A simple hybrid algorithm for improving team sport AI

David King and David Edwards

This is a paper presented at GameOn 2015, the 16th International Conference on Intelligent Games and Simulation, Amsterdam, Netherlands, 2-4 December 2015

A SIMPLE HYBRID ALGORITHM FOR IMPROVING TEAM SPORT AI

David King
David Edwards
University of Abertay, Dundee
40 Bell Street, Dundee, United Kingdom, DD1 1HG
Email: d.king@abertay.ac.uk

KEYWORDS

Adaptive AI, Fuzzy Logic, N-gram prediction, Team Sports Games.

ABSTRACT

In the very popular genre of team sports games defeating the opposing AI is the main focus of the gameplay experience. However the overall quality of these games is significantly damaged because, in a lot of cases, the opposition is prone to mistakes or vulnerable to exploitation. This paper introduces an AI system which overcomes this failing through the addition of simple adaptive learning and prediction algorithms to a basic ice hockey defence. The paper shows that improvements can be made to the gameplay experience without overly increasing the implementation complexity of the system or negatively affecting its performance. The created defensive system detects patterns in the offensive tactics used against it and changes elements of its reaction accordingly; effectively adapting to attempted exploitation of repeated tactics. This is achieved using a fuzzy inference system that tracks player movement, which greatly improves variation of defender positioning, alongside an N-gram pattern recognition-based algorithm that predicts the next action of the attacking player. Analysis of implementation complexity and execution overhead shows that these techniques are not prohibitively expensive in either respect, and are therefore appropriate for use in games.

INTRODUCTION

Artificial intelligence (AI) and video games have always been intrinsically linked. From providing very basic control of enemy characters in the early days of arcade games (the classic example being the ghosts in *Pac-Man* (Namco 1980), to the creation of complex systems that model the behaviour of realistic human characters in more recent titles.

As the games industry has grown over the years, so too have players' expectations of the perceived level of intelligence and realism exhibited by the enemies they now face. Some in the industry have gone as far as saying that "high-quality game AI has become an important selling point of computer games in recent years" (Tan, Tan and Tay 2011). Conversely, publishing a game that features obviously bad or broken AI is now a sure-fire way to draw harsh criticism from both consumers and the media.

Whereas some effort has been taken to address the imbalance between graphic fidelity and NPC 'intelligence' in RPG and the like, the same effort does not appear to have been taken when designing team sport games. It is still the case that once the player has established a specific strategy to defeat the opposition, this tactic will always work and the opposition are unable to learn or adapt to counter these moves. This significantly limits the replayability and shelf life of the game.

There are many existing AI algorithms capable of incorporating an element of learning/prediction that would be appropriate (Millington 2006). However, the method does not necessarily need to be complex. Decision trees and FSM-based systems form effective frameworks that can be adapted and augmented in various ways to exhibit the desired characteristics. The key issue then is selecting the techniques that are most appropriate in terms of code and implementation complexity; while also achieving the desired adaptive effect in the given situation.

ADAPTIVE AI

The goal of this project was to create a defensive system that would detect the use of repeated tactics, and react to this attempted exploitation in a behavioural way. For this purpose, adaptive AI can be defined as any algorithm which takes relevant data from the player's actions and changes the behaviour of the AI system in an appropriate way. In this game-specific context there is no definite solution due to the constantly changing nature of the desired gameplay; the goal is to simply improve the AI system in a way that allows it to be less rigid.

METHODOLOGY

The first step of developing the simple, adaptive AI was to build a basic ice hockey defence simulation. This system acts as a framework upon which the desired features are implemented separately to allow for comparison of results. It also enables access to necessary input data for the desired algorithms as well as application of their outputs to the defending agents. All adaptive functionality is built on top of this baseline application. Additionally, all data acquisition and processing will occur during execution (online adaption) so that the performance of each algorithm can be analysed. The ice hockey game was built using the Unity3D engine (Unity Technologies 2013).

A decision tree was implemented for the base framework as there is only really one state that the defenders can be in; defending. This ensures that the decision tree is kept fairly short and easy to visualise and maintain. A FSM would make more sense in a full hockey simulation where both teams can take possession of the puck, and where players can be on the ice or sitting on the bench.

A basic ice hockey defence was designed and implemented. It features two defending players whose movement and actions are controlled by a decision tree. Two

attacking players have also been created, and can either be controlled directly by the player or via a choice of three scripted offensive plays that have been created to control them automatically. When the attacking player who possesses the puck enters the defensive zone (he must be the first attacking player to do so, otherwise it would be called offside), the decision tree was activated.

Trigger boxes are used to detect when the attacking players have entered the defensive end. A vector is drawn from the puck carrier to the goal and this vector is used to position one of the defending players between him and the net. If a second attacker (a passing target for the puck carrier) is also detected, a vector is drawn between the two attackers and used to position one of the defenders between them. These processes can be seen in Figure 1.

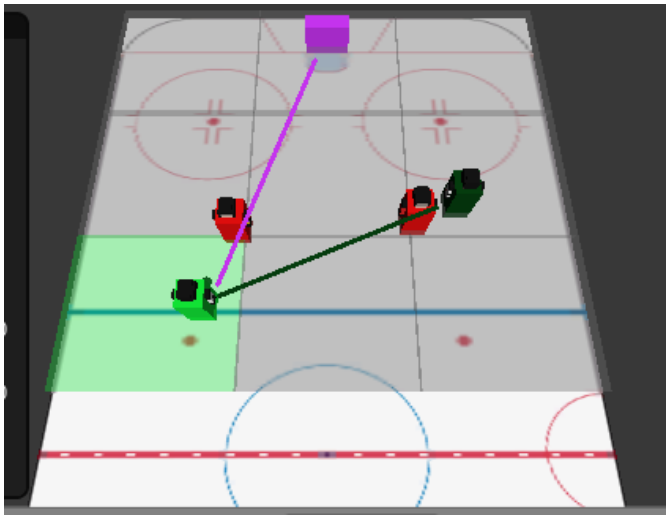

Figure 1: Detection Zones and Position Vectors in Created Application.

The leaves of the decision tree are "block goal" and "take action", which represent actions that the defending players can take. It is at these stages of the defensive process that the adaptive functionality is implemented.

Defender Positioning

The first adaptive aspect of the AI is in the positioning of the defender who challenges the puck carrier. A crucial element of this positioning is how far away from the puck carrier the defender will stay. This "offset" is a scalar value that is applied to the position vector of the defending player to move them closer to or further away from the puck carrier. This value is determined by a Fuzzy Inference System (FIS).

There are two inputs to the FIS. The first is the magnitude of the vector drawn from the puck carrier to the goal (scaled to between 0 and 1) i.e. the current distance from the net. Logically the closer an attacker is to the goal, the more likely they are to shoot. The second input is the "event heat" of the current detection zone (the shaded square area in Figure 1). The event heat is measured separately for each zone, again as a value between 0 and 1. Every time a shot or pass occurs in a zone, a weighted value is added to that zone's heat and another value subtracted from that of other zones (the exact values for which have been reached through an iterative process of trial and error). In this way, repeated actions in the same zone raise its heat very quickly, and the heat of other zones will reduce more slowly. This allows the system to adapt very quickly to repetition, while maintaining some memory of previous choices.

Fuzzy Inference System

The initial rule-base for the FIS took the inputs detailed above and combined them as shown in Table 1, the outputs relating to the amount of offset. The rules were initially set up symmetrically, balanced equally between both inputs and the outputs.

Table 1: Initial Rules of the FIS

	Distance		
Event Heat	Low	Medium	High
High	Low	Low	Medium
Medium	Low	Medium	High
Low	Medium	High	High

Though this seemed like a sensible approach, when applied to the simulation the resulting behaviour appeared rather unresponsive. The puck carrier was allowed far too much space when near the goal and in high-scoring zones that were not very close the net. To balance the defence in a more aggressive way, the rules of the system were changed to those in Table 2.

Table 2: Final Rules of the FIS.

	Distance		
Event Heat	Low	Medium	High
High	Low	Low	Low
Medium	Low	Medium	Medium
Low	Low	Medium	High

As shown, when the distance to the goal is low and the event heat of the current zone is high, the offset output by the system is low. However, the offset should also be low whenever the event heat is high or the distance is low.

The fuzzy set for the input 'Distance' is shown in Figure 2. The fuzzy set for 'Event Heat' is the same. The domain of each membership function was reached through experimentation both in MATLAB (MathWorks 2014) and using the application. Triangular shapes were chosen for the sake of simplicity; ensuring that fuzzifying the input values was as efficient as possible. The output fuzzy set 'Offset' (Figure 3) also has triangular membership functions. In addition to this, they do not overlap to simplify the defuzzification process.

The FIS uses the centroid method of defuzzification to generate a crisp numerical value from the fuzzy output that is calculated. Though far from the simplest method of defuzzification, the centroid method is one that gives the most variation of output values (Nurcahyo, Shamsuddin and Alias 2003). Since lack of variation is generally the problem with crisp rule-based implementations, it made sense to select the centroid method for this reason

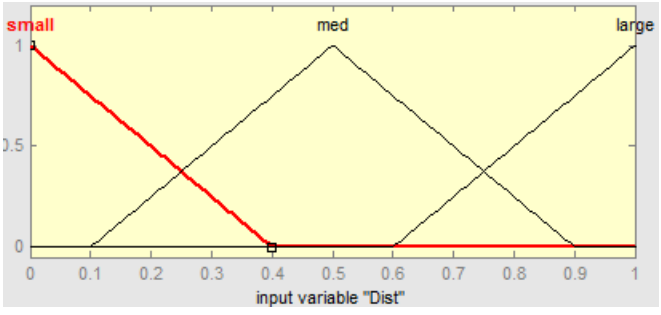

Figure 2: Input Fuzzy Set 'Distance'

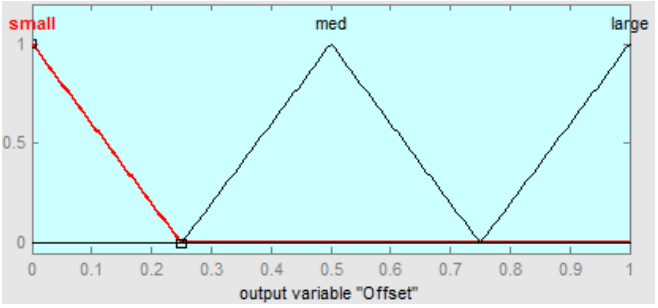

Figure 3: Output Fuzzy Set 'Offset'

Due to the shape of the membership functions, the implemented defuzzification gives an output offset value that is between 0.09 and 0.92. This value is then applied to the defender's offset vector via scalar multiplication, allowing for the gap between the two players to adapt to the given inputs.

Action Prediction

Defender positioning is only one component of an adaptive defence. The defender also has to decide whether the attacker is going to shoot or pass and act accordingly. The defensive end has been split in to 9 detection zones, shown in Figure 4. More than one detection zone can be active at any given time. During play each detection zone stores the number of shots and passes that have occurred and then stores the previous ten events as an ordered string of Char objects.

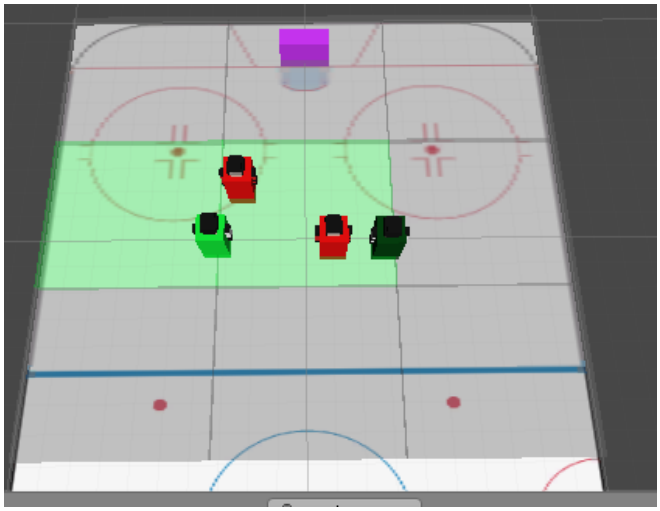

Figure 4: Detection Zones in the Defensive End.

N-gram Pattern Recognition

The action prediction implemented in the application is a 3-gram string-matching algorithm (Muise. et al. 2009). A shot is stored as an 'S', and a pass as a 'P'. An example event history from the completed application is shown in Figure 5. When an event is likely to occur (event heat > 0.5), the history of the current zone will be analysed. In this 3-gram method, the algorithm takes the last 2 events (P and S, in this case) and searches the history for instances of this pattern. The event that follows this pattern most often (another pass, in the above case) is then chosen to be the next predicted action. However, if the previous two actions have not occurred in that order before, the algorithm will be unable to

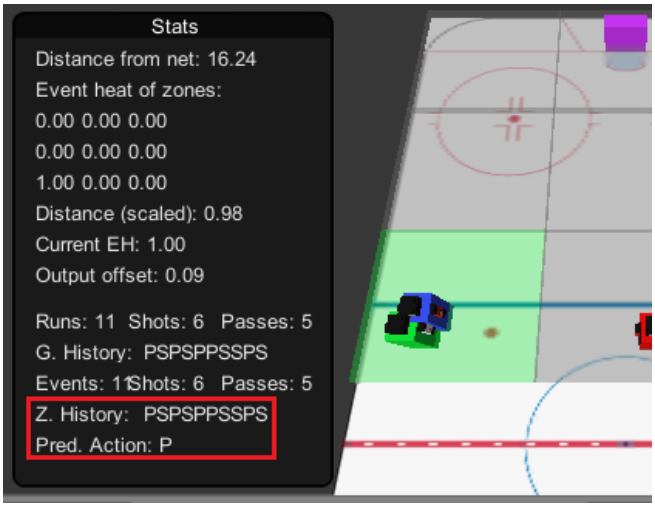

Figure 5: Event History of the Current Active Zone.

predict what the next action will be. For this reason, a final piece of work was carried out in this area.

Probability and N-gram Combination

A combined action prediction system was created. It simply carries out the previously described n-gram string-matching, and if no event can be predicted by that method, reverts to a probabilistic approach. When the event heat of the current active area is above 0.5, the total number of passes and shots for that area is compared, and the one with the larger volume is predicted to be the next action chosen by the player. If they are equal, the defence will play safe and predict a shot.

EVALUATION

Defender Positioning

Presented below are the graphed results of the FIS controller compared to a simple Crisp Rule Based System using the same rule base. One run with no event heat present (Figure 6), one with medium event heat present in the first zone (Figure 7) and a final run with high event heat in zone 1 (Figure 8) have been simulated. It should be noted that the sudden drops shown at the start of the second and third graphs are due to the puck carrier first entering the defensive zone.

As expected, the output from the crisp rule-based system is distinctly rigid and unvaried. The FIS offset values show a much greater degree of variance, resulting in a greater range of output values. Both systems seem to react to the combination of inputs in similar ways, suggesting that they both allow for a similar degree of adaption; though this is to be expected since they make use of the same rule set.

Action Prediction

Each of the three methods of action prediction (probabilistic, pattern recognition and combination) have been presented with a set of defined historical data. Table 3, shows the action predicted by each when exposed to the given string of event history. 'S' denotes a shot, 'P' a pass and an 'E' signifies that no action could be predicted. Though the pattern recognition method could handle this in a number of ways (default to predicting a shot, carry out previous chosen action, choose one at random, etc.), this

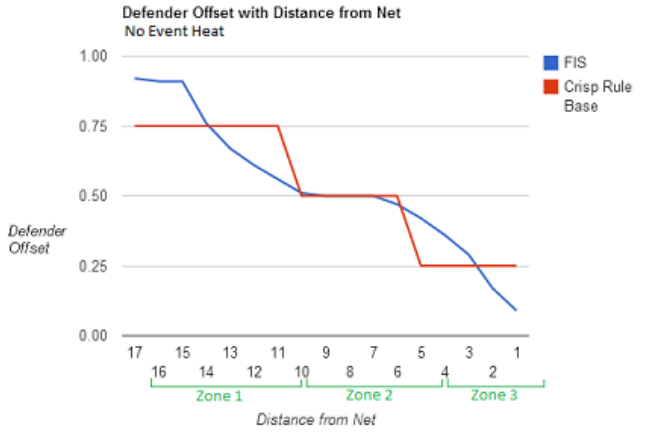

Figure 6: Results with no Event Heat and Decreasing Distance

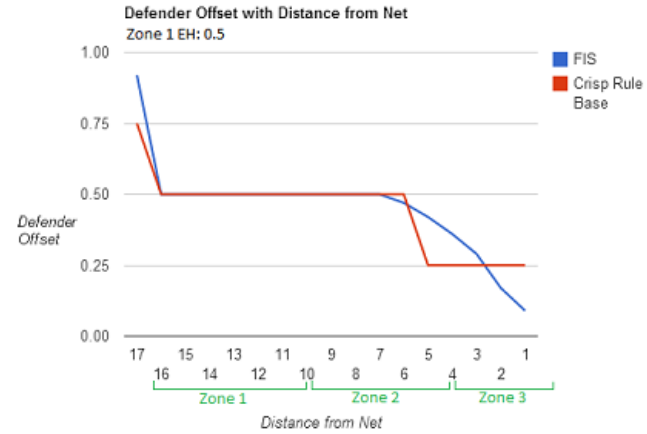

Figure 7: Results with Medium Event Heat in the First Zone

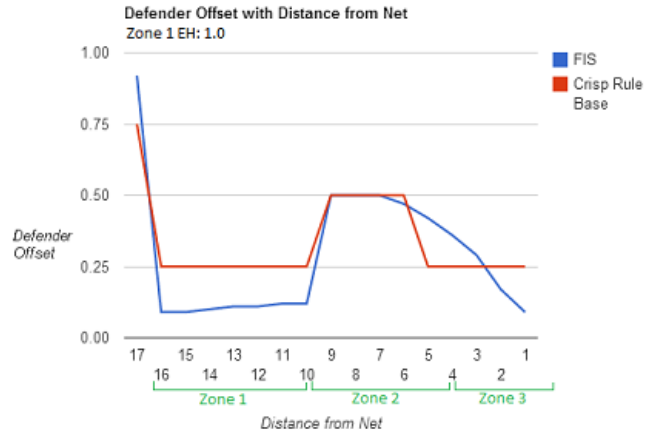

Figure 8: Results with High Event Heat in Zone 1

Table 3: Predicted Actions When Exposed to Historical Data

History	Desc.	Prob.	Pattern	Comb.
SSSSSSSSS	10 S	S	S	S
SSSSSSPPP	7S, 3P	S	P	P
SPSPSPSPSP	5S, 5P	S	S	S
SSPSSPSSPS	7S, 3P	S	S	S
PSSSPSPSPS	6S, 4P	S	P	P
PPSPPSPPSS	6P, 4S	P	E	P
SSSSSSSSP	9S, 1P	S	E	S

would not be a meaningful prediction and therefore has not been implemented for these results.

Complexity

The complexity of each adaptive algorithm is displayed in Table 4. For comparison, the basic decision tree-based defence logic is included at the top.

Table 4: Complexity of Implemented Techniques

Technique	Lines of code	CPU time (ms)
Decision tree defence	281	between 0.02 and 0.05
NC - Crisp rule-based	99	0.01
NC - FIS	438	0.01
AP - Probability	4	0.01
AP - Pattern Recog.	85	0.03
AP - Combination	89	0.03

DISCUSSION

The main issue with evaluating gameplay systems such as those created is that there is often no real scientific way of measuring success in this context. While it is easy to test whether the defender is close to the puck carrier when they should be, it is much harder to say how effective a prediction algorithm is when the next action the player will take cannot be known. For this reason, there is bound to be an inherent level of subjectivity in any analysis of most created game AI systems. That said, much effort has been made to avoid bias both in the implementation of each algorithm and now in their discussion.

The overall results of the project (in terms of what has been created) are generally positive. Multiple methods of creating each desired feature have been researched, designed, implemented and tested. A robust application has been created, which showcases different adaptive AI algorithms in a relevant team sports context.

As shown in the graphs for Defender Positioning, there is a clear difference in the amount of output variation given by the FIS in comparison with a Crisp Rule-Based System. The FIS produces a greater array of different offset values than the crisp rule base does, and covers a wider range of the given domain. Though the range of values given by the rule-based system is somewhat due to implementation choice, there is no way around the lack of variation that results from using such a system. For this reason, if variation in terms of resulting values is desired when implementing a numerical control system of this kind, the FIS is clearly the better choice. The increased variation given by the FIS will make it appear to react in a more natural, realistic way.

Another thing worth noting is that the created system is far too precise in nature, even with the stated increase in variation provided by the FIS. In a sport as fast-paced as ice hockey, it is entirely unnatural for defending players to always be exactly in the right position. For this reason, it would be a good idea to add some kind of constrained random adjustment to the blocking defender's position vector. This would serve to ensure that some shots would actually make it past them.

As for the Action Prediction the purely probabilistic method has the distinct advantage of always being able to return a meaningful prediction, as long as at least one event has occurred in the current zone. This means that it is rapid to detect when an action is being repeated. It does not need to wait for sufficient history data to analyse properly. The n-gram method relies on sufficient history data being present

for it to find any form of pattern. When repetition cannot be found, the purely pattern-based prediction is unable to give a relevant output. While this can easily be fixed by having it default to predicting a shot, it is still a definite weakness of the approach. However, combining the two techniques so that probability is used when a pattern cannot be found results in a very robust system that has the strengths of both approaches, with no obvious drawbacks.

Complexity

As Table 4 shows, the crisp rule-based system required only 99 lines of code to implement, whereas the FIS resulted in over 400. While neither system has been compressed aggressively in terms of code (readability and clarity to a new observer have been emphasised during development), this is a marked increase in implementation complexity and therefore time. The fact that the FIS is substantially longer than the entire baseline decision tree defence logic suggests that it may only be worth implementing if numerical variation is highly important in the game situation. If not, a simple rule-based approach may be entirely appropriate.

However, as evidenced in Table 4, there is no marked increase in performance overhead when a FIS is used instead of a crisp rule set. This is due to the fact that both systems essentially boil down to a set of logical AND operations; with the FIS requiring simple and efficient calculations on either side of these rules. There are no expensive memory operations, so performance overhead is minimal and not of concern for current generation hardware.

Unsurprisingly, the probabilistic action prediction is incredibly easy to implement. A simple comparison of two numbers has almost no implementation cost at all, as shown in Table 4. Though somewhat more complex in concept, the created n-gram prediction only seems costly to implement due to its comparison with the probability approach (85 lines of code vs 4). In context, even the combination of both to create a fairly robust and effective system is far less costly than the original decision tree-based defender control.

Performance-wise, there is a small but noticeable increase in the CPU time used by the pattern recognition. However, the resultant CPU usage is still incredibly small and most definitely not an issue on any form of modern processor.

Future Work

While it is clear that an adaptive system has been created that functions correctly, there has been no evaluation of whether players would actually *enjoy* tackling it. The next step in developing such a system would therefore be to carry out some form of survey-based analysis of whether the average player feels the created system is fair, balanced and actually effective at what it aims to do.

In order to carry out the above evaluation, it would be a good idea to extend the created application into a fully-fledged team-based ice hockey simulation. This would involve making minor changes to how the defence currently functions, as well as implementing some form of attacking AI as well. It is likely that this would be a significant undertaking, but one that is entirely necessary to fully evaluate the performance of the defensive system in a proper context.

A logical extension would be to try to port the created adaptive system to other, similar team sport games. Though the gameplay mechanics of other sports like football and basketball are very different, the core concepts of defending (effective positioning, shot blocking) are completely transferable. In this way the AI performance of not just ice hockey, but all similar team sports games could be improved to provide a better experience for the paying customer.

CONCLUSIONS

In conclusion, the development of this project has shown that intelligent and believable behaviour can be modelled with a combination of fairly simplistic techniques. With graphical improvements in games becoming less and less noticeable with each new generation of hardware, it would seem then that creating better and more engaging game AI is of the utmost importance. There is clearly room for improvement in the games industry's approach to AI development as a whole; this project alone demonstrates that existing rigid systems can be improved without massive costs in development or impacts to performance.

REFERENCES

Millington, I. 2006. *Artificial Intelligence for Games.* Morgan Kaufmann, San Francisco, C.A.

Muise, C. et al. 2009. "Exploiting N-gram analysis to predict operator sequences." In: *Proceedings of the Nineteenth International Conference on Automated Planning and Scheduling*.

Namco *Pac-Man*. 1980. [arcade]. Arcade. Namco.

Nurcahyo, G. Shamsuddin, S. and Alias, R. 2003. "Selection of Defuzzification Method to Obtain Crisp Value for rRpresenting Uncertain Data in a Modified Sweep Algorithm." *Journal of Computer Science & Technology*. 3(2). 22-28.

Tan, C., Tan, K. and Tay, A. 2011. "Dynamic Game Difficulty Scaling Using Adaptive Behavior-based AI." *IEEE Transactions on Computational Intelligence and AI in Games.* 3(4), 289-301.

WEB REFERENCES

MathWorks. 2014. *What is Sugeno-type fuzzy inference?* [online]. Available from: http://www.mathworks.co.uk/help/fuzzy/what-is-sugeno-type-fuzzy-inference.html [Accessed 30 April 2014].

Unity Technologies. 2013. *Unity*. [software]. Version 4.3.2. Available online, from: https://unity3d.com/unity/download/archive

BIOGRAPHY

DAVID KING is a lecturer in Maths and Artificial Intelligence at Abertay University teaching on the Computer Games Technology and Computer Games Application Development Programmes.

DAVID EDWARDS Graduated with First Class Honours in Computer Games Technology from Abertay University in 2014 and is now a programmer for Gameplay, AI & User Experience.