Optimising Humanness: Designing the best human-like Bot for Unreal Tournament 2004

Antonio M. Mora[1], Álvaro Gutiérrez-Rodríguez[2], Antonio J. Fernández-Leiva[2]

[1] Departamento de Teoría de la Señal, Telemática y Comunicaciones.
ETSIIT-CITIC, Universidad de Granada (Spain).
amorag@ugr.es
[2] Departamento de Lenguajes y Ciencias de la Computación,
Universidad de Málaga (Spain).
alvarogutirodri@hotmail.com,afdez@lcc.uma.es

Abstract. This paper presents multiple hybridizations of the two best bots on the BotPrize 2014 competition, which sought for the best human-like bot playing the First Person Shooter game Unreal Tournament 2004. To this aim the participants were evaluated using a Turing test in the game. The work considers MirrorBot (the winner) and NizorBot (the second) codes and combines them in two different approaches, aiming to obtain a bot able to show the best behaviour overall. There is also an evolutionary version on MirrorBot, which has been optimized by means of a Genetic Algorithm. The new and the original bots have been tested in a new, open, and public Turing test whose results show that the evolutionary version of MirrorBot apparently improves the original bot, and also that one of the novel approaches gets a good humanness level.

1 Introduction

Most of modern videogames are designed to provoke intense feelings on the player. To this end, many times they include non-player characters (NPCs) who try to empathize with the human, showing human-like behaviors and feelings. Thus, they implement internally an Artificial Intelligence (AI) engine focused on their *humanness*, i.e. including a set of rules to guide their actions similar to those that a real human would follow. It is also called *believability* of the NPCs.

Thus, the final aim is that those characters would be able to pass a Turing test [1] inside the game, which would mean that the human player could lead to think that they are 'real humans'. In this line, some years ago the *2K Botprize Competition* arose. It had as objective to find the best human-like NPC (or Bot) for the First Person Shooter (FPS) game Unreal Tournament™ 2004[2], also known as UT2K4. That game, in its DeathMatch mode (combats between two or more bots trying to defeat the opponents an survive), was considered as the scenario for evaluating the bots in order to pass an adapted version of the Turing Test inside the game.

In the last edition of the competition, 2014, the two first Bots obtained very good humanness levels. They were named *MirrorBot* [3] and *NizorBot* [4].

This paper continues a previous work [5] in which those bots were analyzed and their advantages and weaknesses were identified. In that study several ways for optimizing and combining those bots were suggested, so, in the present paper some of them have been addressed. Thus, here we describe two different hybrid approaches which combine the best parts of every bot with different features from the other one. Moreover, an evolutionary-based improvement of the winner of the competition, MirrorBot, is also presented.

The bots have been tested using an open/public and online Turing Test, in which anonymous people have judged the humanness of the bots in many different videos showing battles.

2 State of the art

The objective of creating human-like or believable characters in videogames is to show a life-like behaviour, including features such as personality, emotions, empathy or almost-real movements. There are normally interactions between the controlled characters in the game, so, the aim is then to show the illusion that these virtual players are controlled by a human [6]. This is a very important issue in current games, because this can enhance, for instance, the immersion of the player in the game and thus, his/her satisfaction [7].

However, evaluating the level of humanness that a virtual player exhibits is quite difficult, as it is normally a subjective measure. Thus, a way to evaluate this 'believability' can be the Turing test [1], or, currently, its adaptation to the scope of videogames [6] such as the 2K Botprize Competition (see Section 3).

Nevertheless, modelling a credible human-like behaviour is a very hard task, since it is not as simply as following a predefined set of states or mathematical formulae. So, the usual solutions [8] are focused on 'simulating' typically human actions, such as medium or high-level effectiveness in playing, make somehow unexpected mistakes from time to time, take different decisions even in the same conditions (with a stochastic factor), and show any kind of 'emotion'.

FPS games are one of the most considered scenarios for Turing test in videogames. Thus, there have been several proposals of human-like agents in this scope. From the SOAR Bot for Quake, presented by Laird [9] in 2000, which modelled a human-like behaviour through the so-called cognitive architecture. Choi et al. in [10] improved that architecture in an autonomous agent for the game Urban Combat. Their enhanced version was able to use knowledge and learn, by means of memories, such as skill or prioritized list of goals the agent should attempt to achieve.

However, the most extended environment for human-like bots in FPSs has been Unreal Tournament™ 2004 game (UT2K4). Several proposals in this scope have applied a variation of Evolutionary Algorithms (EAs). For instance [11] which implemented evolution and co-evolution techniques, or [12] in which an evolutionary rule-based system was used. Schrum et al. [13] considered the combination of EAs with Artificial Neural Networks (ANNs) in order to learn to play as a human by imitating players' traces. Or the proposal by Soni and Hingston [14] which tried to imitate human's behaviour by means of ANNs.

Finally, the authors presented in [15] a bot which modelled the behaviour of an expert Spanish human player. It included a two-level FSM, in which the main states define the high level behaviour of the bot (such as attack o retreat), meanwhile the secondary states (or substates) can modify this behaviour in order to meet immediate or necessary objectives (such as taking health packages or a powerful weapon that is close to the bot's position). This approach was improved through a parameter-tuning made by means of an EA.

As stated before, this paper tries go a step further to the creation of the best human-like bot through the hybridization of two of the best bots in the last BotPrize Competition, and also enhancing the winner of that competition applying an evolutionary method.

3 Botprize competition: A Turing Test for bots

This test is a variation of the classical Turing Test in which a human judge who looks and interacts with a virtual world (a game), must to distinguish if the other actors (players) in the game are humans or bots. This test was proposed in order to advance in the fields of Artificial and Computational Intelligence in videogames, since a bot which can pass this test could be considered as excellent, and could increase the quality of the game from the players' point of view. Moreover the test tries to prove that the problem of AI for videogames is far from being solved.

The Turing Test for bots is focused on a mutiplayer game in which the bot has to cooperate or fight against other players (humans or bots), making the same decisions that a human would take. This was transformed into an international competition with the features:

- The Deathmatch mode is considered in rounds of 10 minutes.
- There will be (ideally) three players: a human player, a bot and a judge.
- The bot must 'simulate' to be more human than the human player and both of them receive an independent mark.
- The three players cannot be distinguished from 'outside' (even with a random name and appearance).
- Bots cannot have omniscient powers as in other games. They can just react to the same stimuli (caught by means of sensors) than a human player.

In 2008 it was held the first 2K BotPrize competition (BotPrize from now on), in which UT2K4 was considered as the 'world' for this test. The participants should create their human-like bots using an external library to interact with the game by means of TCP connections (Pogamut [16]).

In the first editions of Botprize (2008 to 2011) the marks of the bots were not able to overcome to any of the human players. Anyway, the maximum humanness score for the human players was just 41.4 %. This demonstrates the limitations of the test (or the competition), since even appraise a human behaviour is a quite complex task.

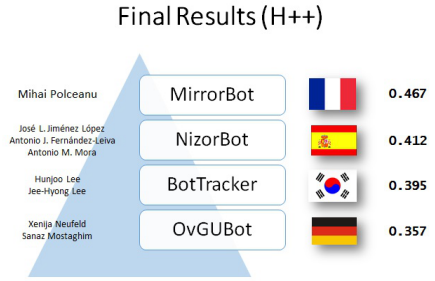
The first two bots in 2014 edition of Botprize were MirrotBot [3] (which also won 2012 edition) created by Mihai Polceanu; and a proposal by J.L. Jiménez and two of the authors of this study, *NizorBot* [4].

In the original competition, a number of judges that participated directly in the matches were responsible with the evaluation of humanness of bots; this means a First Person Assessment (FPA). In the edition of 2014, a Third Person Assessment (TPA) was also included by means of the participation of (external) judges via a crowdsourcing platform. The humanness (H) was evaluated according to the following formula:

$$H = (FPA * FP_{wf}) + (TPA * TP_{wf}) \quad (1)$$

where FP_{wf} and TP_{wf} are weighting factors (ranging in [0.0,1.0]) for FPA and TPA respectively. For the 2014 edition, $FP_{wf} = TP_{wf} = 0.5$.

The results of 2014 Competition are plotted in Figure 1. As the results figures



BotName	FPA	TPA	H++
Xenija	0.17139763	0.8235294	0.4974635
MirrorBot	0.20164771	0.7333333	0.4674905
Player	0.19328127	0.6315789	0.4124301
tmchojo	0.17757519	0.6470588	0.4123170
NizorBot	0.11821633	0.7058824	0.4120493
BotTracker	0.20070203	0.5909091	0.3958056
CCBot	0.06214746	0.7058824	0.3840149
Juan_CVC	0.12372294	0.6190476	0.3713853
OvGUBot	0.10545765	0.6086957	0.3570767
ADANN	0.08351664	0.4761905	0.2798536

Fig. 1. Results of Botprize 2014. Yellow cells are the best competitors whereas blue ones where humans.

show, MirrorBot was very close to completely pass the Turing test proposed in that edition, which was a new and harder evaluation system, it does no reach the value for being consider human (i.e., 0.5) although is relatively close to it. NizorBot showed also a very good performance finishing in the second position, obtaining a humanity factor relatively close to be considered as human.

4 MirrorBot and NizorBot

This section presents the two considered bots in our study as the basis of the created hybridizations.

4.1 MirrorBot

MirrorBot [3] was developed in 2012, specifically for the 2K BotPrize competition, and was submitted again for the 2014 edition, which it won. It is based in two main behavioural modules:

– *Default module*: used frequently to navigate through the map, gathering items and weapons, shooting to enemies and avoiding their attacks. It is composed by several submodules. The three main ones let the bot *aiming* automatically to enemies computing trajectories or to a point in the path to simulate anticipation; *navigating* applying a modified version of the standard graph navigation which adds some kind of 'noise' or distortion to the movement, in order to hide bot-like displacement; and finally, *shooting* to the most appropriate enemy taking into account its weapon type, splash damage and distance to it.
– *Mirroring module*: which is only activated when an enemy is considered as unaggressive (every opponent is considered like this by default, until it shoots to the bot). The reason is that the enemy is probably a (human) judge inside the game (FPA). When the mirroring behavior is activated for a target, MirrorBot will begin recording all observable low-level actions of the opponent: aim, movement, fire, jumping, crouching and weapon choice. These are stored as frames in a sequence, which are to be replayed by MirrorBot itself. The orientation is inverted and movement maintains a constant distance to the target. Additionally a delay is introduced in the sequence in order to cheat the judge looking at this bot.

4.2 NizorBot

NizorBot [4] was based on the idea shown in the aforementioned ExpertBot [15], which modelled the behavior of an expert human player using a two-level finite state machine (FSMs).

ExpertBot was formed by two layers: The first one is the *cognitive layer*, in charge of controlling the FSM taking into consideration the environmental stimuli (perceived by sensors). It decides the transitions between states and substates using the expert system and the knowledge database. The second one is the *reactive layer*, which does not perform any kind of reasoning, and just reacts immediately to events during the match.

NizorBot is an implementation over ExpertBot by applying an Interactive Evolutionary Algorithm (IEA) [17], in which human experts guide the optimization of the bot's parameters in order to obtain a human-like bot. The basic idea is to let the experts rule out those candidate solutions (i.e., individuals) that perform *subjectively* worse than others from the point of view of humanness. More specifically, every **individual** in the IEA is a chromosome with 26 genes, divided into 6 blocks of information. Each block represents the behavior of a specific feature of the bot: distance, weapon selection, weapon priority, profile, risk, time.

The **fitness function** to evaluate the individuals is a combination of enemy kills (frags), number of own deaths, and the damages dealt and received by the bot. The function rewards the individuals with a positive balance (more frags than deaths) and a high number of frags. In addition, individuals which deal a high amount of damage to enemies are also rewarded, even if they have not got a good balance.

The *evaluation of an individual* consists of setting the values of the chromosome in the NizorBot AI engine, then a 1 vs 1 combat is launched between this and a standard UT2K4 bot at its maximum difficulty level. Once the time defined for the match is finished, the summary of the individual (bot) performance regarding these values is considered for the fitness computation.

Regarding the genetic operators considered, a *probability roulette wheel* has been used as **selection mechanism**, with *5 elitism*. **Uniform crossover** operator is applied, so that every gene of a descendent has the same probability of belonging to each one of the parents.

The **interaction of the game expert** has been conducted at some specific points of the evolution, where the expert should conduct a TPA (watching a video of the bot) and identify those specific features (e.g. distance selection, weapon selection, etc.) that they consider more human-like in the bot. Then, the gene blocks associated to the selected features are 'blocked' so that they are not altered by the genetic operators during the evolution. This affects the rest of the population when this individual combines and spreads its genetic information. This interaction guides the search to find more human-like individuals.

5 Hybrid Bots

Two bots have been proposed in this paper as hybrid approaches of MirrorBot and NizorBot. Regarding this one, we have used the parameter setting of the best individual obtained in our previous work [4], after the whole interactive evolutionary process.

Each hybrid bot combines the best part of one of the two reference bots with a complementary part from the other. These are namely:

- **MIRZorBot**: This bot combines the best parts of MirrorBot, namely, the navigation module (target selection, pathfinding, aiming and movement), and the mirroring ability (mirroring module). As these have been considered as the key of MirrorBot's human-like behaviour, after a deep analysis. The aiming and movement seem to be conditioned by the human perception in the game. The aiming is not always perfect. The mirroring module adds an unexpected - but close-to-natural - behaviour.
  These modules and submodules have been included inside the NizorBot's FSM, which also adds the expert weapon selection system - very proficient as it was designed by a human expert taking into account many weapon-related parameters -. Moreover, the division into primary and secondary states seems to be very close to the actual human's priorities in the game.
- **NIZRorBot**: This bot uses almost all the internal structure of NizorBot, however it includes the navigation module of MirrorBot, as it was the weakest part of the initial ExpertBot [15]. Thus, NIZRorBot makes use of the navigation module with all the 'tricks' that MirrorBot implements, including an improved implementation of Pogamut's Navigation Mesh (with optimal obstacle avoidance) and an own RayCasting system based in 24 rays: 16 for the

detection of horizontal collisions (direct obstacles, items, weapons, enemies), and 8 for vertical collisions (45 ˚for the detection of non-floor, holes or falls).

6 Evolutionary MirrorBot

In addition to the two hybrid proposals, here we present an improvement of the initial MirrorBot, *EVOMirBot*. It is based on a parameter tuning or optimization by means of a classic Genetic Algorithm (GA) [18].

The aim is to enhance the overall behaviour of the original bot, in order to show a more 'offensive profile' (i.e. being more aggressive), since sometimes it just waits for the opponent's actions and do not react properly, i.e. as a human would do (just moves around the rival). To this end, the hand-coded parameters on which it depends the behaviour of this bot, have been identified and 'extracted', in order to compose a chromosome or individual for the GA.

Every individual in the GA is a chromosome with 12 genes, namely:

- *Gene 1* (initial value = 8136): time devoted to imitate, once a potential human player has been identified for mirroring. It is measured in milliseconds in the range [1000,10000].
- *Gene 2* (initial value = 2982): time to consider a target enemy as lost. Milliseconds in the range [1000,9000].
- *Gene 3* (initial value = 7): aggressiveness level of the enemy. It the value is greater than this gen, the opponent is discarded for mirroring (it is too aggressive for being a human). Value in the range [1,10].
- *Gene 4* (initial value = 115): imitation delay when the mirroring module is reproducing the recorded movements of the opponent. This value is very important to 'guide' the observer's impression regarding the behaviour the bot is showing. Value in milliseconds in the range [0,500].
- *Gene 5* (initial value = 842): last time the opponent to be imitated was seen. This value represents the end of the imitation flow. It is measured in milliseconds and in the range [0,1000].
- *Gene 6* (initial value = 5): voting value to consider an opponent to be imitated. If it receives more than this number, it will become mirrored. Value in the range [1,7].
- *Gene 7* (initial value = 2000): average distance between the bot and the candidate rival to be imitated. It is adjusted in order to observe the other bot without being attacked by it. Value in the range [1200, 2000].
- *Gene 8* (initial value = 8211): time considering an opponent as 'nemesis', i.e. the bot will attack it as soon as it is detected. Value in milliseconds in the range [1000,9000].
- *Gene 9* (initial value = 2142): time to forget the list of nemeses. Value in milliseconds in the range [1000,5000].
- *Gene 10* (initial value = 300): time for uncontrolled (or pseudo-random) shooting. This is done to show an unexpected behaviour from time to time. Value in milliseconds in the range [100,500].

– *Gene 11* (initial value = 3500): distance considered as far from the enemy. Value in [500,5000].
– *Gene 12* (initial value = 600): distance considered as short from the enemy. Value in [100,800].

The list shows the initial values that MirrorBot had set, as a reference. As it can be seen, the ranges have been defined so a great variation of MirrorBot could be obtained through evolution.

The *fitness function* is defined as:

$$f(fr, d, dmgG, dmgT) = ((fr * 50) - (d * 5)) + (dmgG - dmgT/10) \quad (2)$$

Where fr is the number of enemy kills the bot has obtained (frags), d is the number of own deads, $dmgG$ is the total damage produced by the bot, and $dmgT$ is the total damage it has received. As in NizorBot, this function rewards a lot individuals with positive balances, i.e. more kills than deads and more produced than received damage, aiming to obtain the aforementioned 'offensive profile'.

The *evaluation of an individual* is done setting the values of the chromosome in the MirrorBot's AI code and then running a 1 vs 1 Deathmatch against NizorBot in UT2K4 during 1 minute. Once the battle finishes the fitness is computed considering the performance of the bot.

A *probability roulette wheel* has been used as selection mechanism, considering the fitness value as a proportion of this probability. In addition, a *Stationary* replacement policy has been conducted, so just the worse individual is replaced every generation. Finally, *uniform crossover* has been applied, and the mutation generates a random value in the corresponding interval of the parameter.

7 Experiments and results

This section analyzes the obtained results, first regarding the evolutionary approach of MirrorBot, and by means of a Third Person Assessment Turing Test.

7.1 Evolutionary optimization

In this experiment the so-called EVOMirBot has been obtained as an optimization of MirrorBot. The parameter setting has been: 30 individuals, 50 generations, 1/12 of mutation probability. The evaluation has been conducted as a 1-minute 1 vs 1 combat against NizorBot, always in the map DM-TrainingDay (frequently used in the UT2K4 competitions). 10 runs have been conducted.

The evolution of the fitness for all the runs is plotted in Figure 2.

As it can be seen, there is an improvement tendency on the average best fitness along generations. However it is a bit 'slight', due to the *noisy* nature of the problem [19], i.e. an individual can be valued as good in one combat, but the same bot yield very bad results in another match. This happens due to the high pseudo-stochastic component present in these battles, since the results do

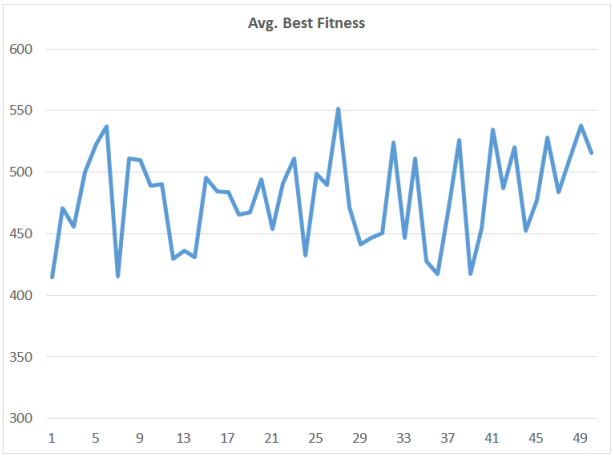


Fig. 2. Evolution of the average of the best fitness per generation considering 10 runs.

not depend completely on our bot, but also on the enemy's actions which we cannot control.

Thus, we selected as the definitive EVOMirBot one individual from the last generations. It was not the one with the highest fitness value, but a bot which showed the most human-like behaviour from our point of view (we checked several combats of each candidate bot). The obtained optimized values for the parameters of this bot are presented in Table 1.

Table 1. Gene values for the final EVOMirBot

Gn1	**Gn2**	**Gn3**	**Gn4**	**Gn5**	**Gn6**	**Gn7**	**Gn8**	**Gn9**	**Gn10**	**Gn11**	**Gn12**
6231	5364	1	25	612	2	1560	3194	2880	447	3398	245

Looking at the results, we can remark that the aggressiveness of the MirrorBot has been increased overall in EVOMirBot. For instance *Gene 2* (changed from 2982 to 5364) means that the bot will 'remember' for longer time its nemesis, *Gene 8* (value 3194 instead of 8211) will affect the time the bot considers another one as its nemesis. *Gene 12* (changed to 245 from 600) will turn the measure for short distances, which will lead to a closer combat style.

Regarding the imitation ability of MirrorBot, its has been enhanced or better adapted for the game. Thus, for instance the value for *Gene 1* is much lower than the original (6231 instead of 8136), which means than the bot will be less time imitating the opponent. *Gene 3* new value (1 instead of 7) will lead to a lower imitation rate, but the opponents, considered as bots will be attacked more frequently, which is more recommended in this game. The new value of *Gene 5* (612 instead of 842) will mean that the imitation will start sooner. *Gene 6* (value 2 instead of 6) will affect the probability of choosing candidates for imitation. The

new value of *Gene 10* (447 instead of 300) will lead to EVOMirBot to conduct more random or imprecise shoots, which probably will increase the perception of an inaccurate player which would be more likely to be a human.

7.2 Open TPA Turing Test

In this experiment the three new bots (MIRZorBot, NIZRorBot, and EVOMirBot) and the originals NizorBot and MirrorBot have been evaluated in an open Turing Test based in a Third Person Assessment (TPA) on the website `http://1-dot-proyecto-tfg.appspot.com/`.

To this end 7 different videos, of 20 seconds each, have been recorded and presented to the 'voluntary judges' (whoever has accessed the web). Every video shows a short combat stage between two players, being each of them one of the bots or a human. Thus, every bot has participated in two videos: one against another bot and one against a human player.

After a video, the judge must decide about the humanness of the contenders, with the options: *a) player 1 is human*, *b) player 2 is human*, *c) both are humans*, *d) none is human*, *e) not sure*, *f) wrong test.*

The test has been open during three weeks, and 61 judges have participated. We have considered the votes for every bot as human, and also the votes to both of them. The rest are omitted in the computation of humanness level for every bot. This value has been calculated as the number of votes received divided by 122, which is the maximum number of votes that a bot could receive (61 votes x 2 videos in which the bot is present). The results are shown in Table 2.

Table 2. TPA Turing Test results.

Bot	Votes as human	Humanness
NizorBot	57	46.72
EVOMirBot	53	43.44
MIRZorBot	51	41.80
MirrorBot	47	38.52
NIZRorBot	46	37.70

It can be seen high voting values for all the bots, which is a good sign of their human-like behaviour.

The most remarkable fact that we can see in these results is the (apparent) improvement that EVOMirBot have meant with respect to MirrorBot. The first has obtained the second best results in the test, just behind NizorBot, which has 'won'. MIRZorBot has also obtained good results, however NIZRorBot has been the worse. The reason for poor performance is probably that the raycasting system or MirrorBot has been in conflict with the target selection method of the original NizorBot, which has meant a non-proper movement behaviour.

However, looking at the whole figures in the results, we think that there is place for improvement in the videos, such as their duration (maybe too short),

or the point of view which is not probably the best to evaluate the opponent. But in this kind of open test, it is very important to reach an accurate number of videos with an accurate duration, in order to avoid tiredness or disappointment in the judges.

8 Conclusions and future work

This paper has presented three different approaches for human-like bots for the First Person Shooter Unreal Tournament 2004. All of them have been obtained as variation/enhancement of the two first bots in the last 2014 edition of the 2K Botprize Competition (a Turing test for bots): MirrorBot and NizorBot.

These are *MIRZorBot* (based on MirrorBot with some components of NizorBot), *NIZRorBot* (structure of NizorBot with movement module of MirrorBot), and *EVOMirBot* (evolutionary optimization of MirrorBot).

In the results we have firstly analyzed the obtained improvement of MirrorBot by means of a Genetic Algorithm, paying attention to the new values for the parameters and their influence on the bot's behaviour, getting, in summary, a more aggressive bot.

Then, an open and online Turing Test has been conducted as a Third Person Assessment, so the voluntary judges have revised some videos of the bots fighting in the game and decided about who is the human (if there is any in the match). The results of this test yield two main conclusions: EVOMirBot seems to be a real improvement of MirrorBot, and MIRZorBot has obtained a very good humanness level. NIZRorBot has got a worse value, but the reason could be an incompatibility between one of the modules with a raycasting system, which we will solve in the near future.

Other future lines of work will be dealing with the noise in the evolutionary process (evaluation function), in order to get a better improvement progression. In addition, taking into account the voting results in the Turing test, there have been some 'not sure' or 'wrong test votes, which lead us to think that the videos must be improved maybe better focusing on every bot, with a longer duration or reducing the vote to just one bot per video.

Acknowledgements

This work has been supported by MINECO project EPHEMECH (TIN2014-56494-C4-1-P, 3-P), and KNOWAVES (TEC2015-68752) (MICINN and FEDER), and Universidad de Málaga (Campus de Excelencia Internacional Andalucía Tech). The authors are very grateful to Mihai Polceanu and José L. Jiménez, authors respectively of MirrorBot and NizorBot, for providing us their source code and their support for the development of this work.

References

1. Turing, A.M.: Computing Machinery and Intelligence. Mind **59**(236) (1950) 433–460

2. http://www.unrealtournament.com/: Unreal tournament (2014)
3. Polceanu, M.: Mirrorbot: Using human-inspired mirroring behavior to pass a turing test. In: Computational Intelligence in Games (CIG), 2013 IEEE Conference on, IEEE (2013) 1–8
4. Jiménez, J.L., Mora, A.M., Fernández-Leiva, A.J.: Evolutionary interactive bot for the FPS unreal tournament 2004. In Camacho, D., Gómez-Martín, M.A., González-Calero, P.A., eds.: Proceedings 2st Congreso de la Sociedad Española para las Ciencias del Videojuego, Barcelona, Spain, June 24, 2015. Volume 1394 of CEUR Workshop Proceedings., CEUR-WS.org (2015) 46–57
5. Polceanu, M., Mora, A.M., Jiménez, J.L., Buche, C., Leiva, A.J.F.: The believability gene in virtual bots. In: Proceedings of the Twenty-Ninth International Florida Artificial Intelligence Research Society Conference, FLAIRS 2016, Key Largo, Florida, May 16-18, 2016., AAAI Press (2016) 346–349
6. Livingstone, D.: Turing's test and believable AI in games. Computers in Entertainment **4**(1) (2006) 6
7. Soni, B., Hingston, P.: Bots trained to play like a human are more fun. In: Neural Networks, 2008. IJCNN 2008. (IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on. (2008) 363–369
8. Yannakakis, G., Togelius, J.: A panorama of artificial and computational intelligence in games. Computational Intelligence and AI in Games, IEEE Transactions on (2014) Accepted for publication.
9. Laird, J.E.: It knows what you're going to do: Adding anticipation to a quakebot. AAAI 2000 Spring Symposium Series: Artificial Intelligence and Interactive Entertainment SS-00-02 (2000)
10. Choi, D., Könik, T., Nejati, N., Park, C., Langley, P.: A believable agent for first-person shooter games. In Schaeffer, J., Mateas, M., eds.: Proceedings of the Third Artificial Intelligence and Interactive Digital Entertainment Conference, June 6-8, 2007, Stanford, California, USA., The AAAI Press (2007) 71–73
11. Priesterjahn, S., Kramer, O., Weimer, A., Goebels, A.: Evolution of human-competitive agents in modern computer games. In: IEEE World Congress on Computational Intelligence 2006 (WCCI'06). (2006) 777–784
12. Small, R., Bates-Congdon, C.: Agent Smith: Towards an evolutionary rule-based agent for interactive dynamic games. In: IEEE Congress on Evolutionary Computation 2009 (CEC'09). (2009) 660–666
13. Schrum, J., Karpov, I., Miikkulainen, R.: Ut2: Human-like behavior via neuroevolution of combat behavior and replay of human traces. In: Computational Intelligence and Games (CIG), 2011 IEEE Conference on. (2011) 329–336
14. Soni, B., Hingston, P.: Bots trained to play like a human are more fun. In: IEEE International Joint Conference on Neural Networks, IJCNN'08. (2008) 363–369
15. Mora, A.M., Aisa, F., García-Sánchez, P., Castillo, P.Á., Guervós, J.J.M.: Modelling a human-like bot in a first person shooter game. IJCICG **6**(1) (2015) 21–37
16. http://pogamut.cuni.cz/main/: Pogamut - virtual characters made easy — about (2014)
17. Takagi, H.: Interactive evolutionary computation: Fusion of the capabilities of EC optimization and human evaluation. Proceedings of the IEEE (9) (2001) 1275–1296
18. Goldberg, D.E.: Genetic Algorithms in search, optimization and machine learning. Addison Wesley (1989)
19. Mora, A.M., Fernández-Ares, A., Merelo, J.J., García-Sánchez, P., Fernandes, C.M.: Effect of noisy fitness in real-time strategy games player behaviour optimisation using evolutionary algorithms. J. Comput. Sci. Technol. **27**(5) (2012) 1007–1023