# The Turing Test Track of the 2012 Mario AI Championship: Entries and Evaluation

Noor Shaker, Julian Togelius, Georgios N. Yannakakis,
Likith Poovanna, Vinay S. Ethiraj, Stefan J. Johansson, Robert G. Reynolds,
Leonard K. Heether, Tom Schumann, Marcus Gallagher

***Abstract*—The Turing Test Track of the Mario AI Championship focused on developing human-like controllers for a clone of the popular game *Super Mario Bros*. Competitors participated by submitting AI agents that imitate human playing style. This paper presents the rules of the competition, the software used, the voting interface, the scoring procedure, the submitted controllers and the recent results of the competition for the year 2012. We also discuss what can be learnt from this competition in terms of believability in platform games. The discussion is supported by a statistical analysis of behavioural similarities and differences among the agents, and between agents and humans. The paper is co-authored by the organizers of the competition (the first three authors) and the competitors.**

## I. Introduction

The creation of bots that play a game as well as possible has long been a main focus of research in the Computational Intelligence and Games community. Less emphasis has traditionally been given to constructing human-like agents, even though the literature contains several interesting attempts [1], [2], [3], [4]. Successfully imitating human behaviour is important for many reasons [5]. Several authors have argued that the appearance of human intelligence and human-likeness increases the quality of gameplay [6], [7], [8]. This goes both for enemies and collaborators or "sidekicks" in games, and for various demonstration modes. Another interesting direction is the construction of AI bots that act in a human-like manner when they stand to play on behalf of the player. These AI bots are important for several problems such as procedural content generation and player experience modeling [9], [10], [11] where these bots can be employed for training models and testing the quality of both learned models and generated content [12]. While we are far from a good understanding of how to achieve believability in bot behaviour, suggestions for characteristics of believable behaviour include variability [13], unpredictability and exhibiting non-repetitive behavior [14].

In the last few years, a number of game AI competitions have been run in association with major international conferences. Several of these competitions have spurred valuable research contributions as reported in [15], [16], [5], [17] (among others). Most of them focus on competitors submitting well-playing bots. One interesting exception is the *2k BotPrize*, where the submitted entries are not supposed to play the game as well as possible, but in an as human-like manner as possible [18].

We created the Turing Test Track within the Mario AI Championship to spur and benchmark development of believable bots. Our competition is obviously inspired by the 2k BotPrize, but differs from it both in the game domain chosen and in the evaluation procedure. To the best of our knowledge, ours is the first competition that focuses on believability within the platform game genre, and where believability is judged by spectators. Competitors participated in the competition by submitting an AI agent that is created to play as if it was controlled by a human player, and the human-likeness was assessed by human spectators. We use *Infinite Mario Bros*[1], a clone of the classic platform game *Super Mario Bros*, as a testbed for our competition. Our hope is that this competition will spur research in methods of creating believable bots for platform games. Many concerns relevant to designing human-like bots recur in the creation of the Non-Player Characters (NPCs) for other games such as first-person shooters and it is likely that principles and findings for generating believable agents carry over to other game genres.

The analysis and results we present in this paper are from the recent competition event held at CIG 2012 where three competitors participated and 73 subjects evaluated the submitted bots. This paper presents the first attempt towards analysing believability in platform games by presenting different techniques that can be employed to build believable agents and shedding some light on the attributes that contribute to our perception of a believable behaviour.

## II. What is Believability?

Lankoski and Björk [19] argue that a believable character is one that behaves consistently with its environment in the game, and that believable behaviour thus is context dependent. According to this definition, NPCs are often more believable than player characters, especially if the player character is e.g. running into walls or constantly jumping. This definition would be hard to apply to Super Mario Bros,

NS and JT are with the Center for Computer Games Research, IT University of Copenhagen, Denmark. GNY is with the Department of Digital Games, University of Malta, Malta. LP, VSE and SJ are with Blekinge Institute of Technology, Sweden. RGR and LKH are with the Department of Computer Science, Wayne State University, USA, TS and MG are with University of Queensland, Australia. emails: nosh@itu.dk, julian@togelius.com, georgios.yannakakis@um.edu.mt, {likithpoovanna.5238, s.e.vinay9986}@gmail.com, stefan.johansson@bth.se, {reynolds, lkinnaird}@cs.wayne.edu, thomas.schumann@uqconnect.edu.au, marcusg@uq.edu.au.

[1]http://www.mojang.com/notch/mario

as Mario is designed as a player character whose "natural" behaviour is to try to clear levels.

According to Togelius et. al. [20], believability can be viewed from two perspectives: *character believability* which defines the character/bot as being real, i.e. an actual human being and *player believability* which states that for a bot to be believable, someone should believe that the player controlling it is human. A similar distinction is proposed by Johansson [21] who differentiates between "believability" and "realism". For Johansson, believability defines the naturalness of a character behavior in terms of how far the actions the character take align with the player believes should happen. Realism is instead related to the appearance, animations, textures and similar aspects of the NPC.

A thorough treatment of the issue and resolution of the different concepts of believability would be both interesting and useful, but will have to wait until another time. In this paper, the term believability refers to *player believability* as defined in [20], as this concept most closely aligns with the original Turing test, and is identical to that used by Hingston in the 2k BotPrize [18], [5]. We are not directly concerned with the modelling or imitation of human cognitive abilities.

## III. Infinite Mario Bros and the Mario AI Championship

The testbed game used for the competition is a modified version of Markus Persson's *Infinite Mario Bros* which is a public domain clone of Nintendo's classical 2D platform game *Super Mario Bros*. The gameplay in Super/Infinite Mario Bros takes place on two-dimensional levels in which the player avatar (*Mario*) has to move from left to right avoiding obstacles and interacting with game objects. Mario can move left, right and duck. An additional two keys can be used to allow Mario to run, jump, or fire (depending on the state he is in). For more details about the game and our modifications the reader may refer to [22].

The Mario AI Benchmark was developed for and used in the *Mario AI Championship*[2], a series of competitions that have been running in association with international academic conferences on games and AI since 2009. The Mario AI Championship has four tracks: the *Gameplay track* [16], [23]; the *Learning track* [23]; the *Level Generation track* [22]; and the *Turing Test track*, the subject of the current paper.

It should be noted that Infinite Mario Bros, like Super Mario Bros, is a game where the seemingly best-performing playing style (in the sense of achieving the highest score or clearing the most levels) is not believable in any reasonable sense of the word. The Gameplay track of the 2009 competition was won by Robin Baumgarten with an agent built around the A* pathfinding algorithm [16],[3] which could clear all levels used in that year's competition; a video of that agent has been watched a million times on YouTube precisely because its playing style is so un-humanlike.

[2]http://www.marioai.org/

[3]http://www.youtube.com/watch?v=0s3d1LfjWCI

## IV. The Turing Test Track

While the Gameplay and Learning tracks focused on controllers that could play Infinite Mario as well as possible, and the Level Generation track focused on software that could design personalised levels for human players, the Turing Test track centred on constructing AI agents that could play Infinite Mario Bros in a human-like manner. The software designed for the learning track [23] is also used for the turing test track. The difference is in the evaluation procedure.

### A. Rules

The competition was open to individuals or teams from all over the world without any limitations. The main technical requirement was that the software should be able to interface to an unmodified version of the Mario AI Benchmark. All important information regarding the Mario AI Championship including rules and software is posted on a dedicated website. Prospective participants and other interested parties were encouraged to join a Google Group devoted to the competition[4].

### B. Competition Organization

Prior to the competition event, videos were recorded of the AI contestants and two human players playing three short levels of varying difficulty (difficulties 0, 1 and 2 in the benchmark software). Levels with difficulty zero contain only goombas (mushroom-like enemies that can easily be killed by stomping) with no gaps. The challenge of the levels increases when setting difficulty to 1 by introducing koopas and placing gaps. The most difficult levels are those of difficulty 2; these levels contain more enemies (mostly koopas) including flying enemies (which are harder to kill) and they also include more and wider gaps.

A web interface was designed that allowed viewing the recorded videos and collecting the judgments' evaluation. The videos are hosted on a web server and the audience of the CIG conference among others were invited through social networks (Facebook and Twitter) to judge the human-likeness of the controllers. A link to the evaluation interface was provided with instructions on the evaluation procedure. When accessing the evaluation web page, each judge was asked to compare two pairs of videos of gameplay sessions for different agents chosen randomly and presented in random order. After watching each pair of two videos, the judge was asked to answer a questionnaire.

### C. Software and interface

We used the standard Mario AI Benchmark representation of environments and actions. For more details, see [23].

1) The Environment interface describes the game state to the agent at each time step. The main types of information presented are:
   - A 22x22 array that describes the world around Mario with block resolution, and with Mario himself in the center. Figure 1 illustrates a small receptive field around Mario,

[4]http://groups.google.com/mariocompetition

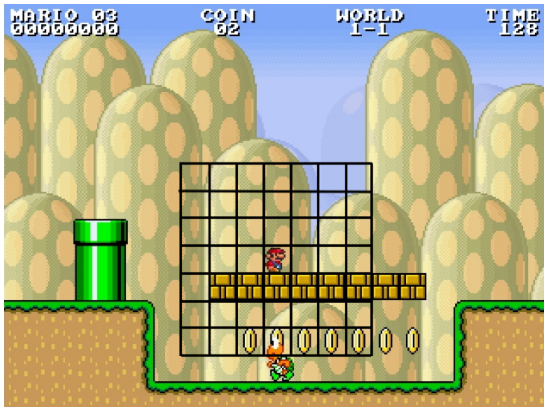


Fig. 1. Mario grid representation. Both enemies and level grids contain the information from the same area around Mario which appears marked.

- Exact positions of enemies on screen,
- State information: whether Mario is currently in Fire mode, on the ground etc.

2) The Agent interface: This is the only interface that needs to be implemented in order to create a functional Mario-playing agent. The key method here is getAction, which takes an Environment as input and returns a five-bit array specifying the action to perform. This yields a total of $2^5 = 32$ actions.

### D. Evaluation Procedure

Believability can be assessed from a first- or third-person perspective. An example of a first-person assessment of believability is the *2k BotPrize* where the assessment is presented as part of the game; subjects are equipped with a special weapon that can be used in-game to distinguish between an AI bot and a human opponent [5].

In Infinite Mario Bros, first-person assessment is not possible since we have only one player character in the game. Moreover, assessing believability from a first-person perspective requires the judge to pay attention to the playing styles of agents while being engaged in the gaming experience, which is distracting. Therefore, in this paper, we follow the third-person assessment approach [20]. We chose to let the judges observe the game for an average duration of one minute, which in the organisers' opinion is enough to get an idea of the playing style of the player. We implemented a subjective assessment method where post-experience questionnaires are presented to observers. The questionnaires are presented after watching a pair of videos, with each video depicting one player. The following two questions were asked:

- Which do you think plays in a more human-like manner?
- Which do you think is more expert?

The possible answers were: *Video A*, *Video B*, *both equally* and *none of them*, following the 4-AFC protocol [24].

## V. The Competitors

Three bots and two human players competed in the Turing Test track. The two human players are chosen based on their expertise in the game. One is a skilled player with considerable experience of the game, and the other is a novice with little experience of this game. In the following sections, we present the participating controllers. Each section is written by the authors of the controller.

### A. Likith Poovanna, Vinay Sudha Ethiraj and Stefan Johansson

*1) Idea and Architecture:* The VLS bot (named after the first names of its contributors Vinay, Likith, and Stefan) is an artificial potential field-based bot that plays Infinite Mario Bros. Artificial potential fields (APFs) is a reactive technique that lets a unit make its decisions based on the local impacts of attracting and repelling forces in its surroundings. Based on the impacts on a number of lookahead positions, the unit chooses the action that will take it to the most attractive position. The technique originates from the field of robotics [25], and is also related to the type of influence maps described e.g. by Tozour [26]. For a closer description of the two techniques and their use in game AI, see [27]. In all APF-based applications, there are a number of things that need to be defined, such as the lookahead positions, the potential fields and the sources of potentials in each field.

*2) Development:*

*a) Lookahead positions::* In each frame, Mario looks at each possible move, and where that move would take him. The resulting states are then considered as lookahead positions. Note that in the case of Mario, the actions may be of different duration in time, e.g. a move to the right is instant, whereas a jump takes several frames to execute.

*b) Artificial potential fields::* By using different fields for different objectives in the game, it becomes easier to tune the balance of the solution. In VLS, we use four fields:

- *The field of progression* is slightly more attractive to the right than to the left, making Mario prefer going in the right direction. The more attractive the right side is, the harder it will be for other attractive sources to gain attention in the choice of action.
- *The field of rewards* makes coins, mushrooms, blocks, and flowers attractive. The stronger this field is, the more eager Mario will be to collect rewards.
- *The field of opponents* makes the positions of the opponents repelling. The more repelling they are, the harder Mario will try to avoid their positions. The exception is jumps that land on the monsters to kill them.
- *The field of terrain* will make sure that Mario avoids gaps. It also identifies dead ends and makes these areas slightly repelling, so that Mario turns to search for another path. A slight reward is given for height, making him want to jump up on platforms when applicable.

*c) The humanization of VLS::* To make the bot play as humanly as possible, we first hand-tuned the parameters of the solution (see Table I) to obtain a reasonably well playing bot. We then set up an experiment in the following way:

1) Initially we picked five human players with various playing styles, which we recorded while they played a specific Mario level.

TABLE I
THE PARAMETERS TUNED IN THE EXPERIMENTS. $v_1$-VALUES ARE THE ORIGINAL VALUES AND $v_2$-VALUES ARE THE RESULTS OF THE TUNING.

| Name | Detailed description | $v_1$ | $v_2$ |
|---|---|---|---|
| Coins | The attraction of coins | 1 | 1.061 |
| blocks | The attraction of block-rewards. | 1.1 | 1.33 |
| Platforms | The attraction of platforms. | 0.8 | 0.9522 |
| Enemies | The likeliness of the bot trying to kill the enemy (rather than avoiding it). | 0.75 | 0.85 |
| Pipes | The approximate distance at which to react on the pipe. | -0.5 | -0.832 |
| Gaps | The distance that the bot keeps to the gap it is trying to cross. | -0.3 | -0.5866 |
| Dead ends | A parameter for how far the human player would move before realizing it is a dead end and turn. | -3 | -3.093 |

2) Then each of these players was asked to select the two videos that they thought expressed the most human-like behavior based on pair-wise comparisons of the recorded games.
3) The two human players whose videos were selected in the previous stage are asked to play the game again. Each player was assigned to play different levels while the data was recorded. These videos are the basis of our tuning.
4) In the tuning, we let 30 test persons each watch two games, one played by our bot while the other is the recording of a human player.
5) The testers are then asked to fill in a form with one question for each parameter presented in Table I: "Which player was better at collecting coins?", or "Which player was better at crossing gaps?". Answers were given on an ordinal 11 step scale.
6) The results where then compiled to averages, $a$, between 0 and 1 and compared with the hand-coded values. The $v_2$-values were calculated simply as the value that would level out the differences, i.e. $v_2 = v_1/2a$.

*3) Evaluation:* The performance of the tuned parameters was validated in a second series of experiments (similar approach but using the $v_2$ values), yielding a result where only one parameter showed a significant difference (VLS was significantly better at jumping on platforms). The rest of the cases showed small or no differences between the human and the bot as measured by the selected factors.

*4) Strengths and Weaknesses:* The strength of the controller lies in its modularity. It is easy to add new fields, representing new objectives, and it is relatively easy to tune (e.g. to the skill level of a specific human style of playing). The drawback is of course that it is fully reactive and thus hard to tune in domains that require extensive planning. However, through the use of A* as a distance measure in the field of progression (instead of e.g. the Euclidean measure), this issue may be addressed, see e.g. Hagelbäck [28]. This is (to our knowledge) the first documented use of potential fields as a way to control characters in a platform game.

### B. Tom Schumann and Marcus Gallagher

*1) Idea and Architecture:* TomAgent was built with the intention of only using rule-based techniques [29]. It uses a nearest neighbor classifier with Hamming distance on the level area surrounding the controller [30]. This view was then classified as one of a set of saved views. Each of these views has associated sets of actions for the controller. There were different sets of actions for each direction and player mode. Influence maps were used to determine which direction was most desirable for the player. The idea of using influence maps came from the work of Wirth [31], Teed [32] and Collett [33]. Although they applied the method on Ms Pac-Man, the technique generalized well to Infinite Mario Bros. The best direction to follow and the current player mode were used to choose the action to be performed. Additional desired behaviors such as shooting at enemies and intermittent pausing supplemented these techniques [29].

*2) Development:* A set of saved level views was assembled manually by testing the controller on random levels. Whenever the controller got stuck during a test, the current level view was added to the set of saved views along with sets of actions to properly navigate that view. This technique generalizes well to unseen situations with similar characteristics to saved views to allow correct classification. As more views are added to the set, the occurrence of misclassifications increased which sometimes resulted in the controller performing incorrect actions and being unable to progress. To mitigate this, additional tests were added to determine if the controller is stuck and a jump action is performed to move away from the problematic situation.

The nearest neighbor classifier was accurate at selecting correct actions for the controller. To make the controller less expert, extra sets of ineffectual actions were added. These sets include actions such as not jumping to the correct height or jumping early/late resulting in inefficient navigation. These sets of actions had a low probability of being chosen but prevented the agent from playing perfectly which is usually considered as a non-human behavior [20].


Fig. 2. Influence mapping around the controller. Backwards is the better direction.

The influences of different entities were roughly prioritised based on their perceived benefit to the controller [29]: the Princess (which marks the end of the level) was given the highest priority as reaching her means winning the level; power-ups were given the next highest priority as they would replenish Mario's health; the areas around enemies come next to encourage the controller to stomp on them while

coins were given the lowest priority as they only gave points. Enemies themselves were given negative influences to discourage contact. Flying and spiky monsters were given much lower priorities and had extra checks so that they could be bypassed without causing penalty.

*3) Evaluation:* The correctness and effectiveness of the nearest neighbour classifier and influence maps were evaluated by testing the controller on random levels. This form of evaluation demonstrated that the techniques were reasonably effective as the controller was able to competently complete most random levels it was tested on.

The controller's ability to play like a human was assessed by testers who were asked to watch a video of it and another one for a human player playing the same level and report which they thought was more human and why, similar to the evaluation procedure of the Turing Test Track. The feedback collected was used to improve the sets of actions and to tweak other aspect of the controller behaviour.

*4) Effectiveness:* The nearest neighboor classifier was effective as it easily allowed the controller to correctly navigate levels. The influence maps were also effective as they made the controller back-track and pick up coins and power-ups. On rare occasions, the influence maps would conflict and cause the controller to get stuck and we added checks to handle this case. To minimise the occurrence of conflicting influences the influence radii were kept small, but this meant that the controller would sometimes "forget" about entities it was seeking when the distance between them was large.

*5) Generalizability:* Both techniques used could be generalised to other 2D platform games. Influence maps can generalize well as they are reasonably abstracted from any sort of game-play and are only used to indicate the desirability of areas adjacent to the player. Using a nearest neighbour classifier for navigation could also be used in other 2D platform games, but its efficiency is limited to situations that require complex navigation.

### C. Robert Reynolds and Leonard Heether

*1) Idea and Architecture:* WSU-Mario-CAT (WSU-M-C) is a client designed for the Mario Gameplay and Turing Test tracks. It was constructed using an artificial neural network and trained using Cultural Algorithms [34]. Since most human video game players see game completion as the ultimate goal of any video game, the client was trained with finishing a level as the primary goal.

*2) Development:* The WSU-M-C controller is designed to take input, in the form of screen data, and provide appropriate output based on that data. The controller accomplishes this by passing this data through an artificial neural network. The internal weights between the layers of the neural network are decided by the Cultural Algorithm training process.

The neural network used by the WSU-M-C controller is an Elman-type neural network, and consists of an input layer, a directly-linked hidden layer, a recurrent hidden layer, and an output layer. The input layer takes in screen data in a grid based format and assigns values to the contents of the each grid cell. An example of this can be seen in Figure 3, however our updated controller consisted of 119 nodes on the input layer, and 25 nodes on each hidden layer. The increase in nodes corresponds to an increase in the size of the screen grid (from 3x3 to 9x13.)

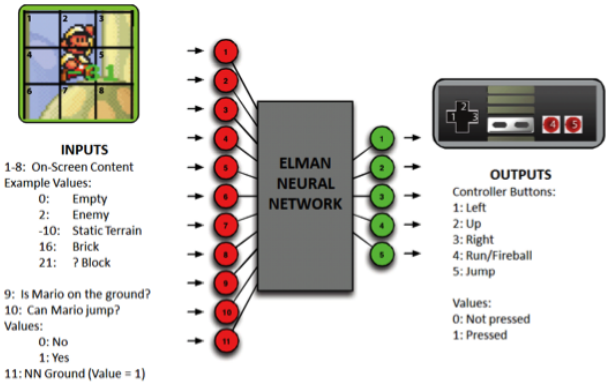


Fig. 3. WSU-Mario-CAT Overview.

*3) Training:* The controller training process involves determining a set of weights for all the connections in the neural network and then evaluating that set of weights based on a multi-objective fitness function. During the learning process, these weights are adjusted by the Cultural Algorithm framework to improve the overall fitness as defined by this multi-objective fitness function.

An example of the learning process can be seen in Figure 4 as a function of the overall fitness, of the best case in a given generation. This figure also defines several key learning points. These learning points demonstrate how the system learns to accomplish key tasks. There are clear parallels to how inexperienced human players learn to play platform style video games. This incremental learning path has been demonstrated before in agents trained using Cultural Algorithms [15], [35].

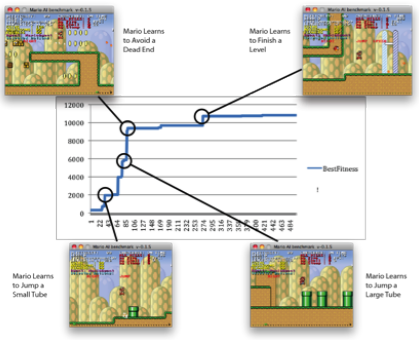


Fig. 4. Visual Example of the Mario Learning Process.

*4) Evaluation:* In creating the WSU-M-C agent for the competition, it was tested against a multiple level combinations. These level combinations involved different level lengths, types (above ground/underground) difficulties and time limits. After the training period was over, the best derived weight set was hard coded into the agent prior to

submission. This weight set defined the agent's behavior during the competition.

*5) Strengths and Weaknesses:* The development of the WSU-M-C agent has brought to light some interesting conclusions. The incremental learning path, demonstrated by the agent, corresponds to the typical learning path shown by novice human players. This human-like learning theoretically can be used to create a better agent, more suitable for a Turing test that looks specifically for human-like behavior. However, due to the fact that the training process focuses heavily on level completion, the agent can exhibit behavior that, while very conducive to level completion, is not necessarily viewed as human-like.

In the future, we believe that this system could be improved by using a "network of networks" rather than a single neural network. This would provide more flexibility to approach different situations. Using the Cultural Algorithm framework, one could train multiple different networks for various in-game applications at the same time. In addition to training the individual networks, the Cultural Algorithm framework could be used to learn which network would be best to use in a given situation. It would also be intriguing to see if what has been learned in the Mario AI spectrum could be transferred to other platform games that use similar input/output schemas. We believe that since there are many situations in platform games that are similar (i.e. jumping, running, collecting) that any learning agent developed for one platform game, could be used for another platform game with a minimal amount of adjustment.

## VI. Results and Analysis

The human-likeness of the submitted agents and the two human players was assessed by 73 participants who took part in the evaluation process. In this section, we present the results and analysis conducted to investigate the differences between the agents and how those affect believability.

### A. Results

Table— II presents the final results of the competition. Each agent was presented 58 times on average. The score for each agent is calculated as the percentage of times the particular agent scores higher than another agent when played in pair over the total number of times this agent was presented. We chose to randomly select the agent and present them in pairs since we are interested in comparing the techniques used with each other as well as with actual human. Since random selection is used, there was no guarantee that all agents would be evaluated an equal number of times and it was hard to know in advance whether we would have enough participants to ensure a uniform distribution of votes along all agents. Therefore, in order to allow a fair comparison, all values obtained are normalised and the score for each agent, $i$, is calculated according to the equation:

$$Score_i = \frac{times\_selected_i / times\_presented_i}{\sum_{a=1}^{agents} times\_selected_a / times\_presented_a}$$

As can be seen from Table II, the winner of the competition was the VLS agent, with a considerable margin to the other agents. This agent managed to convince 34 observers that it was more human than other agents out of the 68 times this agent was presented. The VLS bot is also the bot that comes closest to the human vote baseline of the novice human player with around 5% difference. The other two agents, on the other hand, fail to convince the majority of the observers of being controlled by a human when compared to other agents. The results show that 16 voters out of 54 believed that TomAgent is the more human-like while the WSU-M-C agent was perceived as more human-like in 6 times only out of the 65 times this agent was compared. It is worth noticing that the novice human player received the most human votes (27/45) while the expert human received more computer votes than the VLS bot.

The *Mann-Whitney U Test* ($p < 0.01$) was applied to the results in order to calculate the significance of the difference between each pair of controllers. In order to apply this test, the ranks are calculated as zeros and ones, i.e. when comparing the VLS agent with TomAgent, for example, the VLS was selected nine times and TomAgent five times, then VLS gets nine ones and five zero value, and the same applies for the rest of controllers compared to VLS. Note that to perform this test, only pairs with clear preference were considered (the pairs where the spectators clearly reported whether the first or the second agent is more human-like) while other pairs are removed. This resulted in 37, 56, 48, 42, 36 pairs of clear preferences for the TomAgent, the VLS, the WSU-M-C, the expert and the novice human player, respectively. The results of this test are presented in Table III.

As expected, the novice human player is significantly different from all AI controllers except the VLS agent. In the comparison among the AI agents, WSU-M-C and VLS are, respectively, the least and most human-like AI agents; however, there is not a significant difference between VLS and TomAgent. Somewhat unexpectedly, the expert human player has no significant difference from any agents except for the WSU-M-C agent.

The competition results illustrate the difficulty in assessing believability even in a game as seemingly simple as Super Mario Bros, with low control bandwidth, simple graphics and easy overview of the play area. The results suggest that it is easier to imitate the behaviour of an expert than a beginner player and that expert players are more likely to be mistaken for being an AI bot. In order to further investigate the results obtained, we decided to mine the game logs for relationships between the controllers' playing styles and perceived believability.

### B. Feature Analysis

Several gameplay statistics were calculated from 60 levels of low difficulty (difficulty is set to 0) generated and played by the agents (20 levels for each agent) and 10 levels of the same difficulty played by an expert and a novice human players. The features are the following: the percentage of time spent jumping, ducking, running, moving left, moving

TABLE II
THE RESULT OF THE TURING TEST TRACK FOR THE 2012 MARIO AI CHAMPIONSHIP

| Name | Agent | Presented | Selected | Score |
|---|---|---|---|---|
| Satish, Ethiraj and Johansson | VLS | 68 | 34 | 25.79 % |
| Schumann and Gallagher | TomAgent | 54 | 16 | 15.28 % |
| Heether and Reynolds | WSU-M-C | 65 | 6 | 4.76 % |
| Expert human | — | 60 | 27 | 23.21 % |
| Novice human | — | 45 | 27 | 30.95 % |

TABLE III
THE P-VALUES OBTAINED FROM APPLYING THE *Mann-Whitney U Test* ON EACH PAIR OF CONTROLLERS. THE STATISTICALLY SIGNIFICANT DIFFERENCES ($p - value < 0.01$) ARE PRESENTED IN BOLD.

| Agents | Expert | Novice | VLS | Tom | WSU-M-C |
|---|---|---|---|---|---|
| Expert | – | 0.31 | 0.72 | 0.06 | $\mathbf{4.35 * 10^{-7}}$ |
| Novice | – | – | 0.16 | **0.006** | $\mathbf{8.2 * 10^{-9}}$ |
| VLS | – | – | – | 0.10 | $\mathbf{5.44 * 10^{-7}}$ |
| Tom | – | – | – | – | **0.001** |

right, and standing still out of the total amount of time spent playing each level, the percentage of coins collected out of the total number of coins present in the level, the percentage of all blocks smashed, the percentage of all enemies killed and the percentage of all enemies killed by stomping.

Figure 5 presents a comparison between the average and standard deviation values of the features extracted. All feature values are uniformly normalised along all sessions played by all controllers to the range [0,1] using max-min normalisation. As seen from Figure 5, there are remarkable differences between the behaviours observed. The three agents differ clearly in terms of time spent jumping, ducking, running, standing still, amount of coins collected and blocks destroyed, number of enemies killed and percentage of enemies killed by stomping.

The most believable agent, VLS, appears to be the one that spends time moving left, collects most coins, and succeeds in killing many enemies mostly by stomping. The least human-like agent, WSU-M-C, spends a lot of time running in the right direction, performs a lot of unnecessary ducking actions (note that these levels contain no bill blasters nor flying enemies which are the main reason for ducking), collects fewer coins and kills fewer enemies. For the third agent, TomAgent, who achieved an average score, the features extracted show that this agent exhibits similar behaviour to the VLS agent in terms of collecting coins and the percentage of time spent ducking while the main differences between these two agents are in the percentage of time spent jumping/running and the number of enemies killed. When comparing this agent to the WSU-M-C agent, the results illustrate dissimilarities along the proportion of time running/jumping and standing, the number of coins collected and the percentage of enemies killed by stomping (since we observed similar percentages of enemies killed between these agents, this suggests that this agents mostly kills enemies by shooting fire).

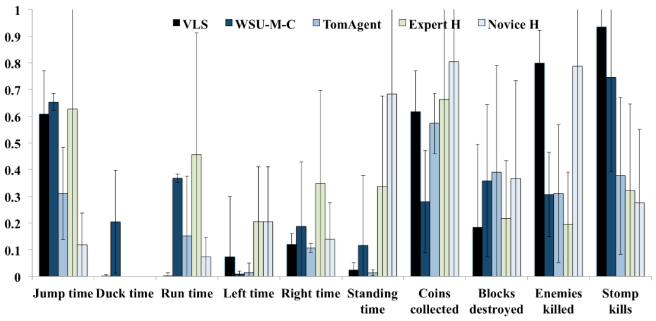


Fig. 5. Average and standard deviation values of several statistical features extracted from gameplay sessions.

The novice and expert players show similar behaviour in terms of interaction with items as can be seen from the number of coins collected, the number of blocks destroyed and the enemies killed by stomping. These two players differ mainly in the amount of time spent jumping, running and standing and the number of enemies killed.

The results show that the three agents show distinctive behaviour compared to the novice and expert human players, although there seems to be a number of similarities along some dimensions. The winning agent, VLS, appears to imitate the style of the expert player when it comes to amount to time spent on movement actions and collecting items, while dissimilarity has been obtained when interacting with enemies.

The above-mentioned behavioural characteristics help us draw a preliminary picture of what contributes to a believable behaviour. Unnecessary ducking appears to be linked to undesirable behaviour while humans seem to spend a reasonable amount of time of standing, switching direction, collecting items and stomping on enemies.

## VII. CONCLUSIONS

In this paper, we presented the turing test track of Mario AI Championship 2012. We described the competition organisation, rules and the submission interface and we further discussed the architecture and techniques followed by competitors to construct AI agents that play our testbed game, Infinite Mario Bros, in a human-like manner. We described the voting interface and the scoring procedure followed to assess believability. Finally, we presented the results of the competition and we run statistical analysis and conducted an experiment to help us understand the factors that contribute to believability in 2D platform games.

The turing test track was run for the first time in 2012 and this paper is the first to describe it. Accordingly, there are a number of lessons that can be learned and many future directions that can be pursued. In general, it appears that assessing believability is not an easy task. The results showed that a human player with an expert playing style can easily mislead voters into being an AI bot while a beginner human is easier to identify correctly. It also appears that

constructing an AI bot that imitates expert behaviour is easier than imitating a beginner style.

The analysis presented in this paper focused on examining the similarities and differences between the AI agents based on several statistical features of gameplay. There are many small differences, but one that stands out is that humans tend to stand still and think now and then; AI agents don't. There were also differences between the different agents, and it interesting to note the agent that was trained to reach the end of the level as quickly as possible performed worst it in terms of human-likeness. A more thorough analysis can and will be performed to test correlations between gameplay features and reported preferences that helps better understand the factors that contribute to believable behavior.

Although gameplay features represented as frequencies of actions give an indication of players' behaviour, combining these features with context information provides a better alternative for understanding believability in situational context. To this end, sequence mining techniques can be used to extract multimodal patterns that combine information from players' behaviour and game content [11] and correlation analysis can be used to relate these features to reported believability.

The data collected and the experiments conducted can potentialluy be used to construct models of believability; levels and gameplay features can be fed into a pairwise preference model that can be trained to predict the human-likeness of an agent based on its individual playing style. The features that correlate the most with believability can be extracted and a similar framework to the one followed in [10], [11] for player experience modelling can be followed to train and evaluate the believability models.

## REFERENCES


[1] B. Gorman, C. Thurau, C. Bauckhage, and M. Humphrys, "Believability testing and bayesian imitation in interactive computer games," *From Animals to Animats 9*, pp. 655–666, 2006.

[2] J. Laird and J. Duchi, "Creating human-like synthetic characters with multiple skill levels: A case study using the soar quakebot," *Ann Arbor*, vol. 1001, pp. 48 109–2110, 2000.

[3] V. Tatai and R. Gudwin, "Using a semiotics-inspired tool for the control of intelligent opponents in computer games," in *International Conference on Integration of Knowledge Intensive Multi-Agent Systems*. IEEE, 2003, pp. 647–652.

[4] J. Ortega, N. Shaker, J. Togelius, and G. Yannakakis, "Imitating human playing styles in super mario bros," *Entertainment Computing*, 2012.

[5] P. Hingston, "A new design for a turing test for bots," *IEEE Transactions on Computational Intelligence and Games (CIG)*, 2010.

[6] A. Champandard, *AI game development*. New Riders Publishing, 2003.

[7] C. Bateman and R. Boon, *21st century game design*. Charles River Media Hingham, MA, 2006.

[8] D. Weibel, B. Wissmath, S. Habegger, Y. Steiner, and R. Groner, "Playing online games against computer-vs. human-controlled opponents: Effects on presence, flow, and enjoyment," *Computers in Human Behavior*, vol. 24, no. 5, pp. 2274–2291, 2008.

[9] C. Pedersen, J. Togelius, and G. Yannakakis, "Modeling player experience for content creation," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 2, no. 1, pp. 54–67, 2010.

[10] G. Yannakakis, M. Maragoudakis, and J. Hallam, "Preference learning for cognitive modeling: a case study on entertainment preferences," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, vol. 39, no. 6, pp. 1165–1175, 2009.

[11] N. Shaker, G. Yannakakis, and J. Togelius, "Crowd-sourcing the aesthetics of platform games," *IEEE Transactions on Computational Intelligence and AI in Games*, 2013.

[12] N. Shaker, M. Nicolau, G. Yannakakis, J. Togelius, and M. ONeill, "Evolving levels for super mario bros using grammatical evolution," *IEEE Transactions on Computational Intelligence and Games (CIG)*, 2012.

[13] R. Wray and J. Laird, "Variability in human behavior modeling for military simulations," in *In Proceedings of Behavior Representation in Modeling and Simulation Conference (BRIMS*. Citeseer, 2003.

[14] J. Schaeffer, V. Bulitko, and M. Buro, "Bots get smart," *Spectrum, IEEE*, vol. 45, no. 12, pp. 48–56, 2008.

[15] D. Loiacono, P. Lanzi, J. Togelius, E. Onieva, D. Pelta, M. Butz, T. Lnneker, L. Cardamone, D. Perez, Y. Sáez, *et al.*, "The 2009 simulated car racing championship," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 2, no. 2, pp. 131–147, 2010.

[16] J. Togelius, S. Karakovskiy, and R. Baumgarten, "The 2009 mario AI competition," in *Proceedings of the IEEE Congress on Evolutionary Computation*. Citeseer, 2010.

[17] D. Perez, P. Rohlfshagen, and S. Lucas, "The physical travelling salesman problem: Wcci 2012 competition," in *Evolutionary Computation (CEC), 2012 IEEE Congress on*. IEEE, 2012, pp. 1–8.

[18] P. Hingston, "A turing test for computer game bots," *Computational Intelligence and AI in Games, IEEE Transactions on*, vol. 1, no. 3, pp. 169–186, 2009.

[19] P. Lankoski and S. Björk, "Gameplay design patterns for believable non-player characters," in *Situated Play: Proceedings of the 2007 Digital Games Research Association Conference*, 2007, pp. 416–423.

[20] J. Togelius, G. Yannakakis, S. Karakovskiy, and N. Shaker, "Assessing believability," in *Believable Bots: Can Computers Play Like People?*, P. Hingston, Ed. Springer, 2012.

[21] A. Johansson, "Affective decision making in artificial intelligence: Making virtual characters with high believability," Ph.D. dissertation, Linköping, 2012.

[22] N. Shaker, J. Togelius, G. Yannakakis, B. Weber, T. Shimizu, T. Hashiyama, N. Sorenson, P. Pasquier, P. Mawhorter, G. Takahashi, *et al.*, "The 2010 mario ai championship: Level generation track," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 3, no. 4, pp. 332–347, 2011.

[23] S. Karakovskiy and J. Togelius, "The mario AI benchmark and competitions," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 4, no. 1, pp. 55–67, 2012.

[24] G. N. Yannakakis and J. Hallam, "Real-time Game Adaptation for Optimizing Player Satisfaction," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 1, no. 2, pp. 121–133, June 2009.

[25] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The international journal of robotics research*, vol. 5, no. 1, pp. 90–98, 1986.

[26] P. Tozour, "Influence mapping," *Game programming gems*, vol. 2, pp. 287–297, 2001.

[27] S. J. Johansson, "The use of artificial potential fields and influence maps in game AI research," in *IEEE Transactions on Computational Intelligence and AI in Games*, 2013.

[28] J. Hagelbäck, "Multi-agent potential field based architectures for real-time strategy game bots," 2011.

[29] T. Schumann, "Using nearest neighbour and influence maps to create a human like agent to play infinite mario brothers," Ph.D. dissertation, 2012.

[30] S. Pogadaev, "Using nearest neighbour and genetic algorithms to evolve an infinite mario bros. agent," Ph.D. dissertation, 2011.

[31] N. Wirth and M. Gallagher, "An influence map model for playing ms. pac-man," in *IEEE Symposium On Computational Intelligence and Games*. IEEE, 2008, pp. 228–233.

[32] B. Teed, "Building an influence maps based agents ms pacman," Ph.D. dissertation, 2011.

[33] C. Collett, "Combining techniques used by intelligent rule-based agents to play ms. pacman," Ph.D. dissertation, 2010.

[34] R. Reynolds, "On modeling the evolution of hunter-gatherer decision-making systems," *Geographical Analysis*, vol. 10, no. 1, pp. 31–46, 1978.

[35] ——, "Networks do matter: The socially motivated design of a 3d race controller using cultural algorithms," *International Journal of Swarm Intelligence Research (IJSIR)*, vol. 1, no. 1, pp. 17–41, 1993.