An Analysis of Artificial Intelligence Techniques in Multiplayer Online Battle Arena Game Environments

Michael Waltham
CSIR Meraka Centre for Artificial Intelligence Research (CAIR)
University of KwaZulu-Natal, Westville Campus, Durban, South Africa
walthammichael@gmail.com

Deshen Moodley
CSIR Meraka Centre for Artificial Intelligence Research (CAIR)
University of Cape Town, Cape Town, South Africa
deshen@cs.uct.ac.za

ABSTRACT

The 3D computer gaming industry is constantly exploring new avenues for creating immersive and engaging environments. One avenue being explored is autonomous control of the behaviour of non-player characters (NPC). This paper reviews and compares existing artificial intelligence (AI) techniques for controlling the behaviour of non-human characters in Multiplayer Online Battle Arena (MOBA) game environments. Two techniques, the fuzzy state machine (FuSM) and the emotional behaviour tree (EBT),were reviewed and compared. In addition, an alternate and simple mechanism to incorporate emotion in a behaviour tree is proposed and tested. Initial tests of the mechanism show that it is a viable and promising mechanism for effectively tracking the emotional state of an NPC and for incorporating emotion in NPC decision making.

Keywords

Artificial Intelligence; 3D Games; MOBA

1. INTRODUCTION

Artificial intelligence (AI) is an extremely diverse area of Computer Science that continues to expand its applications in the real world. AI in Computer Science is focused on allowing computers to act with a degree of intelligence [11]. AI, when applied to the field of 3D game development, allows non-player characters (NPCs), *i.e.* computer controlled players, to effectively challenge human players by giving them a certain degree of "intelligence" with respect to a particular game environment.

The Multiplayer Online Battle Arena (MOBA) style of game was chosen simply because it has recently become one of the most popular 3D game genres [3, 8, 10]. The game development process can be quite time consuming and tedious [6, 7]. Third-party game engines e.g. The Unity Game Engine or CRYEngine are usually used to develop games. For this reason, the Unreal Engine 4 (UE4) was used to develop the test game environment that was used during technique evaluation.

The purpose of this study was to provide a review and analysis of AI techniques that may be used for developing stochastic, autonomous agents, *i.e.* NPCs, within a MOBA style game environment. These techniques were analysed using the literature for usage within MOBA environments with respect to: stochastic behaviour support, resource consumption and modularity. Two techniques were selected for further comparison after a review of the literature was completed. These techniques were the fuzzy state machine (FuSM) and the emotional behaviour tree (EBT), which derive from the popular finite state machine and behaviour tree respectively. These techniques were then implemented to control the behaviour of agents within the test game environment and their performances were compared.

This paper firstly outlines common behaviour control techniques used in 3D game development as well as provide a review of these techniques (section 2). An outline of the game scenario is then discussed (section 3). The experimental design and technique implementation is then described (section 4). Lastly, the performance results of each implemented technique is presented and discussed (section 5).

2. LITERATURE REVIEW

The following section identifies and describes five artificial intelligence techniques that are commonly used to control the behaviour of NPCs within game environments. The advantages and disadvantages of using each technique is then listed. These advantages and disadvantages relate to aspects such as: stochastic behaviour support, resource consumption and modularity.

2.1 Techniques

2.1.1 The Finite State Machine

The most popular AI technique used in game development has, up until now, been the finite state machine (FSM) [12, 15]. It contains a number of behaviour-defining states that describe the current situation of an agent. Transition logic is inserted into states to allow the agent to move from one state to another when various conditions are met. Inherently, this may lead to issues when there are a large number of behaviours to consider. This poses a major issue with

SAICSIT '16, September 26-28, 2016, Johannesburg, South Africa

DOI: http://dx.doi.org/10.1145/2987491.2987513

the finite state machine in that it does not scale well with large systems [5, 10, 16]. The FSM has, however, proven to be a relatively simple technique to implement within game environments which is one of the reasons why it has been extremely popular [15].

Modern commercial games use high amounts of resources on aspects such as game logic and graphics processing, leaving very little resources for artificial intelligence [13]. The FSM inherently has a very low computational cost [12, 15]. This, together with its simplicity, proves to be a major advantage in using the FSM for games in which the focus of the game is not centred around NPCs. Therefore, the FSM can be an effective solution for games that do not require advanced, adaptive agents [15]. Games in which the FSM has been successful include: Age of Empires (Ensemble Studios) and Quake (Id Software).

2.1.2 Fuzzy Logic

Fuzzy logic is an extension of conventional boolean logic that accounts for partial truth values. It allows developers to work with partial or incomplete information [15]. Many researchers believe that fuzzy logic is similar to the way in which humans perform reasoning [12].

In the field of game development, agents may alter their decisions based on the values of fuzzy variables. It is notable that fuzzy logic is a relatively popular artificial intelligence technique for game development and has been used in several commercial games [9, 13].

Non-programmers are able to assist in the design process due to the fact that fuzzy logic simply requires knowledge of boolean logic. The extremely low learning curve of fuzzy logic proves to be a major advantage. Experts in the particular style of game, who may have no knowledge of how the game is actually developed, are able to write fuzzy rules for the agents to follow. This therefore will produce seemingly intelligent agents with the advantage of a simple design. The disadvantage of this is that if experts in the field cannot be found, fuzzy rules may be time-consuming to develop [12].

2.1.3 The Fuzzy State Machine

The finite state machine may be incorporated with fuzzy logic to produce a fuzzy state machine (FuSM). The result of this is a finite state machine with a degree of non-determinism [15]. Commercial games such as Unreal (Epic Games) successfully made use of the FuSM to control NPC behaviour. Certain advantages of the FuSM are listed below:

- The fuzzy state machine does not require as many states as the finite state machine while still providing less predictable behaviour [2].
- Finite state machines can easily be converted to fuzzy state machines [12].
- The range of possible responses from characters is increased [15].

2.1.4 Behaviour Trees

The behaviour tree is a directed acyclic graph that can contain various types of nodes [4, 5]. Behaviour trees have become a very popular form of NPC behaviour control in game development [5]. This is evident due to the fact that commercial games such as Grand Theft Auto (Rockstar Games) and Halo (Bungie) employ the use of behaviour trees to control NPC behaviour [4, 5].

Behaviour trees consist of various types of nodes to form a tree structure. Each node may have multiple parent nodes as this allows for different parts of the tree to be reused. When the tree is traversed, each node is able to return a value indicating the result of their execution. The return types are as follows: success, failure and running.

Leaf or terminal nodes may either be an action or a condition node. An action node represents a specific task or process that the agent may carry out. A condition node simply checks if a certain condition holds at any given point in execution.

The four types of non-leaf nodes are sequence, selector, parallel and decorator. A sequence node executes its children sequentially until a child returns failure. A selector node executes its children sequentially until a child returns success. A parallel node will execute all children in parallel, the stopping criteria for this node depends on the particular behaviour tree implementation. A decorator node may contain only one child node and is able to prevent the execution of the child until a certain condition is met. The advantages of using behaviour trees have been listed below [5, 9, 10, 16].

- Behaviour trees scale well with a system.
- Previously defined trees are easily re-used.
- Easy to debug.
- Memory efficient.
- Additional functionality can easily be added.
- State transition logic does not need to be coded into the behaviour.
- Different areas of an agent's behaviour can easily be kept separate in the tree.
- Commercial game engines such as the Unreal Engine 4 and CryENGINE have built-in support for behaviour trees. This gives developers quick access to a graphical user interface environment in which they can develop NPC behaviour. As mentioned in [9], behaviour trees are highly effective when used with a graphical user interface.

2.1.5 Emotional Behaviour Trees

Emotion is an important factor for the human user in games. Many researchers argue that emotions should play a vital role in the decision making process [1, 5]. Take for example the emotions happiness and anger. Individuals that are currently happy or angry are more likely to make decisions involving a high risk factor [5].

Although it is apparent that the addition of emotions into NPC decision making has strong potential to bring about human-like behaviour, the effectiveness of emotion within

NPC decision making is still new and requires further research and testing [14]. It is important to balance non-deterministic behaviour along with the required behaviour of that specific agent. Human players expect non-player characters to react rationally while still maintaining a certain style of behaviour required of their role in the game [10].

One approach to incorporate emotions into NPC decision making is to extend behaviour trees. The emotional behaviour tree model presented in [5] proposes the addition of an emotional selector node. This proposed model was tested and yielded successful results.

2.1.6 *Artificial Neural Networks and Evolutionary Algorithms*

There are various cases where artificial neural networks (ANN) and evolutionary algorithms have been used successfully in game environments [16]. The non-deterministic nature of these techniques allows the production of intelligent, adaptive agents in games. These techniques however, have not been fully accepted within the game industry for producing non-player characters [15]. The main reason for this is simply the fact that these techniques are extremely resource intensive [15, 16]. Most commercial games that incorporate resource intensive techniques such as artificial neural networks and genetic algorithms are games in which the goal of the game is centred around the non-player characters [12, 13]. One example of this is Black and White (Lionhead Studios).

2.2 Comparison of Techniques

This section reviewed AI techniques that are commonly used in game development in order to identify suitable techniques to be applied to the MOBA game environment. Table 1 provides a summary of the strengths and limitations of the five techniques found to be used previously.

Table 1: A table summarising advantages and disadvantages of each artificial intelligence technique evident in previous literature.

Technique	Advantages	Disadvantages
The Finite State Machine	- Simple. - Low computational cost.	- Difficult to manage in large systems. - Deterministic.
The Fuzzy State Machine	- Works with incomplete information. - Requires knowledge of simple boolean logic. - Experts in the field may define fuzzy rules to be used. - Less states required to provide less predictable behaviour. - FSM can easily be converted to a fuzzy state machine.	- Fuzzy rules may take longer to develop and be of less quality when experts are not available.
The Behaviour Tree	- Scales well with system. - Modular. - Memory efficient. - Easy to debug. - Extensible. - Tool support available in various game engines.	- Deterministic.
Emotion	- A way to add unpredictable behaviour to agents.	- Not sufficiently tested within game environments.
ANN and Evolutionary Algorithms	- Non-deterministic and produces intelligent, adaptive agents. - Useful when the focus of the game resides around agents.	- Resource intensive.

3. GAME SCENARIO

The following section defines the MOBA game genre as well as the test bed used in this research.

3.1 The MOBA Game Genre

A MOBA style game is generally situated in a 3D world and involves two opposing teams which both consist of a number of characters (i.e. agents), and structures. The goal of each team is to eliminate the core structure of the opposing team which is situated on the opposite end of the world. Each team's core structure is connected to pathways known as lanes. Each lane contains a number of structures belonging to each team. Characters in the game may engage each other in either ranged or melee combat in order to progress down a particular lane and reach the enemy core structure.

3.1.1 *Character and Structure Types*

The following character and structure types are evident in this case study:

- The hero character - The primary class of character in a MOBA. These characters may either be human or computer controlled and will be the main focus of this research. These characters are able to increase in strength as the game progresses. This is done by gaining experience points which contribute to their strength.
- The lane character - A purely computer controlled character that assists hero characters in completing objectives.
- The neutral character - A character that exists purely to provide hero characters with a source of experience. They do not belong to a particular team.
- The tower structure - An defensive structure that is an objective that the enemy team must destroy before destroying the core structure. This structure may attack nearby enemies.
- The core structure - A passive structure that is the primary objective of the enemy team.

3.1.2 *Agent Capabilities*

Each agent is able to move around the world to certain locations and engage another agent or structure in combat. Basic combat is limited to either a ranged or melee attack however heroes contain certain special abilities that they may utilize at times to gain a certain advantage.

3.2 The MOBA Test Bed

The following MOBA game scenario was implemented using the Unreal Engine 4 to analyze and evaluate certain behaviour control techniques:

- Two teams each containing one hero with the same special abilities. A human player may choose either team to join.

- Each team has three structures: two tower structures and one core structure. The tower structures appear along the lane.

- A simplified MOBA map consisting of a single lane and which was designed to fit only two hero characters. The single lane runs through the centre of the world with a team's core structure at each end.

A screenshot of a hero character and a tower structure within the test bed is shown in Figure 1.

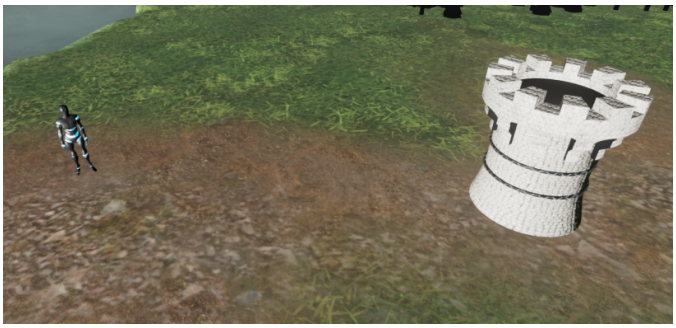

Figure 1: A screenshot within the test bed

4. EXPERIMENTAL DESIGN

The following section provides an overview of the implementation of the EBT and FuSM. The finite state machine and behaviour tree were implemented to control the behaviour of characters that did not require stochastic behaviour. These characters included the lane characters and tower structure AI. The focus of this research was on stochastic NPC behaviour and therefore no quantitative results are presented for these techniques. A qualitative analysis for these techniques is given in Table 1.

4.1 FuSM Implementation

The design of the fuzzy state machine was taken from that mentioned by Priovano [13]. Each fuzzy state has a corresponding membership value. The machine will then select the state with the highest membership value at a given time.

The fuzzy state machine was implemented to control the behaviour of the hero character. The implementation of the hero FuSM was kept close to that of the hero EBT as each state in the FuSM encapsulates a branch of the EBT. The three states are the defensive state, the aggressive state and the training state as seen in Figure 2.

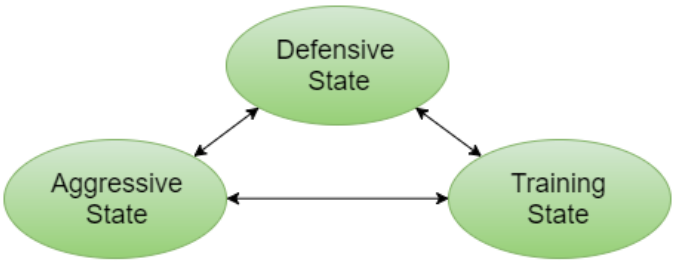

Figure 2: The hero fuzzy state machine

The defensive state handles situations whereby the hero has a high probability of dying. These situations could include aspects such as: low health, a high enemy count, or the presence of a stronger enemy hero.

The aggressive state controls the activity of engaging enemy characters and structures with the short-term goal of winning the game.

The training state enables the hero to seek out neutral characters with the goal of gaining experience points.

4.2 EBT Implementation

The emotional behaviour tree was implemented to control the behaviour of the hero character. The process was split up into two parts. Firstly, a standard behaviour tree was developed to control hero behaviour. Emotional aspects were then incorporated into the existing behaviour tree structure.

4.2.1 Hero Behaviour Tree Implementation

A simplified representation of the hero behaviour tree is depicted in Figure 3.

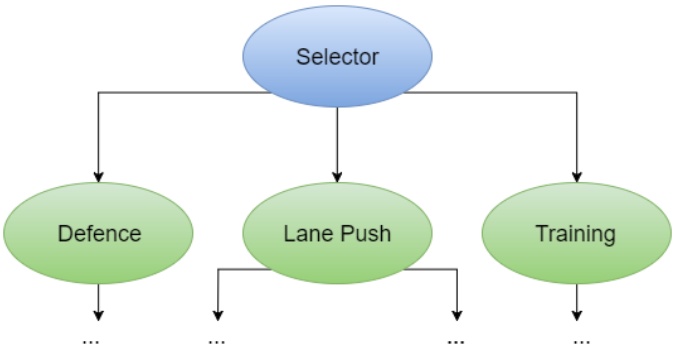

Figure 3: The hero behaviour tree

4.2.2 Incorporating Emotion

Emotion was added to the standard hero behaviour tree through the use of a fuzzy state machine. Each character emotion was represented by a state in the machine. Due to the fuzzy nature of the machine, the character may be in more than one emotional state at a particular time with varying membership. The diagram in Figure 4 provides an overview of the implemented emotion fuzzy state machine. The FuSM contains three nodes which each correspond to a particular emotion. Fear, anger and courage were the three emotions that were included in the EBT decision making process.

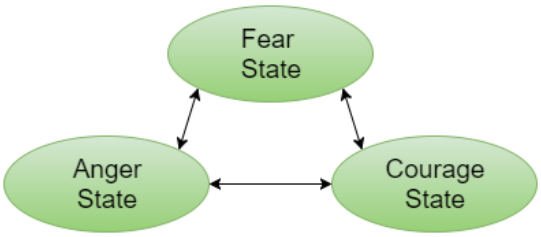

Figure 4: The emotional fuzzy state machine

The method proposed in [5] requires the addition of a new type of selector node. The proposed method does not require the existing structure of the behaviour tree to be changed. In this approach the hero EBT (Figure 3) is adjusted as follows. Each node in the EBT will have access to the emotion FuSM (Figure 4) as depicted in Figure 5. The current emotional state of the agent is extracted from the emotion FuSM and used as an additional input to the behaviour tree. In this way, the emotional state of the agent is continuously calculated in the emotion FuSM and used to influence decision making in the behaviour tree.

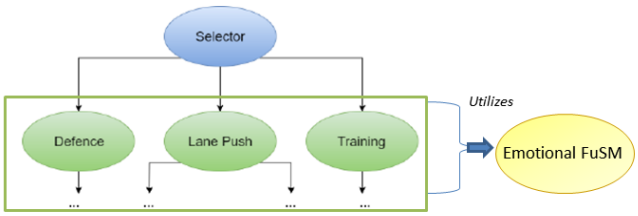

Figure 5: The EBT's relationship to the emotional FuSM

5. RESULTS AND DISCUSSION

The following section provides results obtained as well as a discussion.

5.1 EBT and FuSM Performance Experiment

The purpose of this experiment was to compare the performance of the EBT and FuSM with respect to creating a challenging opponent within a MOBA game environment. Each technique was implemented to control the behaviour of a hero agent. Each agent was set against the same human player for three full length game simulations. The corresponding player kills, death counts, wins and structures i.e. objectives destroyed were recorded. The player kills served as the primary performance measure.

Each simulation lasted an average of fifteen minutes, the preliminary results of which are shown in Tables 2 and 3.

Table 2: Performance of the hero agent using the emotional behaviour tree

Run	Player Kills	Objectives Destroyed	Win	Death Count
1	2	1	False	5
2	2	1	False	6
3	0	0	False	5

Table 3: Performance of the hero agent using the fuzzy state machine

Run	Player Kills	Objectives Destroyed	Win	Death Count
1	2	0	False	1
2	3	0	False	4
3	4	1	False	5

5.2 The EBT and the FuSM Compared

Both the emotional behaviour tree and the fuzzy state machine were successfully implemented to control the behaviour of the hero character class. Preliminary performance results in Tables 2, 3 and Figure 6 indicate that the emotional behaviour tree proved to be a slightly greater challenge for the human player in terms of structure defence. In most runs, the emotional behaviour tree was able to destroy one of the human player's towers. The fuzzy state machine however, had a significantly higher total player kill count and a lower total death count. This shows that the fuzzy state machine proved to be a greater challenge for the human player in terms of combat. In summary of performance, it is evident from Figure 6 that the fuzzy state machine performed better than the emotional behaviour tree within the implemented MOBA test environment. The player kill count has been given a higher weighting than the amount of towers destroyed simply because it involves direct engagement with the human player.

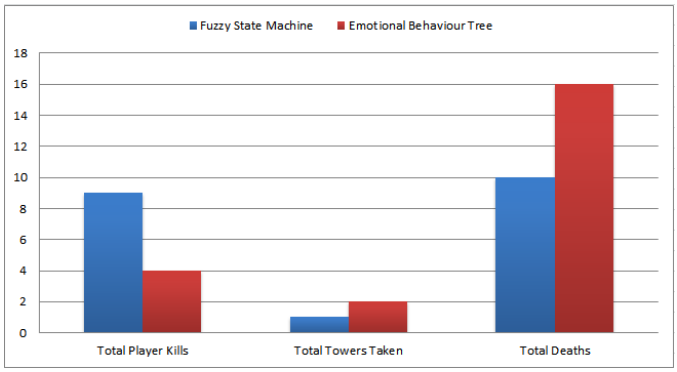

Figure 6: A graph summarising the performance results obtained in Tables 2 and 3.

For a MOBA style game, the emotional behaviour tree does provide many evident implementation advantages. One such advantage is its modular nature and the ability to create sub-behaviour trees. This is particularly useful when developing behaviour for hero characters in large MOBA games. In commercial MOBA style games such as Dota 2 and Heroes of Newerth, there are multiple types of hero characters each with unique ability sets. Although this research considered the case of a single hero character type, the hero behaviour tree engagement branch, as seen in Figure 7, that was developed may easily be extended to cater for engaging different types of heroes. Sub-behaviour trees may be developed to cater for encountering different classes of heroes in the environment. Heroes would then be able to adapt behaviour based on the type of enemy hero they are currently engaging as proposed in Figure 8.

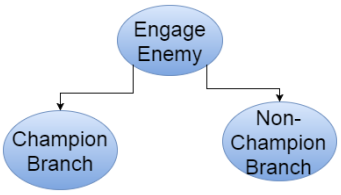

Figure 7: Existing hero EBT engagement branch.

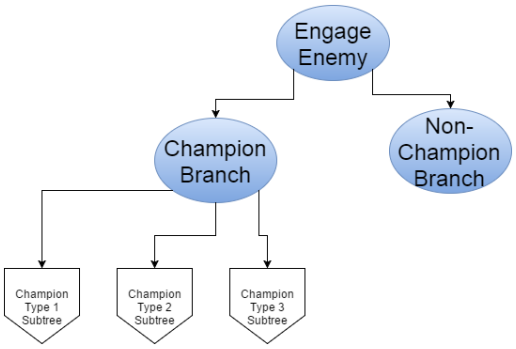

Figure 8: Proposed hero EBT engagement branch.

When making the decision to use either the emotional behaviour tree or the fuzzy state machine to control the behaviour of the hero character class, it is important to consider both the development environment being used and the amount of expert assistance available. If an engine which provides behaviour tree development tools, such as the Unreal Engine 4, is being utilized then the behaviour tree is highly recommended. If this is not the case, the fuzzy state machine should be considered only if expert assistance is available. If fuzzy rules are to be effective, experts in the field, such as professional MOBA players, should be consolidated when rules are written.

5.3 Incorporating Emotion in MOBA Games

The use of emotion in this research was to provide non-determinism to the hero behaviour tree. According to the results evident in Table 2, the emotional behaviour tree appeared to be successful in that it was able to destroy on average one enemy tower per game. The performance overall however, did not match that of the fuzzy state machine. Although data was not recorded, users stated that the emotional behaviour tree controlled hero proved to be a far more interesting and satisfying opponent. One may argue that this is of more importance than the actual performance of the algorithm.

This research has therefore shown that emotion can successfully be integrated with aspects of the hero character decision making process however the extent to which emotion influences decisions should be restricted. A large portion of character decision making within a MOBA style game is required to be deterministic. The portion of the hero character class that should be mostly non-deterministic is enemy engagement. Consider the case of a hero within a MOBA game which is to create a challenge for the human player. The hero is expected to gain experience and eliminate lane characters at a reasonable rate. Therefore, main decisions made by the hero such as deciding when to seek additional experience and when to eliminate lane characters, should be mostly deterministic to ensure that the hero poses a challenge in every game simulation. When the hero notices or engages an enemy player, a higher level of non-determinism is required to create adaptive and exciting combat situations for the human player.

6. CONCLUSION AND FUTURE WORK

This research reviewed and evaluated different artificial intelligence techniques for autonomous control of non-player characters within a MOBA game environment. AI techniques were reviewed based on previous work and a comparison of the most widely used techniques is provided in Table 1. A MOBA test bed was developed within the Unreal Engine 4 to evaluate selected techniques. The fuzzy state machine and emotional behaviour tree were implemented and evaluated within this test bed and preliminary results were obtained.

A novel aspect of the research was the proposal of a new mechanism to incorporate emotions into behaviour trees. The mechanism uses a fuzzy state machine to track the emotional state of the agent and this state is incorporated into the decision making at each node of the behaviour tree. An initial evaluation of the mechanism compared the performance of the Emotional Behaviour Tree (EBT) to a fuzzy state machine using the test bed. While the results showed that the FuSM outperformed the EBT in terms of player kills, human players reported a considerably more interesting and enjoyable game experience with the EBT than the FuSM.

There are various control mechanisms available to introduce elements of unpredictable behaviour for NPCs in game environments. Mechanisms which incorporate emotions have the potential to provide a more engaging and interesting game experience over others. While this research provided an alternate mechanism to that provided in [5], further testing is still required to evaluate this mechanism for widespread adoption in real world game engines.

This research implemented two of the techniques identified from literature and gathered preliminary performance results. Future work may therefore include a wider range of techniques that are compared with respect to the test game environment and further testing within the environment.

7. ACKNOWLEDGMENTS

The author would like to acknowledge the financial assistance given by the National Research Foundation (NRF) of South Africa and the UKZN/CSIR Meraka Centre for Artificial Intelligence Research.

8. REFERENCES

[1] R. Bernhaupt, A. Boldt, T. Mirlacher, D. Wilfinger, and M. Tscheligi. Using emotion in games: Emotional flowers. In *Proceedings of the international conference on Advances in computer entertainment technology*, pages 41–48. ACM, 2007.

[2] M. Dickheiser. *Game Programming Gems 6 (Book & CD-ROM)(Game Development Series)*. Charles River Media, Inc., 2006.

[3] A. Drachen, M. Yancey, J. Maguire, D. Chu, I. Wang, T. Mahlmann, M. Schubert, and D. Klabajan. Skill-based differences in spatio-temporal team behaviour in defence of the Ancients 2 (DotA 2). In *2014 IEEE Games Media Entertainment (GEM)*, pages 1–8, Oct. 2014.

[4] A. Johansson and P. Dell'Acqua. Comparing behavior trees and emotional behavior networks for NPCs. In *2012 17th International Conference on Computer Games (CGAMES)*, pages 253–260, July 2012.

[5] A. Johansson and P. Dell'Acqua. Emotional behavior trees. In *2012 IEEE Conference on Computational Intelligence and Games (CIG)*, pages 355–362, Sept. 2012.

[6] S. L. Kim, H. J. Suk, J. H. Kang, J. M. Jung, T. Laine, and J. Westlin. Using Unity 3d to facilitate mobile augmented reality game development. In *2014 IEEE World Forum on Internet of Things (WF-IoT)*, pages 21–26, Mar. 2014.

[7] R. Maddegoda and A. Karunananda. Multi agent based approach to assist the design process of 3d game environments. In *2012 International Conference on Advances in ICT for Emerging Regions (ICTer)*, pages 36–44, Dec. 2012.

[8] J. Meng, D. Williams, and C. Shen. Channels matter: Multimodal connectedness, types of co-players and social capital for Multiplayer Online Battle Arena gamers. *Computers in Human Behavior*, 52:190–199, 2015.

[9] I. Millington and J. Funge. *Artificial intelligence for games*. CRC Press, 2012.

[10] J. Ness, A. Olsen, M. RÃÿdland, and C. A. Sand. Project NORS: a Multiplayer Online Battle Arena Game Implemented in Unreal Engine 4. 2015.

[11] N. J. Nilsson. *Principles of artificial intelligence*. Morgan Kaufmann, 2014.

[12] M. Pirovano. The use of Fuzzy Logic for Artificial Intelligence in Games. *University of Milano, Milano*, 2012.

[13] M. Pirovano and P. L. Lanzi. Fuzzy Tactics: A scripting game that leverages fuzzy logic as an engaging game mechanic. *Expert Systems with Applications*, 41(13):6029–6038, Oct. 2014.

[14] M. Spraragen and A. M. Madni. Modeling of Emotional Effects on Decision-making by Game Agents. *Procedia Computer Science*, 28:736–743, 2014.

[15] P. Sweetser and J. Wiles. Current AI in games: A review. *Australian Journal of Intelligent Information Processing Systems*, 8(1):24–42, 2002.

[16] E. Tomai and R. Flores. Adapting In-Game Agent Behavior by Observation of Players Using Learning Behavior Trees. In *Proceedings of the 9th International Conference on the Foundations of Digital Games (FDG 2014)*, 2014.