic scripting, to a dynamic 3D role-playing game Neverwinter Nights. Generally, the method works according to the following scheme. When a system generates a new character, it compiles a "script" that defines its behavior from basic rules, stored in a database. Each rule has an associated numerical weight, adjusted in run-time with reinforcement learning algorithm. The weight adjustment function takes into account both the goals of individual agents and of the whole team (as agents do act in coordinated teams). Since the game had no alternative AI implementations, the authors have evaluated dynamic scripting by comparing it with straightforward rigid agent strategies ("offensive", "disabling", "defensive", etc.). Dynamic scripting clearly outperforms such static strategies. It worth noting that the authors used reinforcement learning not only to find winning behavioral patterns, but also to adjust AI to player's skills. For instance, if the computer team starts to beat the player too often, AI stops using rules having the highest weights (i.e. removes the most effective behavioral patterns).

The later example highlights two important facts: (1) it is possible to use reinforcement learning to adapt AI actions to human players rather than to search for a winning strategy; and (2) reinforcement learning can be applied to a set of already available behavioral patterns. The second observation makes us believe that human-like behavior can be combined with reinforcement learning-based optimization to achieve both believable and effective behavior. If all behavioral patterns in the knowledgebase are obtained through observation, the behavior should be believable. If the AI system uses effective patterns only (and adjusts its own strategies if necessary), the behavior becomes effective.

In the subsequent research by Bakkes, Spronck, and van den Herik (2009, 2011) this idea of "adaptation plus effectiveness" is explicitly emphasized. The authors apply a variation of a case-based reasoning algorithm to obtain adaptive and effective AI system for a real-time strategy game. The proposed solution extracts the behavioral patterns of existing game players, and uses them to build a knowledgebase for case-based reasoning. During the game, the system selects the best performing actions, according to past observations. The authors developed an evaluation function that assesses the quality of player's current situation, thus giving the ability to judge the outcome of any chosen action. As the next step, the authors are planning to implement the means for automatic adaptation to human player's skill level.

Using the background of this research into what makes believable and effective AI agents, we now can look at a specific case study of the same.

BUILDING A BELIEVABLE AND EFFECTIVE AI AGENT: A CASE STUDY

The authors of this work used a 3D boxing video game environment to experiment with some of the topics discussed above, such as: (a) ensuring believable behavior through learning by observation and case-based reasoning; (b) believability evaluation with Turing tests and automated schemes; (c) optimizing AI behavior in terms of effectiveness by means of reinforcement learning (Mozgovoy & Umarov, 2010a,b). Since the boxing game world is relatively simple, our research can serve as an easily understandable project that binds together several core concepts, and thus serves as a useful case study.

We have designed and implemented an AI system for controlling virtual boxers, having the following goals in mind:

- complex, non-repetitive behavior of AI agents;
- distinct personalities of AI boxers, exhibiting a variety of skill levels and playing styles;
- the capability to design, edit, and adjust AI's behavior (for a game designer); and
- a "train your own boxer" mode as a user-end feature.

The project included two stages. In the first stage, we implemented a learning-by-observation based AI system, capable of reproducing human behavior. We trained several AI-controlled boxers and verified their believability by means of Turing-based and automated believability tests. During the second stage, we employed reinforcement learning to optimize agents' effectiveness by encouraging repetition of the most successful behavioral patterns.

Memory Model of an AI-controlled Boxer

The above stated goals encouraged us to use a variation of finite-state machine that we call *acting graph* as a primary data structure of an AI-controlled boxer's knowledgebase (a similar solution was used by Le Hy et al. (2004)). Normally, the acting graph is being constructed automatically during learning by observation phase. A human expert plays the game, and the computer system builds the acting graph on the fly.

The nodes of this graph correspond to game situations. A game situation is a unique description of the current state of the game world, represented with a set of numerical attributes, defined by the game designer. For the game of boxing such attributes include: the coordinates of both opponents, their directions (where opponents look), body position (standing, leaning, blocking, etc.), health state of each player, and so on.

The edges of the graph correspond to the observed character's actions that introduce changes into the game states. For example, a simple action "move left" connects two game situations that have a difference in character's horizontal coordinate. Each edge also has an associated probability: while a certain game situation may have numerous outgoing actions, not all of them may be equally preferable (see Figure 1).

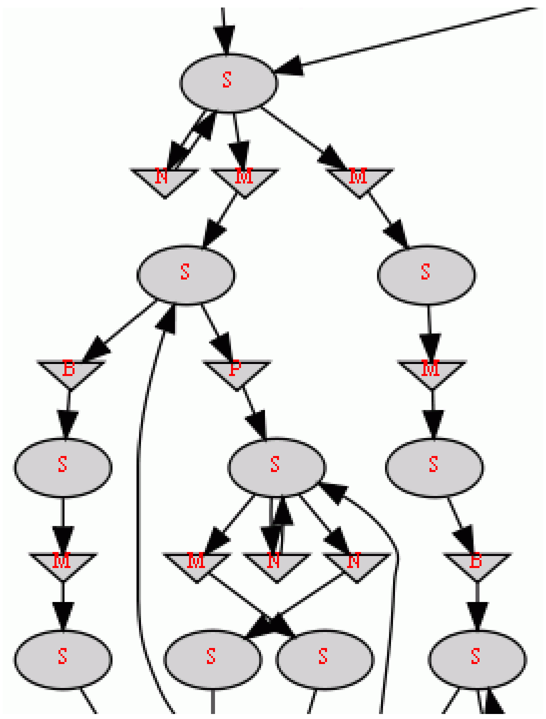

Figure 1. A fragment of acting graph (rendered with Graphviz package[10]).

The acting graph possesses the following features that help us to achieve our goals:

1. The graph stores all behavioral patterns, demonstrated by human players. Unlike many knowledge representation mechanisms, such as neural networks, it does not suppress rare training samples (often treated as "noise"): every action sequence occurred during training sessions will be preserved in the graph. Thus an AI agent acquires all idiosyncratic elements of its trainer's style.
2. Another important advantage of acting graph is related to the possibility of manual modifications. The acting graph can be visualized and edited by the game designer. One can remove unwanted or unintentional sections, create artificial acting sequences, and join separate graphs into a single knowledgebase. While this point might not seem major, it is an important factor for practical game developers, who are responsible for AI quality and prefer to have more control over system configuration.
3. Acting graph provides facilities for basic planning functions. For example, the game designer might want to implement a heuristic ranking procedure that selects the best action from a set of admissible actions, produced by the case-based reasoning subsystem. Such a procedure can traverse a graph, discover that a certain action is always weak (e.g., it always leads to game states with lower health level of the character), and discard it. While long-term planning may not be needed for boxing, it is crucial for several other types of games.

Learning and Acting

As already mentioned, in order to train an agent our system is set-up for watching a human expert actually play a match against a human- or computer-controlled opponent. Every time

[10] Available at http://www.graphviz.org.

the observed player makes an action (including “do nothing” case), the system updates AI agent’s knowledgebase. Most individual actions last 1-60 frames of animation, while the game runs at a constant speed of 60 frames per second. Thus, during a typical 15 minute learning session the system can record around 4000 samples.

In AI acting mode, the system performs case-based reasoning: it identifies a node in the acting graph that matches the current game situation, and applies one of the actions, found in outgoing edges. This process involves more complicated techniques, required by heuristic nature of matching algorithm: perfect matches are rare, so the system needs to be able to relax matching conditions gradually until an approximate match is found. Since the details are not relevant here, let us just mention that the game designer is given an ability to specify a sequence of calls of matching algorithm, parameterizing each call with a set of relaxations. A relaxation may include either excluding a certain attribute from matching or specifying a range of admissible deviations for an attribute’s value.

The readied agent has no built-in capability of distinguishing strong and weak actions; it repeats human-demonstrated behavioral patterns, even if they lead to inevitable defeat. Moreover, as experiments show, the agent, trained by a certain player, is less skillful than its trainer. It happens because the case-based reasoning system is imperfect: sometimes it fails to find the most appropriate actions that should be chosen according to the training session. Therefore, we decided to include an optional module that optimizes the agent’s behavior with the help of reinforcement learning.

Without reinforcement learning enabled, the case-based reasoning module extracts the set of actions, applicable in the given game situation, and uses weighted random choice to select the next action (action frequency is used as a weight). Reinforcement learning procedure analyzes the outcome of the just applied action (we simply check whether our boxer got more or less damage than his opponent), and modifies action weight accordingly. Successful actions get higher weights and higher probabilities of being chosen. A backpropagation routine distributes action reward among preceding actions.

Testing Believability

In order to test believability of our agent, we used a Turing test-based technique, similar to the one described in Hingston (2009). We recorded a number of one-minute game fragments between random opponents. Each opponent could be a human player; a boxer, controlled with our AI; or a boxer, controlled with a boxing engine’s built-in AI system, based on a handcrafted finite-state machine.

We asked six people with various gaming experience to watch the fragments and to guess the opponents (each boxer could be marked as “human” or “computer”). Each person was asked to analyze nine video clips. Our AI-controlled boxer was identified as human in 80.6% of cases. In contrast, the engine’s built-in AI system was able to deceive the judges in only 14% of cases (see Table 1).

Table 1. Results of the Turing test

Scenario	**Probability**	**σ**	**σ_{mean}**
Built-in AI system identified as an AI agent	86.1%	1.07	0.44
Human player identified as a human	77.8%	0.75	0.30
Machine learning-based AI identified as a human	80.6%	0.69	0.28

Since we were also curious to try automated believability tests, we implemented a method, similar to the one described by Tencé and Buche (2008). Our algorithm calculates a "style similarity ratio" for the given pre-recorded action sequences, belonging to two boxers. For each action sequence we compute a vector of probabilities of each possible combination of two successive actions. A style similarity ratio is simply a dot product of these vectors. Using this algorithm, we calculated a matrix of similarity ratios between the following boxers:

- three boxers, controlled by the built-in AI system (AI1-AI3);
- two boxers, controlled by two different human players (WEAK_H, AVG_H);
- six AI-controlled boxers, trained by the same human players (WEAK1-WEAK3, AVG1-AVG3).

The analysis of the obtained matrix revealed two clearly separated clusters: the one with the built-in AI-controlled boxer and the one with all other boxers. Thereby, the automated test supported the earlier observation that AI agents, created through learning by observation, are closer (in terms of behavior style) to human-controlled boxers than to the boxers controlled by the built-in handcrafted AI system (see Figure 2[11]).

[11] The plot is visualized with *neato* tool (included into Graphviz package); dashed lines are drawn manually.

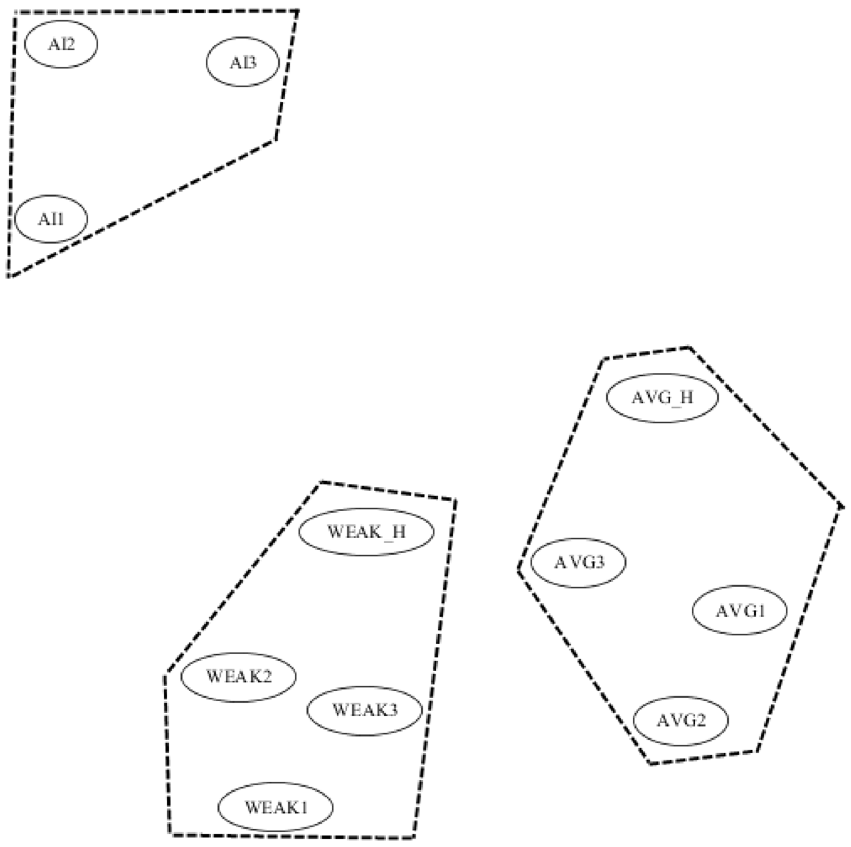

Figure 2.Visualized results of automated believability tests.

During these experiments, we made another important observation: behavior optimization through reinforcement learning does not reduce agent's believability. A believable agent remains believable even after adjustments, made by reinforcement learning. However, the style of such an agent resembles the style of its human trainer to a lesser extent. In our case, reinforcement learning introduces no new behavioral patterns: the boxer still uses the same human-supplied action sequences, only their weights, used by the weighted random choice procedure, are adjusted.

DISCUSSION

An obvious factor in the development of virtual worlds is their growing complexity. This growth concerns literally all aspects of virtual worlds — audiovisual presentation, ways of character-character interaction, physics, and world use scenarios. Generally, virtual worlds are designed to be more and more realistic, fun and immersive. As we have seen, worlds are often inhabited by both human-controlled and computer-controlled characters. So it is natural to ask, how this growing complexity affects AI systems, used in virtual worlds.

We have to admit that not all virtual worlds are tightly bound with AI-controlled characters. Worlds like World of Warcraft consider game bots as unwelcomed participants[12], and worlds like Second Life, being descendants of internet chats, are primarily populated with humans,

[12] This concerns user-created AI agents. World of Warcraft is inhabited with a limited number of AI-controlled characters, developed by Blizzard.

who are mostly interested in social interaction. But there is also a high variety of game world genres, where AI systems were always present: first-person shooters, real-time strategies, and sport games. Moreover, AI is often named by researchers and developers among important fun factors that contribute to the overall success of the game.

In its turn, complex realistic virtual worlds need complex and realistic AI-controlled characters that are able to act believably and effectively. This trend is also widely recognized, so in recent years we have seen numerous novel approaches to AI design, and even "believability competitions" for game bots. Probably, the primary problem in AI development is scalability. Most current AI systems are still handcrafted (at least, partially), but growing complexity of a virtual world entails the growing complexity and increasing volume of work, needed to create a satisfactory decision making module. Modern AI design instruments, such as behavior trees, are able to remedy the problem to some extent, but are unable to solve it completely. Scripted, scenario-based decision making methods become insufficient, and more advanced approaches are on rise as themes of practical studies.

Among general approaches to believable AI development, we would highlight two: the use of learning by observation and the use of cognitive architectures. Most probably, they can be seen as complimentary to each other.

Learning by observation is a method of behavior creation by inferring behavioral patterns from existing game characters, usually human-controlled. This idea is a variation of a well-known concept of machine learning, and can be used to acquire both believable and effective elements of behavior. As we have seen, learning by observation and case-based reasoning were indeed applied to a variety of game worlds.

Cognitive architectures are usually responsible for high-level reasoning, so their role is better tailored for strategic rather than tactic decision making. The researchers, who adopt cognitive architectures, often mention human-likeness of the obtained AI behavior as the consequence of human-likeness of the decision making procedures themselves. The algorithms, implemented in cognitive architectures, are typically based on psychophysiological models of reasoning, and this fact makes AI-controlled game characters human-like.

However, every particular game world has own peculiarities and fine points, so it is probably impossible to suggest an ultimate approach. For example, in sport games strategic decision making is less important than quick and smart reaction to immediate actions of the opponent. In contrast, real-time strategies need careful goal-based, long-term planning. Another notable topic not covered in this paper, is agent-agent coordination and cooperation. If a certain virtual world requires believable behavior of AI-controlled teams of characters, the researchers face a new dimension of complexity. This topic is a subject of numerous studies (Panait & Luke, 2005; Chernova & Veloso, 2007; Ros, Veloso, de Mántaras, Sierra, & Arcos, 2007; Stone, Kaminka, Kraus, & Rosenschein, 2010).

CONCLUSION

As virtual worlds become more and more complex, the importance of high-quality AI solutions increases. While in relatively simple “shoot ‘em up” games, traditional game AI algorithms work reasonably well, modern virtual worlds require more advanced approaches that can provide more realistic and effective behavior. Realism (or believability, human-likeness) can be itself a goal for an AI agent — especially in simulation and training applications, where believability is crucial. Effectiveness (in terms of achieving an agent’s own goals) is also harder to achieve with traditional methods in complex environments, where a winning strategy might require hierarchical goal planning and real-time adjustment to human player’s actions.

The problem of introducing believable and effective AI agents is raised both by academic researchers and video and computer games developers. However, current commercial games rarely implement AI algorithms that go beyond traditional finite state machines and rule-based systems. At the same time, research show that believable agents should possess certain features, hardly achievable without methods that rely on observing and/or simulating human behavior.

A number of research projects prove that human behavior acquisition methods, such as learning by observation and imitation learning, are applicable to complex domains such as first-person shooters and real-time strategy games. There is evidence that these methods can be combined with automatic winning strategy-selecting algorithms to provide both believable and effective behavior, as shown, for example, in the AI system for a 3D boxing game described as the case study in this paper and our earlier works (Mozgovoy & Umarov, 2010a,b). Long-term planning and strategic thinking support can be obtained from general-purpose cognitive architectures, also successfully applied in the domain of computer game worlds (Wintermute et al., 2007; Choi et al., 2007).

We believe that further research in learning by observation, advanced reasoning methods, and agent-agent coordination will be extremely useful for constructing future believable and effective AI agents, inhabiting complex, immersive virtual worlds.

REFERENCES

Adobbati, R., Marshall, A., Scholer, A., Tejada, S., Kaminka, G., Schaffer, S., & Sollitto, C. (2001). Gamebots: A 3D Virtual World Test-Bed For Multi-Agent Research, *Proc. of 2nd Workshop on Infrastructure for Agents*, pp. 47-52.

Arrabales, R., & Muñoz, J. (2010). The Awakening of Conscious Bots: Inside the Mind of the 2K BotPrize 2010 Winner. Retrieved 02.10.2011, from http://aigamedev.com/open/articles/conscious-bot/.

Arrabales, R., Ledezma, A., & Sanchis, A. (2009). CERA-CRANIUM: A Test Bed for Machine Consciousness Research, *International Workshop on Machine Consciousness*.

Bakkes, S., Spronck, P., & van den Herik, J. (2009). Rapid and Reliable Adaptation of Video Game AI. *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 1(2), pp. 93-104.

Bakkes, S., Spronck, P., & van den Herik, J. (2011). A CBR-Inspired Approach to Rapid and Reliable Adaption of Video Game AI, *Proc. of the 19th International Conference on Case-Based Reasoning*, pp. 17-26.

Balla, R.-K., & Fern, A. (2009). UCT for Tactical Assault Planning in Real-Time Strategy Games , *21st International Joint Conference on Artificial Intelligence*, pp. 40-45.

Bell, M. (2008). Toward a Definition of "Virtual Worlds". *Journal of Virtual Worlds Research*, vol. 1(1).

Blizzard. (2010). World of Warcraft Terms of Use, from http://us.blizzard.com/en-us/company/legal/wow_tou.html.

Bonacina, S., Lanzi, P., & Loiacono, D. (2008). Evolving Dodging Behavior for OpenArena using Neuroevolution of Augmenting Topologies, *PPSN'08 Workshop (Computational Intelligence and Games)*.

Bonse, R., Kockelkorn, W., Smelik, R., Veelders, P., & Moerman, W. (2004). Learning Agents in Quake III (Technical Report). *University of Utrecht, Department of Computer Science*.

Booth, M. (2004). The Official Counter-Strike Bot, *Game Developers Conference'04*.

Champandard, A. (2007). Top 10 Most Influental AI Games. *AIGameDev.com*, from http://aigamedev.com/open/highlights/top-ai-games/.

Champandard, A. (2011). This Year in Game AI: Analysis, Trends from 2010 and Predictions for 2011. Retrieved 02.10.2011, from http://aigamedev.com/open/editorial/2010-retrospective/.

Chernova, S., & Veloso, M. (2007). Multiagent Collaborative Task Learning through Imitation, *Proc. of the 4th International Symposium on Imitation in Animals and Artifacts*, pp. 74-79.

Choi, D., Kim, H., & Kim, J. (1999). Toward the Construction of Fun Computer Games: Differences in the Views of Developers and Players. *Personal and Ubiquitous Computing*, vol. 3(3), pp. 92-104.

Choi, D., Konik, T., Nejati, N., Park, C., & Langley, P. (2007). A Believable Agent for First-Person Perspective Games, *Proc. of the 3rd Artificial Intelligence and Interactive Digital Entertainment International Conference*.

Cole, N., Louis, S., & Miles, C. (2004). Using a Genetic Algorithm to Tune First-Person Shooter Bots, *Proc. of the International Congress on Evolutionary Computation*, pp. 139-145.

Csikszentmihalyi, M. (1991). *Flow: The Psychology of Optimal Experience*. New York: Harper Perennial, 320 p.

EIS. (2010). Expressive Intelligence Studio: 2010 StarCraft AI Competition Results. Retrieved 30.09.2011, from http://eis.ucsc.edu/StarCraftAICompetition#Results.

El Rhalibi, A., & Merabti, M. (2008). A Hybrid Fuzzy ANN System for Agent Adaptation in a First Person Shooter. *International Journal of Computer Games Technology*, pp. 1-18.

Eyewitness: Complete AI Interviews. (2002, September 18). *PC Gamer*. Retrieved 30.09.2011 from http://tinyurl.com/eyewitness-2002-09-18

Friedman, D., Steed, A., & Slater, M. (2007). Spatial Social Behavior in Second Life. *Lecture Notes in Computer Science*, vol. 4722, pp. 252-263.

Glende, A. (2004). Agent Design to Pass Computer Games, *Proc. of the 42nd Annual ACM Southeast Regional Conference*, pp. 414-415.

Gorman, B., Thurau, C., Bauckhage, C., & Humphrys, M. (2006). Believability Testing and Bayesian Imitation in Interactive Computer Games. *Lecture Notes in Computer Science*, vol. 4095, pp. 655-666.

Graesser, A., Lang, K., & Roberts, R. (1991). Question Answering in the Context of Stories. *Journal of Experimental Psychology: General*, vol. 120(3), pp. 254-277.

Hagelbäck, J., & Johansson, S. (2009). AMultiagent Potential Field-Based Bot for Real-Time Strategy Games. *International Journal of Computer Games Technology*, pp. 1-10.

Hecker, C. (2011). My Liner Notes for Spore. Retrieved 20.09.2011, from http://chrishecker.com/My_liner_notes_for_spore.

Hingston, P. (2009). A Turing Test for Computer Game Bots. *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 1(3), pp. 169-186.

Hirono, D., & Thawonmas, R. (2009). Implementation of a Human-Like Bot in a First Person Shooter: Second Place Bot at BotPrize 2008, *Proc. of Asia Simulation Conference*.

ICARUS. (2007). ICARUS Project, from http://cll.stanford.edu/research/ongoing/icarus/.

Isla, D. (2005). Handling Complexity in the Halo 2 AI, *Game Developers Conference'05*.

Kaelbling, L., Littman, M., & Moore, A. (1996). Reinforcement Learning: A Survey. *Journal of Artificial Intelligence Research*, vol. 4, pp. 237-285.

Kemmerling, M., Ackermann, N., Beume, N., Preuss, M., Uellenbeck, S., & Walz, W. (2009). Is Human-Like and Well Playing Contradictory for Diplomacy Bots?, *IEEE Symposium on Computational Intelligence and Games*, pp. 209-216.

Kitano, H., Asada, M., Kuniyoshi, Y., Noda, I., Osawai, E., & Matsubara, H. (1998). RoboCup: A Challenge Problem for AI and Robotics. *Lecture Notes in Computer Science*, vol. 1395, pp. 1-19.

Knafla, B. (2011). Introduction to Behavior Trees. Retrieved 15.09.2011, from http://bjoernknafla.com/introduction-to-behavior-trees.

Laird, J. (2002). Research in Human-level AI using Computer Games. *Communications of the ACM*, vol. 45, pp. 32-35.

Langley, P., Choi, D., & Rogers, S. (2005). Interleaving Learning, Problem-Solving, and Execution in the ICARUS Architecture (Technical Report). *Computational Learning Laboratory, Stanford University*.

Le Hy, R., Arrigoni, A., Bessière, P., & Lebeltel, O. (2004). Teaching Bayesian Behaviours to Video Game Characters. *Robotics and Autonomous Systems*, vol. 47, pp. 177-185.

Lee, F., & Gamard, S. (2003). Hide and Seek: Using Computational Cognitive Models to Develop and Test Autonomous Cognitive Agents for Complex Dynamic Tasks, *Proc. of the 25th Annual Conference of the Cognitive Science Society*.

Lichocki, P., Krawiec, K., & Jaśkowski, W. (2009). Evolving Teams of Cooperating Agents for Real-Time Strategy Game. *Lecture Notes in Computer Science*, vol. 5484, pp. 333-342.

Livingstone, D. (2006). Turing's Test and Believable AI in Games. *Computers in Entertainment*, vol. 4(1), pp. 6-18.

Malone, T. (1981). What Makes Computer Games Fun? *Byte*, vol. 6(12), pp. 258-278.

Manojlovich, J., Prasithsangaree, P., Hughes, S., Chen, J., & Lewis, M. (2003). UTSAF: A Multi-agent-based Framework for Supporting Military-based Distributed Interactive Simulations in 3D Virtual Environments, *Proc. of 2003 Simulation Conference*.

Mehta, M., Ontañón, S., & Ram, A. (2010). Meta-Level Behavior Adaptation in Real- Time Strategy Games, *ICCBR 2010 Workshop on Case Based Reasoning for Computer Games*.

Milward, D. (2009). List of Neverwinter Nights 2 NPCs. Retrieved 03.10.2011, from http://www.sorcerers.net/Games/NWN2/Walkthrough/NPCList.php.

Mitchell, D. (1995). From MUDs to Virtual Worlds. *Microsoft Virtual Worlds Group*.

Mozgovoy, M., & Umarov, I. (2010a). Building a Believable Agent for a 3D Boxing Simulation Game, *Proc. of the 2nd International Conference on Computer Research and Development*, pp. 46-50.

Mozgovoy, M., & Umarov, I. (2010b). Building a Believable and Effective Agent for a 3D Boxing Simulation Game, *Proc. of the 3rd IEEE International Conference on Computer Science and Information Technology*, pp. 14-18.

Naveed, M., Kitchin, D., & Crampton, A. (2010). Monte-Carlo Planning for Pathfinding in Real-Time Strategy Games, *Proceedings of PlanSIG Workshop*, pp. 125-132.

Ontañón, S., Mishra, K., Sugandh, N., & Ram, A. (2007). Case-based Planning and Execution for Real-time Strategy Games. *Lecture Notes in Computer Science*, vol. 4626, pp. 164-178.

Orkin, J. (2006). Three States and a Plan: the AI of FEAR, *Game Developers Conference'06*.

Panait, L., & Luke, S. (2005). Cooperative Multi-Agent Learning: The State of the Art. *Autonomous Agents and Multi-Agent Systems*, vol. 11(3), pp. 387-434.

Pao, H.-K., Chen, K.-T., & Chang, H.-C. (2010). Game Bot Detection via Avatar Trajectory Analysis. *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 2(3), pp. 162-175.

Pillosu, R. (2009). Coordinating Agents with Behavior Trees: Synchronizing Multiple Agents in CryEngine 2. *AIGameDev.com*.

Riedl, M., & Young, R. (2005). An Objective Character Believability Evaluation Procedure for Multi-Agent Story Generation Systems. *Intelligent Virtual Agents*, pp. 278-291.

Robertson, J., & Good, J. (2005). Adventure Author: an Authoring Tool for 3D Virtual Reality Story Construction, *AIED-05 Workshop on Narrative Learning Environments*, pp. 63-69.

Ros, R., Veloso, M., de Mántaras, R., Sierra, C., & Arcos, J. (2007). Beyond Individualism: Modeling Team Playing Behavior, *Proc. of the National Conference on Artificial Intelligence*, vol. 22(2), pp. 1671-1674.

Sánchez-Crespo Dalmau, D. (2005). Postcard from GDC 2005: Tutorial — Machine Learning. *Gamasutra (March, 8)*.

Schrum, J., Karpov, I., & Miikkulainen, R. (2010). UT^2: Human-like Behavior via Neuroevolution of Combat Behavior and Replay of Human Traces. *The University of Texas at Austin*.

Smith, M., Lee-Urban, S., & Munoz-Avila, H. (2007). RETALIATE: Learning Winning Policies in First-person Shooter Games, *Proc. of the National Conference on Artificial Intelligence*, pp. 1801-1806.

Soni, B., & Hingston, P. (2008). Bots Trained to Play Like a Human are More Fun, *IEEE International Joint Conference on Neural Networks*, pp. 363-369.

Spronck, P., Ponsen, M., Sprinkhuizen-Kuyper, I., & Postma, E. (2006). Adaptive Game AI with Dynamic Scripting. *Machine Learning*, vol. 63, pp. 217-248.

Stone, P., Kaminka, G., Kraus, S., & Rosenschein, J. (2010). Ad Hoc Autonomous Agent Teams: Collaboration without Pre-Coordination, *Proc. of the 24th AAAI Conference on Artificial Intelligence*, pp. 1504-1509.

Sweetser, P., & Wyeth, P. (2005). GameFlow: a Model for Evaluating Player Enjoyment in Games. *Computers in Entertainment*, vol. 3(3).

Taatgen, N., van Oploo, M., Braaksma, J., & Niemantsverdriet, J. (2003). How to Construct a Believable Opponent using Cognitive Modeling in the Game of Set, *Proc. of the 5th International Conference on Cognitive Modeling*, pp. 201-206.

Tencé, F., & Buche, C. (2008). Automatable Evaluation Method Oriented toward Behaviour Believability for Video Games, *International Conference on Intelligent Games and Simulation*, pp. 39-43.

Thurau, C., Bauckhage, C., & Sagerer, G. (2004). Learning Human-like Movement Behavior for Computer Games, *Proc. of the 8th International Conference on the Simulation of Adaptive Behavior (SAB'04).*

Turing, A. (1950). Computing Machinery and Intelligence. *Mind*, vol. 59, 433.

van der Heijden, M., Bakkes, S., & Spronck, P. (2008). Dynamic Formations in Real-Time Strategy Games, *IEEE Symposium on Computational Intelligence and Games*, pp. 47-54.

Wang, H., Shen, C., & Ritterfeld, U. (2009). Enjoyment of Digital Games. *In: U. Ritterfeld, M. Cody, & P. Vorderer (Eds.), Serious Games: Mechanisms and Effects*, pp. 25-47.

Westra, J., & Dignum, F. (2009). Evolutionary Neural Networks for Non-Player Characters in Quake III, *IEEE Symposium on Computational Intelligence and Games*, pp. 302-309.

Wintermute, S., Xu, J., & Irizarry, J. (2007). SORTS Tech Report. *Artificial Intelligence Lab, University of Michigan.*

Yannakakis, G., & Hallam, J. (2004). Evolving Opponents for Interesting Interactive Computer Games. *From Animals to Animats*, vol. 8, pp. 499-508.

Yannakakis, G., & Hallam, J. (2005). A Generic Approach for Obtaining Higher Entertainment in Predator/Prey Computer Games. *Journal of Game Development*, vol. 1(3), pp. 23-50.

Yannakakis, G., & Hallam, J. (2007). Towards Optimizing Entertainment in Computer Games. *Applied Artificial Intelligence*, vol. 21, pp. 933-971.

Zyda, M., Hiles, J., Mayberry, A., Wardynski, C., Capps, M., Osborn, B., et al. (2003). The MOVES Institute's Army Game Project: Entertainment R&D for Defense. *IEEE Computer Graphics and Applications*, vol. 23, pp. 28-36.