Dempster-Shafer's Approach for Autonomous Virtual Agent Navigation in Virtual Environments

Jafreezal Jaafar and Eric McKenzie

***Abstract*—This paper presents a solution for the behavioural animation of autonomous virtual agent navigation in virtual environments. We focus on using Dempster-Shafer's Theory of Evidence in developing visual sensor for virtual agent. The role of the visual sensor is to capture the information about the virtual environment or identifie which part of an obstacle can be seen from the position of the virtual agent. This information is require for vitual agent to coordinate navigation in virtual environment. The virual agent uses fuzzy controller as a navigation system and Fuzzy $\alpha - level$ for the action selection method. The result clearly demonstrates the path produced is reasonably smooth even though there is some sharp turn and also still not diverted too far from the potential shortest path. This had indicated the benefit of our method, where more reliable and accurate paths produced during navigation task.**

***Index Terms*—Agent, Navigation, Demster Shafer, Fuzzy Logic**

I. Introduction

Navigation is the process where people control their movement using environment cues and artificial aids such as maps so that they can achieve their goal without getting lost [1]. Autonomous virtual agent navigation in a virtual environment can be described as the ability of a virtual agent to move purposefully without user intervention. The navigation task may be decomposed into three sub-tasks: mapping and modeling the environment; path planning and selection; path following and collision avoidance [2]. Virtual agent navigation can occur in known and unknown environments. For a known environment, the virtual agent will have knowledge about the environment and can generate the navigation path. The methods used are based on optimization and computational intelligence. In contrast, in an unknown environment in which the virtual agent does not have any knowledge about the environment, the navigation path is generated according to user specifications and the virtual agent cannot be prepared ahead of time [3].

II. Previous Work

The basic problem of navigation is moving from one place to another by the coordination of planning, sensing and control. The challenge is generating an optimal traversing sequence through the user-specified locations of interests and computation of a collision free path. [3] had shown an example of a path traversing through all user-specified locations . In order to navigate in an unknown environment, a virtual agent needs to deal with the environment in a timely manner.

Approaches such as discrete grid based [4], central path computation [5] and roadmap with tactical information approaches [6] have been used for collisions free path planning. For example [7] studied navigation among static and movable obstacles. The planner takes advantage of the navigational structure through state-space decomposition and a heuristic search. The planning complexity is reduced to the difficulty of the specific navigation task, rather than the dimensionality of the multi-object domain.

Inspired by studies in human behaviour, [8] proposed a general model to simulate the navigation process inside indoor and outdoor environments. Techniques such as set hierarchy, regular graph, artificial potential field and corner graph have been used but are only suitable for 2D environments. One of the reasons is those algorithm require high computational resource in 3D environments. For a 3D environment, navigation mesh and waypoint graph techniques are very popular. A navigation mesh technique is a representation that covers the walkable surface of the world with convex polygons [9]. Waypoint is a set of points in the 3D environment with reachability links between them [10], where we can place a waypoint at any point in 3D space. The disadvantages of these two techniques are large memory usage, and they require a powerful processor. Even though some of these techniques have been used in computer games, it is still not clear that these approaches have been used in autonomous navigation in virtual environments [11].

Artificial intelligence techniques, for example neural networks [12], genetic algorithms [13] and reinforcement learning [14] have been used. [15] presented a multi-agent based evolutionary artificial neural network (ANN) for general navigation. The virtual creature explores unknown environments as far as possible with obstacle avoidance. Through constant interaction with the environment, the virtual agent systems co-decide and consult with each other for the move decision. [12] have integrated attention and navigation skills in a 3D virtual agent. They divided their neural model into two main phases. First of all, the environment categorization phase, online pattern recognition and categorization of the virtual agent current input sensor data is carried out by an adaptive resonance driven self organizing neural network. Then, the model must learn how and when to map the current short term memory state into navigation and the attention of the virtual agent. However the majority of 3D virtual agents focus on low cost global techniques to solve navigation problems and attention is less frequently considered in virtual worlds.

The reactive virtual agent [16] is capable of carrying out

Jafrezal Jaafar is a lecturer with Computer and Information Sciences Department, Universiti Teknologi PETRONAS, Malaysia (email: jafreez@petronas.com.my)
Eric McKenzie is a senior lecturer with the Institute of Perception, Action and Behaviour, School of Informatics, University of Edinburgh (email: ram@inf.ed.ac.uk).

autonomous navigation. The virtual agent extends the artificial potential field approach, used for trajectory formation, to environment exploration and symbolic feature detection. The virtual agent's capabilities range from obstacle avoidance to maze navigation, carried out autonomously or under the supervision of higher cognitive levels. Other methods by [11] have been used in a known environment. On the other hand, in an unknown environment, methods such as sensor based control in [2] use Adaptive Dynamic Points of Visibility (ADPV) for moving virtual agents in dynamical unconfigured environments.

III. Navigation System

The navigation system can be divided into three main components, which are the fuzzy navigator, virtual agent and the environment, as in Figure 1. The main component of the navigation system is the virtual agent itself. The virtual agent should be able to make its own decisions; does not require any information about the virtual environment; and does not require any training or learning before the navigation task.

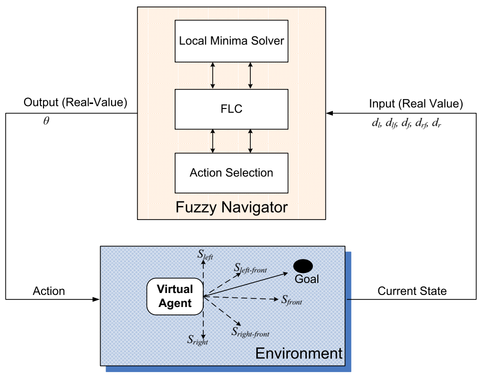

Figure 1. Navigation System

The fuzzy navigator is the main engine for the virtual agent. It comprises of three main components:

1) *Fuzzy Logic Controller (FLC)* - using a behaviour-based architecture which comprises of Path-Planning Behaviour (PP), Goal-Seeking Behaviour (GS) and Obstacle-Avoidance Behaviour (OA).
2) *Local Minima Solver (LMS)* - responsible for helping the virtual agent escape from dead-ends.
3) *Fuzzy Action Selection Mechanism (Fuzzy-ASM)* - to make the final decision in selecting the possible action required by the virtual agent to reach the goal.

The fuzzy navigator receives input from the visual sensor and produces the final action needed to be executed by the virtual agent. Each component in the fuzzy navigator is integrated and works independently.

A. Visual Sensor

The main information between environment and virtual agent is retrieved using a visual sensor. This visual sensor differs from vision systems in robotics, since all information about pattern recognition and noisy images can be ignored [17]. The visual sensor captures the information about the virtual environment or identifies which part of an obstacle can be seen from the position of the virtual agent as in Figure 2. Also, the visual sensor only identifies whether a square (cell) in the vision range is occupied by an obstacle or not. The assumption has been made that all objects are opaque.

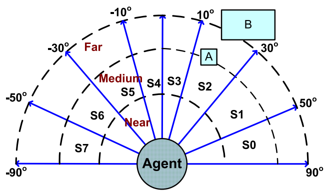

Figure 2. Example of Vision Field and Sensor's Region based on location.

The visual sensor field of the vision range is 180^o. The vision field is divided into eight main sectors which are represented as $S0$, $S1$, $S2$, $S3$, $S4$, $S5$, $S6$ and $S7$. Hence, there is a probability that the cells located in the proximity may be occupied. Cells well inside the vision field sector are likely to be empty. An occupancy grid is essentially a data structure that indicates the certainty that a specific part of space is occupied by an obstacle. It is a representation of an environment as a two-dimensional array. Each element of the array corresponds to a specific square on the surface of the actual world, and its value shows the certainty that there is some obstacle there.

The visual sensor in [18] has been modified by using Dempster-Shafer evidence theory [19]. Whenever the virtual agent moves, it catches new information about the environment and updates the map. To facilitate building an occupancy map [13] of the environment, a grid representing the whole space needs to be constructed. Every discrete region of the map (each cell) may be in two states, $Empty$ is (E) and $Full$ is (F). Then, a frame of discernment, κ, is defined by the set $\kappa = \{E, F\}$, where E and F represent the possibility that a cell is *Empty* or *Full*. The advantage of this technique is that the building of occupancy maps is well suited to path planning and obstacle avoidance [20].

Review of [20] Use of Dempster-Shafer's Theory of Evidence: A basic probability assignment is a function $m : \kappa \rightarrow [0, 1]$, where Γ is a set of all subsets of κ. In our case, $\Gamma = 2^\kappa = \{\phi, \{E\}, \{F\}, \{E, F\}\}$. The state of each cell is described by assigning a basic probability number to each label in Γ. For each cell (i, j) in the grid, it is required that:

$$m_{i,j}(\phi) = 0 \tag{1}$$

$$\begin{aligned}\sum_{A\subset\Gamma} \{m_{i,j}\}(A) &= m_{i,j}(\phi) + m_{i,j}\{E\} \\ &\quad + m_{i,j}\{F\} + m_{i,j}\{E, F\} \\ &= 1\end{aligned} \tag{2}$$

Every cell in the environment is initialized as follows:

$$m_{i,j}\{E\} = m_{i,j}\{F\} = 0 \tag{3}$$

$$m_{i,j}\{E, F\} = 1 \tag{4}$$

Then, the virtual agent moves and scans the environment. If n cells exist in the vision field sector, the basic probability assignment for the vision field sector is as follows:

$$m_{i,j}(F) = \frac{1}{n}, m_{i,j}(E) = 0, \ \forall \text{cells}(i,j) \in \text{sector} \quad (5)$$

$$m_{i,j}(F) = 0, m_{i,j}(E) = 0, \ \forall \text{cells}(i,j) \notin \text{sector} \quad (6)$$

By adding subscripts S and M to basic probability masses m, we can describe the basic probability assignment of the sensor as equations (8) and (9):

$$K = 1 - m_M(E)m_S(F) - m_M(F)m_S(E) \quad (7)$$

$$m_M \oplus m_s(E) = \frac{\{m_M(E)m_s(E)+m_M(E)m_s(\{E,F\})+m_M(\{E,F\})m_s(E)\}}{K} \quad (8)$$

$$m_M \oplus m_s(F) = \frac{\{m_M(F)m_s(F)+m_M(F)m_s(\{E,F\})+m_M(\{E,F\})m_s(F)\}}{K} \quad (9)$$

However, the number of states can be reduced to two $(m_{i,j}(E), m_{i,j}(F))$, assuming that $m_{i,j}(\phi) = 0$ and applying equation 2. The state $(0,0)$ means total ignorance, and so $m_{i,j}(\{E,F\}) = 1$. When the virtual agent is sure about cell occupancy, $m_{i,j}(F) = 1$, the other labels are made equal to zero. On the other hand, $m_{i,j}(E) = 1$ when the virtual agent is sure that the cell is empty.

The input value Θ of the virtual agent, which is a real number normalized in the interval $[0,1]$, then results from a weighted sum of all the points in the visual field.

$$\Theta = \sum_x (2^{-2d(x)}\mu(x)) \quad (10)$$
summed over all x in visual field

where $d(x)$ is the distance of a point x from the current position of the virtual agent, and $\mu(x)$ indicates the availability of the point x. Since the visual sensor is related to availability of spaces in the visual field, it is independent of specific environments and objects. The result is that the occupancy of cells is increased. This process will be carried on until the virtual agent reaches the goal.

B. Fuzzy Controller

A Fuzzy Associative Memory (FAM) is used as a process of encoding and mapping the input fuzzy sets to the output fuzzy set [21]. The fuzzy controller is based on our proposed method [22], [23]. Consider a set of fuzzy rules, $R = \{R_1, R_2, \ldots, R_i, \ldots, R_k\}$, where R_m is the m^{th} rule of the fuzzy controller. The rule R_m is given as follows:

$$\begin{array}{c}\text{IF } X_1 \ is \ A_1^m \text{ AND } X_2 \ is \ A_2^m \text{ AND} \\ \ldots \text{AND } X_n \ is \ A_n^m \text{ THEN } Z \ is \ C_n^m\end{array} \quad (11)$$

The following fuzzy relation will implement R_i:

$$R_m \begin{pmatrix} X_1, X_2, \\ \ldots, X_n, Z \end{pmatrix} = \begin{bmatrix} A_1^m(X_1) \wedge A_2^m(X_2) \\ \wedge \ldots \wedge A_n^m(X_n) \end{bmatrix} \to C_n^m(Z) \quad (12)$$

where $X_1, X_2, \ldots, X_n$ are input variables which are the sensor data of the virtual agent, $A_1^m, A_2^m, \ldots, A_n^m$ are the input fuzzy sets, C_n^m is the output fuzzy set, Z is the output variable, n is the dimension of the input vector.

The weighted sum C for each individual membership can be defined by using $minmax$ aggregation [24] operators as given below:

$$\begin{aligned} C &= \sum_{m=1}^{k} U_m C_m^{'} \\ &= \sum_{m=1}^{k} U \left(\begin{bmatrix} A_1^m(X_1) \wedge A_2^m(X_2) \wedge \\ \ldots \wedge A_n^m(X_n^m) \end{bmatrix} \Rightarrow C_m(Z) \right) \end{aligned} \quad (13)$$

where the non-negative weight U_m, summarises the strength of the m^{th} FAM rule for the m^{th} FAM entry and $n \times m$ is the number of rules in the system. In order to relate the n^{th} fuzzy set of the m^{th} fuzzy rule, the fuzzy implication model using the Mamdani min operator [25] interprets the logical rules for rule firing. we obtain the final defuzzification response for a k output membership function $\mu_C(z)$ is defined as:

$$\mu_C(z) = \max_{\substack{m=1 \\ X=U}}^{k} \left[\min_{m=1}^{n} \left[\mu_{A_n^m}(X_n), \mu_{R_m}(X_1, X_2, \ldots, X_n, Z) \right] \right] \quad (14)$$

Equations (12) and (14) are used to derive the FAM model and the output fuzzy system, respectively.

C. Action Selection Method

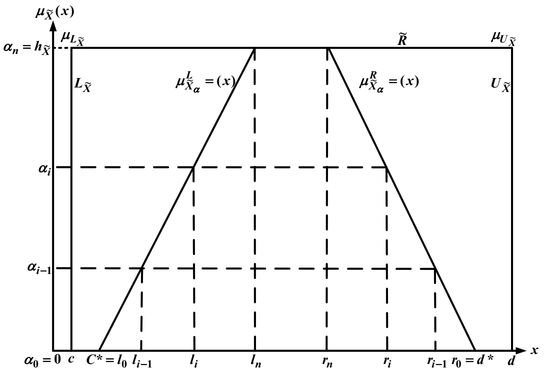

Figure 3. Trapezoidal fuzzy number

Behaviour selection is based on our proposed method [26]. Based on Figure 3, the α values produced from each behaviour will be used to determine the behaviour weight, W, which can be defined as:

$$W\left(\tilde{X}_i, \tilde{R}\right) = \frac{\sum_{i=0}^{n}(r_i - c)}{\sum_{i=0}^{n}(r_i - c) - \sum_{i=0}^{n}(l_i - d)}, \quad (15)$$
as $n = \infty$

Once the behaviour weight value is determined for each behaviour, it is multiplied by an index of optimism, σ, or weight factor. The final action is selected based on the Hurwicz criterion. The σ value is chosen by the programmer and allows certain behaviours to be given preference over others. For instance, if it is desired that a ghost act more aggressively, the hunting approach output can be multiplied by a larger value of σ. Alternatively, if the goal is to make the game easier, a ghost's tendency to move randomly can be increased by multiplying the defuzzified output of the random behaviour by a larger weight.

IV. Results

Experiments were also conducted to observe the effect of using different degrees of optimism, σ, by the virtual agent to navigate in complex environments. Figure 4 shows the result of the experiment conducted in a cluttered environment using different degrees of optimism, σ, which are (a), $\sigma = 0.9$ and (b) $\sigma = 0.4$. The environments contain different sizes of obstacle and narrow passages. The virtual agent in Figure 4(a) has produced a shorter path compared to the virtual agent in Figure 4(b). However the number of steps is higher compared to Figure 4(b). The main reason is that the virtual agent is required to go through a narrow passage in order to produce the shortest path. In Figure 4(b), the virtual agent has made a sharp turn and high number of time steps at this point. As a result the virtual agent take a big turn to the wider passage before turning and reaching the goal. Time steps at the rest of the path are consistent since there is no complex obstacle to avoid. The results show that the decision process by the virtual agent is affected by the degree of optimism. Using a higher value of σ makes the virtual agent enter the narrow passage compare to a low value of σ which makes the agent prefer to select the wider passage. However the number of decisions and steps might vary depending on the complexity of the environment.

Further experiments with complex environments have been conducted. The environments contain a combination of maze and cluttered obstacles and three random goals have been selected. The degree of optimism, $\sigma = 0.5$, was used for the first trial. Unfortunately this value did not give a very promising result as in Figure 5. The virtual agent had successfully reached the goal, but paths produced are long with many sharp turns and a high number of time steps.

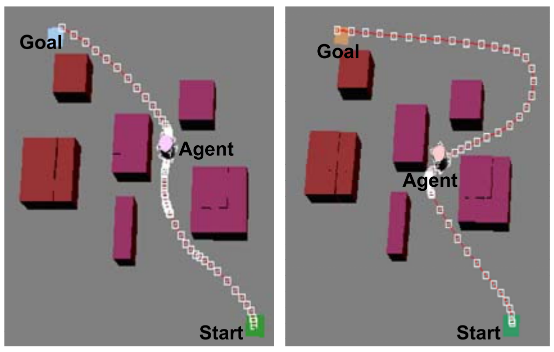

Figure 4. Different Degrees of Optimism (a) $\sigma = 0.9$ and (b) $\sigma = 0.4$.

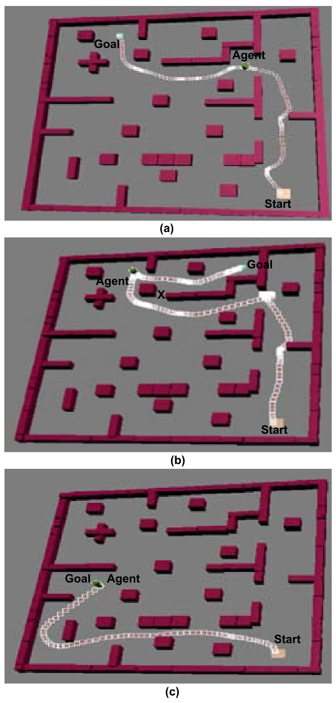

Figure 5. Navigating in combination of cluttered and maze environment ($\sigma = 0.5$).

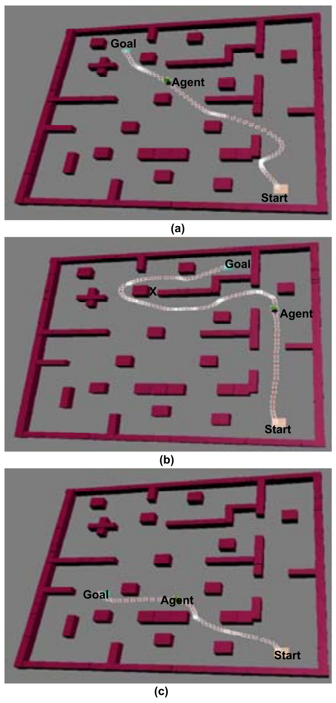

Figure 6. Navigating in combination of cluttered and maze environment ($\sigma = 0.8$).

Based on result in Figure 4, using a higher value of σ will give a better result. A new value of $\sigma = 0.8$ has been selected. Figure 6(a) and (c) produce smooth and short paths compared to the results in Figure 5(a) and (c). In Figure 6(b), the virtual agent follows a similar path compared to Figure 5(b) but with a small number of sharp turns. From the figures, we also notice that the virtual agent does not take the narrow path at **X**. One probable is that the passage is too narrow and might require a higher value of σ. However having a higher value of σ, the virtual agent might follow a longer and unsafe path.

Also in Figure 5(b) and Figure 6(b), notice that the virtual agent does not produce the shortest path. The virtual agent moves forward to the goal even though there are a walls and a dead-end. Then the virtual agent makes a left turn to escape from dead-end and follow the wall toward the goal. The virtual agent tried to reach the goal by moving straight ahead towards the goal by having a high value for Goal-Seeking behaviour. The virtual agent starts to switch to Path-Planning behaviour and Obstacle-Avoidence behaviour when it encounters an obstacle and needs to make a turn to reach the goal. This shows that the virtual agent has imitated how a human might make decisions during a navigation task in an unknown environment by making a good assumption that the path to the goal is ahead of them even though they cannot see the goal.

The experiment has shown that the σ value might vary depending on complexity of the environment. This is because some environments might have many narrow passages or two walls. With a high value of σ, the virtual agent can go through the narrow passage. Alternatively, with a low value of σ, the virtual agent might look for a wider passage. The paths produced might not be the shortest paths but they are safe paths (no collision). This is due to the ability of the virtual agent to identify its goal and the capability of the visual sensor in detecting potential obstacles.

V. Conclusions

Our visual sensor had show that how information is captures in virtual environment and identifies which part of an obstacle can be seen from the position of the virtual agent. Also, the visual sensor only identifies whether a square (cell) in the vision range is occupied by an obstacle or not. This information is critical for virtual agent in coordination its navigation task. The visual sensor also easy to integrate with our fuzzy controller. The evaluation results show that the virtual agent has deviated with minimum distance when avoiding the obstacles. The results also clearly demonstrate the mapping of inputs to output with a smooth path in a navigation task. This presents a natural way of dealing with a virtual environment without having to use a complex mathematical model.

References

[1] R. P. Darken and J. L. Sibert, "A toolset for navigation in virtual environments," in *Proceedings of the 6th annual ACM symposium on User interface software and technology*, (Atlanta, Georgia, United States), pp. 157–165, ACM Press, 1993.

[2] T. R. Wan, H. Chen, and R. Earnshaw, "Real-time path planning for navigation in unknown environment," in *Theory and Practice of Computer Graphics 2003 (TPCG 03)*, (Birmingham, UK), p. 138, IEEE Computer Society, 2003.

[3] T. Y. Li, J. M. Lien, S. Y. Chiu, and T. H. Yu, "Automatically generating virtual guided tours," in *'99 Computer Animation Conference*, (Geneva, Switz), p. 99, IEEE, 1999.

[4] S. Bandi and D. Thalmann, "Path finding for human motion in virtual environments," *Computational Geometry: Theory and Applications*, vol. 15, no. 1-3, p. 103, 2000.

[5] P. Chaudhuri, R. Khandekar, D. Sethi, and P. Kalra, "An efficient central path algorithm for virtual navigation," in *International Computer Graphics (CGI04)*, (Crete, Greece), p. 188, IEEE, 2004.

[6] M. Rook and A. Kamphuis, "Path finding using tactical information," in *Eurographics/ACM SIGGRAPH Symposium on Computer Animation (2005)* (K. Anjyo and P. Faloutsos, eds.), (Los Angeles, USA), pp. 18–19, ACM SIGGRAPH, 2005.

[7] M. Stilman and J. J. Kuffner, "Navigation among movable obstacles: Real-time reasoning in complex environments," in *4th IEEE-RAS International Conference on Humanoid Robots, 2004*, vol. 1, (Santa Monica, USA), p. 322, IEEE Inc., 2004.

[8] F. Lamarche and S. Donikian, "Crowd of virtual humans: a new approach for real time navigation in complex and structured environments," *Computer Graphics Forum*, vol. 23, no. 3, pp. 509–518, 2004.

[9] P. Tozour, "Search space representations," in *AI Game Programming Wisdom 2* (S. Rabin, ed.), USA: Charles River Media Inc., 2003.

[10] Elusive, "Omicron bot," 1998.

[11] B. Salomon, M. Garber, C. L. Ming, and M. Dinesh, "Interactive navigation in complex environments using path planning," in *2003 Symposium on Interactive 3D graphics*, (Monterey, California), pp. 41–50, ACM Press, 2003.

[12] M. Lozano and J. Molina, "A neural approach to an attentive navigation for 3D intelligent virtual agents," in *2002 IEEE International Conference on Systems, Man and Cybernetics*, vol. 6, (Hammamet, Tunisia), p. 5, IEEE, 2002.

[13] J. Velagic, B. Lacevic, and B. Perunicic, "A 3-level autonomous mobile robot navigation system designed by using reasoning/search approaches," *Robotics and Autonomous Systems*, vol. 54, no. 12, p. 989, 2006.

[14] H.-S. Seo, Y. So-Joeng, and O. Kyung-Whan, "Fuzzy reinforcement function for the intelligent agent to process vague goals," in *Annual Conference of the North American Fuzzy Information Processing Society - NAFIPS*, (Atlanta, GA, USA), p. 29, IEEE, Piscataway, NJ, USA, 2000.

[15] F. Wang, *Study of an Adaptive and Multifunctional Computational Behaviour Generation Model for Virtual Creature*. PhD thesis, University of Edinburgh, 2002.

[16] M. Piaggio, A. Sgorbissa, G. Vercelli, and R. Zaccaria, "Autonomous robot navigation using a reactive agent," in *AI*IA 97: Advances in Artificial Intelligence*, p. 96, 1997.

[17] J. J. Kuffner, *Autonomous Agent for Real-Time Animation*. PhD thesis, Stanford University, 1999.

[18] F. Wang and E. McKenzie, "A multi-agent based evolutionary artificial neural network for general navigation in unknown environments," in *Third Annual Conference on Autonomous Agents*, (Seattle, United States), pp. 154–159, ACM Press, 1999.

[19] G. Shafer, *A Mathematical Theory of Evidence*. Princeton, NJ: Princeton University Press, 1976.

[20] Y.-C. Kim, S.-B. Cho, and S.-R. Oh, "The dempster-shafer approach to map-building for an autonomous mobile robot with fuzzy controller," in *2002 AFSS International Conference on Fuzzy Systems*, vol. 2275 of *Lecture Notes in Artificial Intelligence - Advances in Soft Computing*, (Calcutta, India), p. 40, Springer-Verlag, 2002.

[21] B. Kosko, *Neural Networks and Fuzzy Systems: A Dynamical Systems Approach to Machine Intelligence*. New Jersey: Prentice-Hall inc., 1992.

[22] J. Jaafar and E. McKenzie, "A reactive architecture for autonomous agent navigation using fuzzy logic," in *Artificial Intelligence and Soft Computing 2007*, (Palma de Mallorca, Spain), pp. 57–62, IASTED, 2007.

[23] J. Jaafar and E. McKenzie, "Behaviour coordination of virtual agent navigation using fuzzy logic," in *International Conference on Fuzzy Systems (Fuzz-IEEE 2006)*, (Vancouver, Canada), pp. 1139–1145, IEEE, 2006.

[24] T. J. Ross, *Fuzzy Logic with Engineering Application*. West Sussex: John Wiley & Sons, 2 ed., 2004.

[25] L.-X. Wang, *A Course on Fuzzy System and Control*. London: Prentice-Hall International Inc., 1997.

[26] J. Jaafar, E. McKenzie, and A. Smaill, "A fuzzy action selection method for virtual agent navigation in unknown virtual environments," in *IEEE International Conference on Fuzzy Systems*, (London), pp. 1–6, IEEE, 2007.