International Journal of Automotive and Mechanical Engineering (IJAME)
ISSN: 2229-8649 (Print); ISSN: 2180-1606 (Online); Volume 2, pp. 171-180, July-December 2010
©Universiti Malaysia Pahang
DOI: http://dx.doi.org/10.15282/ijame.2.2010.6.0014

MOTION MODELLING USING CONCEPTS OF FUZZY ARTIFICIAL POTENTIAL FIELDS

O. Motlagh[1], A.R. Ramli[1], S.H. Tang[2], F. Motlagh[1] and N. Ismail[2]

[1]Intelligent Systems and Robotic Laboratory, Institute of Advanced Technology
[2]Department of Mechanical and Manufacturing Engineering,
Faculty of Engineering, University Putra Malaysia (UPM)
43400 Serdang, Selangor, Malaysia
Email: motlagh7@gmail.com, arr@eng.upm.edu.my

ABSTRACT

Artificial potential fields(APF) are well established for reactive navigation of mobile robots. This paper describes a fast and robust fuzzy-APF on an ActivMedia AmigoBot.Obstacle-related information is fuzzified by using sensory fusion, which results in a shorter runtime. In addition, the membership functions of obstacle direction and range have been merged into one function, obtaining a smaller block of rules. The system is tested in virtual environments with non-concave obstacles. Then, the paper describes a new approach to motion modelling where the motion of intelligent travellers is modelled by consecutive path segments. In previous work, the authors described a reliable motion modelling technique using causal inference of fuzzy cognitive maps (FCM) which has been efficiently modified for the purpose of this contribution. Results and analysis are given to demonstrate the efficiency and accuracy of the proposed motion modelling algorithm.

Keywords: Motion Modelling; Artificial Intelligence; Potential Fields

INTRODUCTION

In local navigation, the task is to locally explore the space, seeking a designated target while avoiding obstacles or certain regions. The path planning strategy adopted by a reactive mobile robot is therefore based on route-like spatial cognition through sensory information and motion control. Quite a number of different algorithms have been developed for reactive motion of mobile robots including: virtual wall (Ordonez et al., 2007), graph-based method (Kelarev, 2003), landmark learning (Krishna and Kalra, 2001), fuzzy logic minimum risk (Wang and Liu, 2007), target switching strategy (Xu and Tso, 1999; Motlagh et al., 2009a), 3-step potential field (Tu and Baltes, 2006), and many others. A summary of the recent literature is given in (Motlagh et al., 2009a). One of the established reactive techniques is based on using artificial potential fields (APF) (Khatib, 1986). The APF strategy is to make a compromise between target seeking behavior, with the target's attracting the robot, and obstacle avoidance behaviour, with the obstacle's repelling the robot. In fuzzy-APF, the inputs of obstacle and target orientations are fuzzified and given to usually a rule-based inference engine for generating control outputs. Ideally, in spaces with non-concave obstacles or culdesac, a robot working under an APF algorithm can simply navigate towards the target.

THE FUZZY-APF ALGORITHM

The ActivMedia AmigoBot was chosen for the mobile robot of this project as shown in Figure 1. A fast fuzzy-APF algorithm is developed for reactive navigation of the robot in environments with only non-concave obstacles. The robot is equipped with sonar range-finders and self-localization sensors for online local navigation, i.e., APF with one known target (attractor) and unknown obstacles (repeller).The robot working under APF is regarded as an example of an intelligent traveller. The robot brain (a microprocessor or PC) can be actually connected to the platform via wired or wireless connection. However, in this project, a standard ActivMedia simulator has been used for virtual experiments.

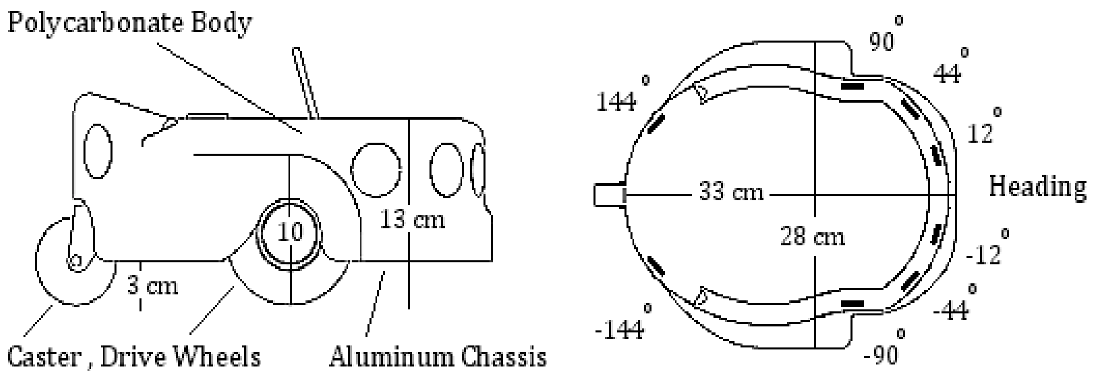

Figure1. ActivMedia AmigoBot used as an intelligent traveller

A rule-based strategy is developed for APF navigation of the AmigoBot. As shown in Table 1,twelve rules are defined to coordinate obstacle avoidance and target seeking behaviours. Figure 2 depicts the fuzzy states of the input membership functions of obstacle and target, including: obstacle at left (OL), in front (OF), at right (OR); target at left (TL), in front (TF), at right (TR). These are all relative to the robot's heading direction. There is another state of having no-obstacle shown by (NO) for the short-sighted robot. The state of no-obstacle allows for merging all of the obstacle-related information including distance and directional information, i.e., by satisfying OL+OF+OR+NO=1.The obstacle-related information is therefore fuzzified only by fusion of sensory information. This results in a smaller size of the rule-set and therefore a shorter runtime.

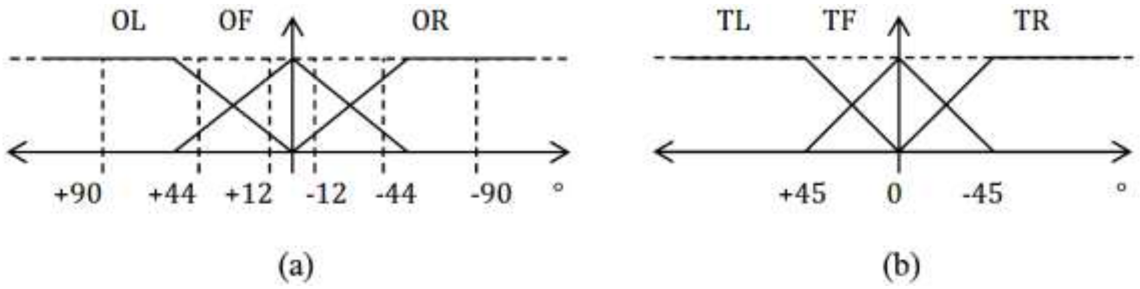

Figure 2. The membership functions of relative obstacle and target orientations

Table 1. Fuzzy rules R1 through R12

R1:	If	OL	&	TL	Then,	VL	,	SR
R2:	If	OL	&	TF	Then,	VU	,	SR
R3:	If	OL	&	TR	Then,	VH	,	SR
R4:	If	OF	&	TL	Then,	VL	,	SL
R5:	If	OF	&	TF	Then,	VL	,	-
R6:	If	OF	&	TR	Then,	VL	,	SR
R7:	If	OR	&	TL	Then,	VH	,	SL
R8:	If	OR	&	TF	Then,	VU	,	SL
R9:	If	OR	&	TR	Then,	VL	,	SL
R10:	If	NO	&	TL	Then,	VH	,	SL
R11:	If	NO	&	TF	Then,	VH	,	SU
R12:	If	NO	&	TR	Then,	VH	,	SR

The inference mechanism of Table 1 accounts for the brain of the robot. The control outputs are the linear velocity (v) and the steering (s) to be applied to the robot's actuators. Velocity might get low (VL), high (VH), or (VU) for unchanged velocity. Steering control is represented by (SL) for turning to left, (SR) for turning to right, or (SU) for no-change. From calculation of the rules, aggregation, and defuzzification, the outputs are derived (Eq. 1 and Eq. 2) to be applied to the actuators, i.e., the left and right wheels' motors. This is similar to intelligent and biological systems where the brain makes muscles and bones move. The APF of this project is expert-defined. However, it provides a fair level of intelligence for unguided way finding of the robot. It is stressed that there must be only non-concave obstacles present in the environment. More complex situations are described by the authors using an extended APF algorithm for tackling concave and cul-de-sac obstacles (Motlagh et al., 2009a).

$$v = \frac{vl\sum VL + vn\sum VU + vh\sum VH}{\sum VL + \sum VU + \sum VH} \tag{1}$$

where *vl*, *vn*, *vh* are output constants

$$s = \frac{sl\sum SL + sn\sum SU + sr\sum SR}{\sum SL + \sum SU + \sum SR} \tag{2}$$

where *sl*, *sn*, *sr* are output constants

As depicted in Figure 3, there are several trajectory examples available from experimental results using the standard simulation software of ActivMedia. At each time instance, the fuzzified values of obstacle and target orientations are given to the rule-based inference mechanism. The outputs of velocity and steering are then derived, defuzzified, and applied to the robot. Accordingly, the robot efficiently approaches the target while avoiding the obstacle.

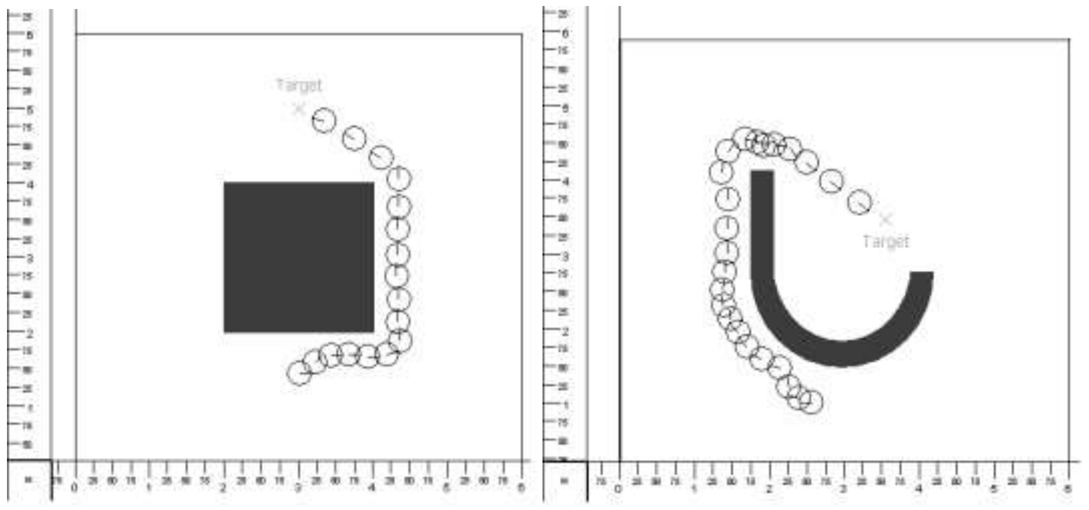

Figure 3. Sample trajectories obtained under the developed fuzzy-APF control

APF-BASED MOTION MODELLING

It was shown in the previous section that reactive motion of mobile robots can be governed by artificial potential fields (APF). The robot's trajectory is made up from motion productions, i.e., path segments. Taking a robot working under an APF as an intelligent traveller, there are decisions made for generation of path segments at time instances, i.e., decision productions. Each path segment or production P_i can be shown by a displacement vector with length d_i and a change in heading direction θ_i. Alternatively, to reduce the effect of sampling frequency (timing), the length of a production P_i can be defined in terms of the robot's linear velocity (v). The motion production is therefore defined $P_i = (v_i , \theta_i)$ as shown in Figure 4.

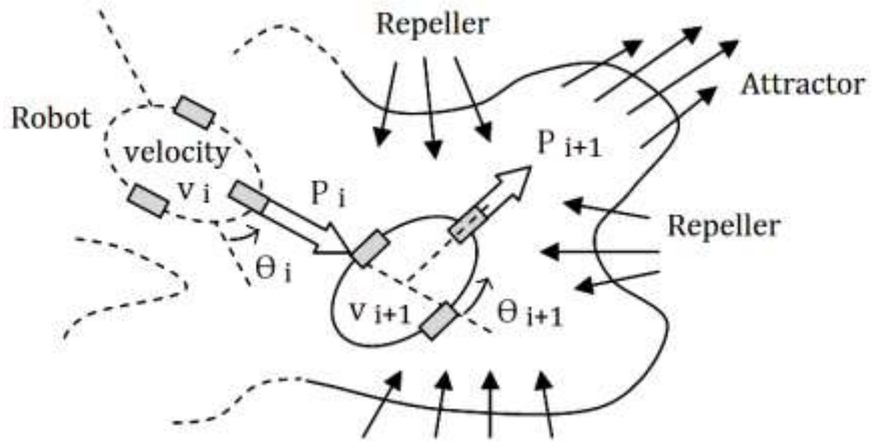

Figure 4. Series of motion productions generated due to consecutive decisions for motion: the robot enters a sub-space (i) with self-kinematics of production P_i. It then makes a decision causing motion production P_{i+1}.

In this article, it is proposed and shown that from observation of successive motion productions, corresponding decision productions can be extracted. The goal is to create a model to replicate the way finding behaviours of the robot. For the scenario of Figure 4, it is supposed that the robot's control algorithm is unknown. However, the robot's self-kinematical and the environmental information such as exerted forces are known. Therefore, at any time t_i, the robot's velocity and heading are available as v_i (linear velocity), and θ_i (heading angle in form of $\theta L_i = s_i$ where $0 < s_i < +180$, and $\theta R_i = -s_i$ where $0 > s_i > -180$ which are the amounts of turn angle to left or right) revealing the self-kinematical information. In addition, the fuzzy states of the obstacle and target at time t_i or $(OL, OF, OR, NO, TL, TF, TR)_i$ reveal the environmental information. Therefore, there are ten motion concepts describing the robot's status at any time. These are the inputs to the robot for making a decision for the following motion production (output of v_{i+1} and θ_{i+1}). The key point is to discover what decision has been made by the robot by looking at these inputs (at t_i) and outputs (at t_{i+1}).

PROPOSED DECISION MODELLING ALGORITHM

The decision modelling strategy is based on the capabilities of the fuzzy cognitive map (FCM) in a causal inference mechanism. FCM (Kosko, 1992; 1996; Stylios et al., 2008) is a graph-like soft computing tool that is based on fuzzy logic (FL) and neural network (NN) methodologies. FCM is widely used in all areas of AI using complementary learning techniques. Figure 5 shows how the ten specified motion concepts are used to define the FCM's concepts (nodes), i.e., c_1 ... c_{10}, while on the other hand, the graph of causal interactions, i.e., matrix of FCM's events or edges $\{e_{1,1} \dots e_{1,10} ; \dots ; e_{10,1} \dots e_{10,10}\}$, is used to represent the decisional behaviours.

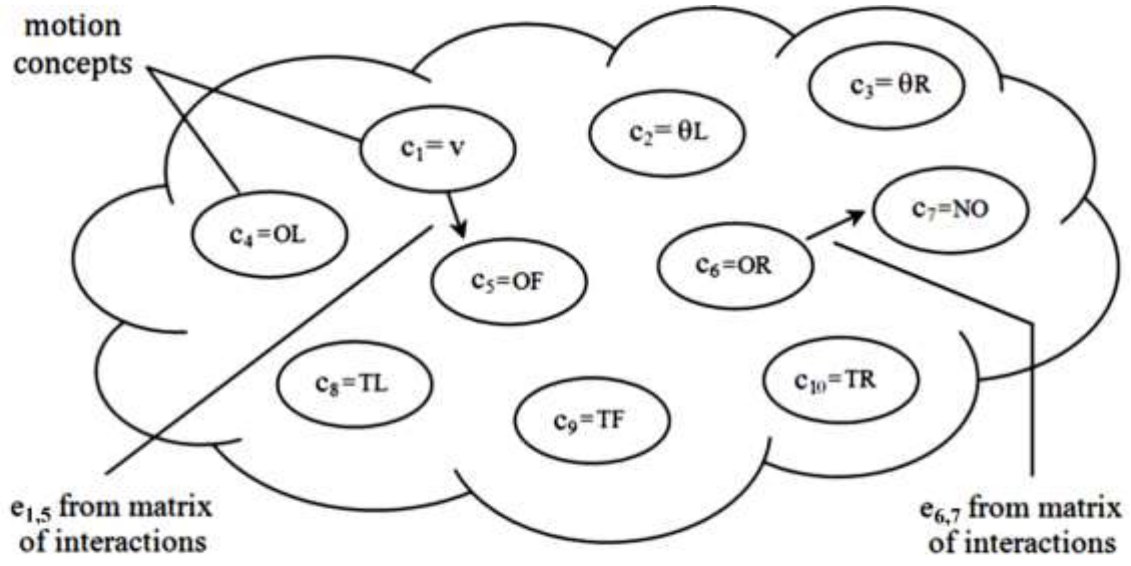

Figure 5. The FCM's 10×10 matrix of edges is used for decision modelling

FCM-based inference works in a cyclic fashion in such a way that during each cycle the weights of the affecting concepts are multiplied by the weights of their causes on the affected concepts. Summations of these amounts are then squashed into a certain range using a logistic function and assigned to the affected concepts as their new weights. The difference between the formula of *definition* (Eq. 3) and *incremental* is in defining the initial weights of the concepts (McNeill and Thro, 1994). In the definition method, each concept weight c_j is completely defined anew during each forward step (cycle). Incremental method adds each concept value to its previous value during each forward step of the dynamic map. FCM is run for an unlimited number of cycles until convergence, i.e., formation of a decision. The finalized concepts' weights represent the decision output of the FCM. FCM is therefore used to anticipate a future action, i.e., the successive path segment or motion production P_{i+1}.

$$c_j = \left(1 + exp\left(-\sum_{i=1}^{10}(c_i \times e_{i,j}) \times gain\right)\right)^{-1} \quad , \quad j = 1 \dots 10 \tag{3}$$

Now if the anticipated P_{i+1} happens to be similar to the one actually made by the robot in terms of steering (s) and velocity (v), then the FCM has just replicated the robot's decision making behavior. And the related decision matrix (the matrix of FCM edges), accounts for that specific decision for making P_{i+1} after P_i. The problem to tackle is to discover which set of event weights or matrix of interactions could make this come about. In other words, which weights have to be assigned to $\{e_{1,1} \dots e_{10,10}\}$ in order for the FCM to anticipate P_{i+1} correctly. This problem is resolved using a supervised learning strategy for determining the FCM event weights at the different times. There are many approaches to training FCM models (Konar and Chakraborty, 2005; Stach et al., 2005; Ghazanfari et al., 2007). Heuristic search and the genetic algorithm (GA) are widely used as FCM training and tuning techniques (Stach et al., 2005; Motlagh et al., 2009b) through supervised learning of the event matrix.

Each event weight implies two pieces of information in a single trait. One is called causality, and the other, the intensity of the event. In the graphical representation, an FCM event is shown by a signed weighted edge starting from the affecting node and pointing towards the affected one. For example, the causal effect of concept c_1 on c_2 can be shown by $e_{1,2}$= -0.45 in the range (-1, 1). This means c_1 has a negative influence on c_2, with 0.45 units of intensity. When necessary, weights can be magnified into other ranges (-b, b) while the gain of the FCM formula must be set to (1/b). In this method, the range of the weights was chosen to be (-100, 100) for efficient binary computation, i.e., assigning only 1 byte per event instead of 4 bytes as in the floating-point format. Accordingly, the event matrix is represented by 10×10=100 event weights within (-100, 100) where each weight is encoded as one signed byte (8 binary bits within range of -127, +127). For example, $e_{1,2}$ can be represented by -101101 or 10101101.

Having FCM events ordered as $\{e_{1,1} \dots e_{1,10}, e_{2,1} \dots e_{2,10}, \dots, e_{10,1} \dots e_{10,10}\}$, a single binary string with length 100 bytes is used to record their weights in the same order. In fact, this binary string contains the entire event matrix. It is therefore an overall representation of the decisional behaviours shown by the robot. Such information plays the key role in the FCM-based decision making. Therefore, by loading different binary values into this string, different motion productions can be generated by the FCM. The goal is to find the fittest binary values, i.e., chromosomes containing 100 event weights, by which the FCM can generate the same decision productions as actually generated by the robot at each decision point. A genetic algorithm (GA) is used to obtain the fittest decision productions. The binary string of encoded event weights is used as a chromosome. For example {01000100, 10001011 … 00010101} represents $\{e_{1,1}$= +68, $e_{1,2}$= -11, … $e_{10,10}$= +21}.

Production P_{i+1} is used as the GA cost function for fitness scoring. The FCM learns to generate decisions by which it can replicate the same motion productions made by the robot. In order to train the FCM, the initial population (N chromosomes) is generated. Then, the following steps are repeated until a solution is found which accounts for the fittest decision trial, i.e., matrix of edges. At first, each chromosome is fed to the FCM which means it is decoded and assigned to the matrix of FCM events. The FCM is then run to generate a decision production. The decision production is used to predict P_{i+1}. According to the amount of error between the actual and the predicted P_{i+1}, a fitness score is assigned to each chromosome. Then, elites will be selected for reproduction from the current population. The chance of being selected is proportional to the chromosomes' fitness (roulette wheel selection). And, finally, through crossover and mutation, a new generation is created for the next GA run.

NAVIGATION RESULTS USING DECISION MATRX

The following pseudocode shows much of the learning algorithm. The first WHILE loop evaluates the GA-trails of event matrices by feeding them into the FCM and measuring the fitness of the resulted outputs. The second WHILE loop show how FCM is run for generating decisions for motion.

```
concept = CALL   motion concepts (i)
CALL   reproduction
WHILE   n < population
        Decision_trial = chromosome (n)
        event_matrix = CALL   decode_decision_trial
        CALL   run_FCM for prediction of motion concepts (i+1)
        CALL   fitness
END WHILE
…
run_FCM
{
  WHILE   cycle <very large limit
        convergence = concept
        FOR   c = 1 to 10
              total_effect = COMPUTE total effect on concept (c)
              concept (c) = CALL squash_total_effect
        END FOR
        IF   concept = convergence
        THEN   BREAK WHILE
        END IF
  END WHILE
}
```

The performance of the developed decision modelling system has been evaluated using ActivMedia and MATLAB simulation. The programming and experimental work were carried out from June to September 2009. At first, the AmigoBot was put to explore several virtual environments under the control of the developed fuzzy-APF algorithm as described in Section 2, (Figure 3). While the robot was navigating in the environments, at time instances, the fuzzified states of obstacle and target orientation, i.e., OL, OF, OR, NO, TL, TF, TR, the amounts of steering, i.e., θL and θR or changes in heading direction to left and right, and the velocity (V) were extracted from the ActivMedia simulator and recorded into a data-base. Therefore, each record of the data-base represented the environmental and self-kinematical information of a single motion production as shown in Table 2.

Table 2. Tracking and recording the values of the robot motion concepts at time instances

Motion concepts at subspace				
Self-kinematical	Obstacle-relates	Target related	Decision production	For moving to subspace
---	---	---	P_i?	i
$(c_1,c_2,c_3$	c_4,c_5,c_6, c_7	$c_8,c_9,c_{10})_i$	P_{i+1}?	i+1
$(c_1,c_2,c_3$	c_4,c_5,c_6, c_7	$(c_8,c_9,c_{10})_{i+1}$	P_{i+2}?	i+2
---	---	---	---	---

For learning each of the available paths of Figure 3, the corresponding data-base (sequence of motion productions) is loaded into the decision modelling software developed in MATLAB. For learning each segment of the loaded path, two records of the data-base have to be used: one for loading the initial weights of the FCM nodes, and another for GA fitness test. Accordingly, a new data-base of decision matrices was obtained. Each decision matrix accounts for robot behaviour in moving from one sub-space to another. In order to test the sanity of the matrix-like decisions a new motion planning software was created in the ActivMedia simulator. However, this time instead of using APF-based navigation algorithm, the data-base of decision matrices was used for motion planning and robot navigation.

Figure 6 shows a sample trajectory obtained from ActivMedia under control of the data-base of decision matrices. At each sub-space, e.g., robot at a point in the vicinity of the obstacle, all of the ten motion concepts describing self-kinematical and environmental factors are detected. These are then compared against the data-base from which the corresponding decision matrix is selected and retrieved. An FCM is then run using the available motion concepts (loaded into its nodes) and the selected decision matrix (loaded into its edges). Upon FCM convergence, a motion production (v, s:(θL, θR)) is generated for navigating the robot towards the following sub-space.

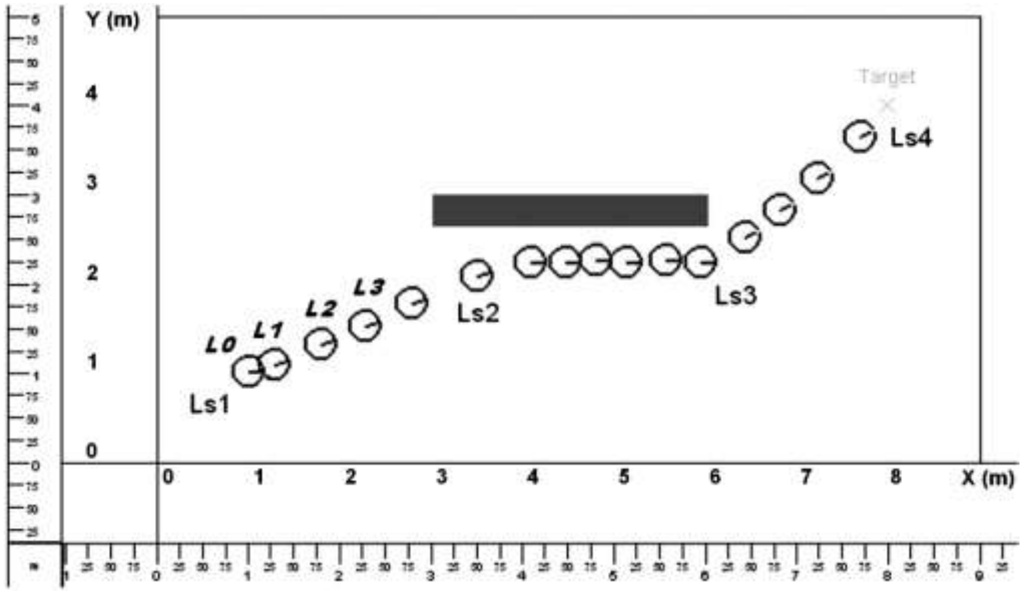

Figure 6. A trajectory made under supply of decision matrices

Each sub-space or location along the generated trajectory is represented in *x-y* coordinates, e.g., L_i:$(X_{DM},Y_{DM})_i$. Therefore, L0, L1 … are the successive sub-spaces while Ls1, Ls2 … are the points of major switchesin behaviors, e.g., from wall following to target seeking, etc. With a high precision of up to 90% through more experiments, the path generated by this method (data-base of decision matrices) is identical with the one generated by the fuzzy-APF algorithm in the ActivMedia simulator. There are two approaches for the generation of error graphs from comparison of the path in Figure 6 with the path generated under APF control in the same environment. In the one-to-one method, the robot's 2D coordinates are compared at time instances (Eq. 4) where the paths generated under decision matrices and the fuzzy-APF control are called DM and APF, respectively. As the second approach, the amount of the augmented error is obtained using the Π function of Eq. 5 which gives the overall amount of error (E_t) between the two paths over elapsed time (t).

$$error_i = \sqrt{(x_{APF} - x_{DM})_i^2 + (y_{APF} - y_{DM})_i^2} \quad (4)$$

whereT_{DM},T_{APF} arethedurationsofthe two trajectories

$$E_t = \prod_{i=0}^{t} error_i \quad , \quad t \in \{0 \ldots T\} \ , T = \min(T_{DM}, T_{APF}) \quad (5)$$

CONCLUSION

A fast and robust motion control system was presented for mobile robots in environments with non-concave obstacles. There are two types of motion concepts in APF-based navigation. Self-kinematical concepts include linear velocity and steering to left and right. Environmental concepts include target and obstacle information, i.e., fuzzy states of relative orientation at left, front, and right. A new decision modelling technique was described based on the concepts of fuzzy-APF. The motion concepts are used to set the weights of FCM nodes. The weights of the FCM edges are tuned using GA-based supervised learning. The decisions for moving along sub-spaces one after another are determined and recorded as matrices. Using such a data-base of decision matrices, the robot motion can be controlled in any environment of the same type. The decision model of this research is in progress to be applicable to all kinds of intelligent travellers, including human subjects. Future applications are in path prediction, security, and surveillance, where the prediction of intelligent motion is required.

ACKNOWLEDGEMENT

This research has been partially supported by research university grant scheme No. 5/1/7/0222RU at University Putra Malaysia. The authors therefore wish to convey their thanks to the Ministry which provided this financial support.

REFERENCES

Ghazanfari, M., Alizadeh, S., Fathian, M. and Koulouriotis, D.E. (2007) Comparing Simulated Annealing and Genetic Algorithm in Learning FCM. *Applied Mathematics and Computation*, 192(1): 56–68.

Kelarev, A.V. (2003)*Graph Algebras and Automata*. New York: Marcel Dekker.

Khatib, O. (1986) Real-Time Obstacle Avoidance for Manipulators and Mobile Robots. *International Journal of Robotics Research*, 5(1): 90–98.

Konar, A. and Chakraborty, U.K. (2005) Reasoning and Unsupervised Learning in a Fuzzy Cognitive Map. *Information Sciences*, 170: 419–441.

Kosko, B. (1992) Neural Networks and Fuzzy Systems: A Dynamical Systems Approach to Machine Intelligence. Upper Saddle River, NJ: Prentice-Hall.

Kosko, B. (1996)*Fuzzy Engineering*. Upper Saddle River, NJ: Prentice-Hall.

Krishna, K.M. and Kalra, P.K. (2001) Perception and Remembrance of the environment During Real-Time Navigation of a Mobile Robot. *Robotics and Autonomous Systems*, 37: 25–51.

McNeill, F.M. and Thro, E. (1994) Fuzzy Logic a Practical Approach. San Diego,CA: Academic Press.

Motlagh, O., Tang, S.H. and Ismail, N. (2009a) Development of a New Minimum Avoidance System for a Behavio4r-Based Mobile Robot. *Fuzzy Sets and Systems*, 160(13): 1929–1946.

Motlagh, O., Tang, S.H., Ismail, N. and Ramli, A.R. (2009b) A Review on Positioning Techniques and Technologies:ANovel AI Approach. *Journal of Applied Sciences*, 9(9): 1601–1614.

Ordonez, C., Collins Jr., E.G., Selekwa, M.F. and Dunlap, D.D. (2007) The Virtual Wall Approach to Limit Cycle Avoidance for Unmanned Ground Vehicles. *Robotics and Autonomous Systems*, 56(8): 645–657.

Stach, W., Kurgan, L., Pedrycz, W. and Reformat, M. (2005) Genetic Learning of Fuzzy Cognitive Maps. *Fuzzy Sets and Systems,* 153: 371–401.

Stylios, C.D., Georgopoulos, V.C. and Malandraki, G.A. (2008) Fuzzy Cognitive Map Architectures for Medical Decision Support Systems. *Applied Soft Computing,* 8(3): 1243–1251.

Tu, K.-Y. and Baltes, J. (2006) Fuzzy Potential Energy for a Map Approach to Robot Navigation. *Robotics and Autonomous Systems*, 54(7): 574–589.

Wang, M. and Liu, J.N.K. (2007) Fuzzy Logic-Based Real-Time Robot Navigation in Unknown Environment With Dead Ends. *Robotics and Autonomous Systems,* 56(7): 625–643.

Xu, W.L. and Tso, S.K. (1999) Sensor-based Fuzzy Reactive Navigation for a Mobile Robot Through Local Target Switching. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 29(3): 451–459.