Fuzzy Motivations for Evolutionary Behavior Learning by a Mobile Robot

Tomás Arredondo V., Wolfgang Freund, Cesar Muñoz, Nicolas Navarro and Fernando Quirós

Universidad Técnica Federico Santa María, Valparaíso, Chile,
Departamento de Electrónica,
Casilla 110 V, Valparaíso, Chile,
tarredondo@elo.utfsm.cl

Abstract. In this paper we describe a fuzzy logic based approach for providing biologically based motivations to be used in evolutionary mobile robot learning. Takagi-Sugeno-Kang (TSK) fuzzy logic is used to motivate a small mobile robot to acquire complex behaviors and to perform environment recognition. This method is implemented and tested in behavior based navigation and action sequence based environment recognition tasks in a Khepera mobile robot simulator. Our fuzzy logic based motivation technique is shown as a simple and powerful method for a robot to acquire a diverse set of fit behaviors as well as providing an intuitive user interface framework.

Keywords: Fuzzy logic, evolutionary, mobile robot, environment recognition, AEM.

1 Introduction

Providing more natural and intuitive interfaces between robots and people is clearly seen as desirable and beneficial. Much recent research has focused on providing more intuitive and natural interfaces for robotic control [1, 2].

Towards this goal we have developed a fuzzy logic based method that provides a natural interface in order to give a variety of motivations used in robotic learning. To test the validity of the proposed method we tested the fuzzy logic based method on behavior based navigation and environment recognition tasks within a Khepera robot simulator. The results show that the method has potential for improving human understanding of robotic behavior learning as well as provides a method for generating greater diversity of robotic behaviors. To the best of our knowledge fuzzy logic has not been used before in this manner in a robotics application.

In our experiments we studied two different tasks for testing with fuzzy based motivations in the Khepera robot simulator: basic behavior based navigation and action based environmental modeling (AEM).

Behavior based architectures (e.g. subsumption) in general do not use world models, representative or symbolic knowledge and there is a tight coupling between sensing and action (moving). This design philosophy promotes the idea that robots should be inexpensive, robust to sensor and other noise, incremental, uncalibrated and without complex computers and communication systems. Planning actions based on internal world representations in not seen as something beneficial because of its inherent error and associated costs. Behavior based learning systems typically used include reinforcement learning, neural networks, genetic algorithms, fuzzy systems, case and memory based learning [3, 4]. Behavior based navigation as implemented in the Khepera simulator YAKS [5] inputs sensor values directly into a neural network that drives left and right motors for navigation in different rooms.

Action-based environmental modeling (AEM) also follows this less is more philosophy by using a simple mobile robot with local sensors in order to navigate and perform environment recognition in various scenarios (rooms). AEM uses a small action set (e.g. go straight, turn left, turn right, turn around) in order to perform a sequence of actions based on sensed states in a specific environment. The search space of suitable behaviors is huge and designing suitable behaviors by hand is very difficult therefore Yamada [6] has used a genetic algorithm within a Khepera simulator to find suitable behaviors for AEM.

In the training phase of the AEM procedure and for each room of the set of rooms being recognized the robot executes an action sequence which is converted into an environment vector. These vectors are repeatedly fed into a SOM [18] network in order for the neural network to learn without supervision which is the output node (r-node) that corresponds to each room. The environment vector used for each room and the winning output node is also stored as a room instance. The next step is a test phase in which the robot executes an action sequence in one of the rooms previously used and the r-node for the test room is determined using the previously trained SOM network. The robot then determines which room used during training has the minimum distance to the current test room by using 1-Nearest Neighbor with Euclidean distance. Fitness considerations include returning to his original starting neighborhood (homing), ability to identify the room it is in (accuracy), and using the shortest possible sequence (efficiency). Collision avoidance is implicit in the efficiency measure; the final fitness is the sum of all three [6].

In Section 2, we describe fuzzy logic in robotic control. Our method and how it was implemented is described in Section 3. In Section 4 and 5 we describe and summarize our test results. Finally, in Section 6 some conclusions are drawn.

2 Fuzzy Logic in Robotic Behavioral Control

Fuzzy logic systems produce actions using a set of fuzzy rules and variables. These variables are referred to as linguistic variables and indicate a degree of membership of the variable to a particular fuzzy set. The degree of membership (between 0 and 1) is defined by a membership function which maps a crisp input to a fuzzy output (fuzzifier). Fuzzy logic control systems consist of: fuzzifier, fuzzy rule base, fuzzy inference engine, and a defuzzifier. In many applications fuzzy logic provides a more natural interface that gives greater flexibility than traditional logic [7].

Fuzzy logic has been used widely in various robotic applications such as robotics behavioral fusion. Flakey and Marge are two robots that used fuzzy based implementations to blend different possible behaviors for things such as collision avoidance, goal seeking, docking, and wall following [8, 9]. These robots had issues with scalability and exponential growth in the rule base which they attempted to manage by: reduction of input spaces and using contexts with a limited world model [8], or by using independent distributed fuzzy agents and weighted vector summation via fuzzy multiplexers for producing the final command signals for its drive and steer mechanism [9]. More recently other fuzzy logic strategies have been used in mobile robotics including: neuro-fuzzy controllers for behavior design (based on direction, distance and steering) [10], fuzzy based modular motion planning [11], fuzzy integration of groups of behaviors [12], multiple fuzzy agents used in behavior fusion [13], GA based neuro fuzzy reinforcement learning agents used in training a walking robot [14], and behavior based fuzzy logic integration for robotic navigation in challenging terrain [15]. As far as our research shows fuzzy logic has not been previously used in robotics in terms of motivating actions and behaviors. We have implemented such motivations as fuzzy fitness functions for robotic behaviors.

3 Fuzzy Motivations for Robotic Learning

Motivation as currently viewed by psychologists as an internal state or condition (e.g. a need, desire or want) that serves to influence the intensity and direction of behavior. Motivation is generally accepted as involved in the performance of learned behaviors. That is a learned behavior may not occur unless it's driven by a motivation. There are many sources for motivations including: behavioral, social, biological, cognitive, affective, and spiritual [16]. Differences in motivations are key drivers in helping to produce a variety of behaviors which have a high degree of benefit (or fitness) for the organism.

In our experiments, we use motivation settings in order to determine the fuzzy fitness of a robot in various environments. In terms of robotic learning the motivations that we consider include: curiosity (C), homing (H), orientation (O), and energy (E, the opposite of laziness). The membership functions used for each of the four motivations in our experiment is shown in Fig. 1.

The Takagi-Sugeno-Kang (TSK) fuzzy logic model is used, TSK fuzzy logic does not require deffuzification as each rule has a crisp output that is aggregated as a weighted average [17]. As shown in Fig. 2, the fuzzy motivations considered include the parameters of C, H, O and E, which are used as input settings (between 0 and 1) prior to running each experiment, with the sum kept at one. A run environment (room) is selected and the GA initial robot population is randomly initialized. After this, each robot in the population performs its task (navigation and optionally environment recognition) and a set of fitness values (a, g, l, b) corresponding to the performed task are obtained.

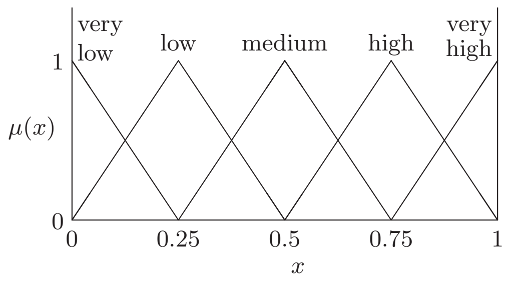

Fig. 1. Fuzzy membership functions

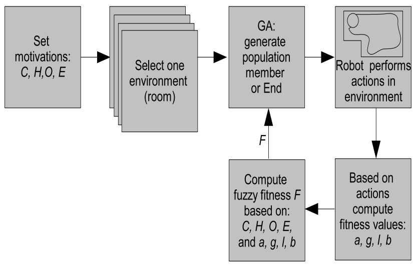
Set motivations: C, H,O, E
Select one environment (room)
GA: generate population member or End
Robot performs actions in environment
Based on actions compute fitness values: a, g, l, b
Compute fuzzy fitness F based on: C, H, O, E, and a, g, l, b
F

Fig. 2. System Overview

The fitness criteria and the variables that correspond to them are: amount of area explored (a), proper action termination and escape from original neighborhood area (g), environment recognition (l) and percent of battery usage (b). These fitness values are calculated after the robot completes each run. The a value is determined by considering the percentage area explored relative to the optimum, g is determined by

$$g = 1 - \frac{\text{final distance to the robot home}}{\text{maximum possible distance}},$$

l is currently not used and determined only off-line, finally b is the estimated total energy consumption of the robot considering each step.

The final fuzzy motivation fitness value (F) is calculated using TSK based fuzzy logic (four fuzzy variables with five membership functions each: $4^5 = 1024$ different fuzzy rules) as shown in Fig. 3 and using the membership functions from Fig. 1 to compute μ values. For the coefficient array C we used a linear function. A sample fuzzy rule (number 10) is given as follows:

```
if (Y[1] == V.H.) and (Y[2] == L) and (Y[3] == V.L.) and (Y[4] == V.L.) then
  f[10] = X[1]Y[1]C[5]+X[2]Y[2]C[2]+X[3]Y[3]C[1]+X[4]Y[4]C[1]
```

```
**Algorithm FuzzyFitness**
**Input:**
  $N$ :  number of fuzzy motivations;
  $M$ :  number of membership functions per motivation;
  $X[N]$ :  array of motivation values preset;
  $Y[N]$ :  array of fitness values;
  $C[N]$ :  array of coefficients;
  $\mu[N][M]$ :  matrix of membership values for each motivation;
**Variables:**
  $w[n]$ :  the weight for each fuzzy rule being evaluated;
  $f[n]$ :  the estimated fitness;
  $n, x_0, x_1, \ldots, x_N$ :  integers;
**Output:**
  $F$ :  the fuzzy fitness value calculated;
**begin**
  $n := 1$;
  **for each** $x_1, x_2, \ldots, x_N := 1$ **step** 1 **until** $M$ **do**
  **begin**
    $w[n] := \min\{\mu[1][x_1], \mu[2][x_2], \ldots, \mu[N][x_N]\}$;
    $f[n] := \sum_{i=1}^{N} X[i]Y[i]C[x_i]$;
    $n := n + 1$;
  **end**;
  $F := (\sum_{i=1}^{N^M} w[i]f[i])/(\sum_{i=1}^{N^M} w[i])$;
**end**;
```

Fig. 3. Fuzzy Fitness Algorithm

3.1 Implementation Environment

We used the simulator YAKS (Yet Another Khepera Simulator) for our implementation. YAKS is a simple open source behavior based simulator [5] that uses neural networks and genetic algorithms in order to provide a navigation environment for a Khepera robot. Sensor inputs are directly provided into a multilayer neural network in order to drive left and right wheel motors. A simple genetic algorithm is used with 200 members, 200 generations, mutation of 1%, and elite reproduction. Random noise (6%) is injected into motor actions and sensors to improve realism. The *GA* provides with a mechanism for updating neural network weights used by each robot in the population that is being optimized. Figure 4 shows where the fuzzy fitness algorithm and AEM are incorporated into the simulator.

As seen in Fig. 5, to implement AEM, we select a highly fit robot (corresponding to the neural network in Fig. 4) and make him navigate in all environments (rooms). This navigation produces actions which are saved as action sequences. These action sequences are converted using chain coding into an environment vector [6]. These vectors are fed into the SOM network for unsupervised learn-

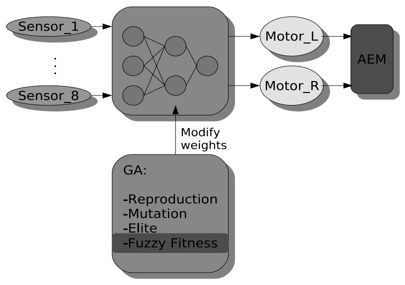

Fig. 4. Fuzzy fitness and AEM implementation

ing. After learning the SOM network associates a vector with one of its output nodes (r-nodes).

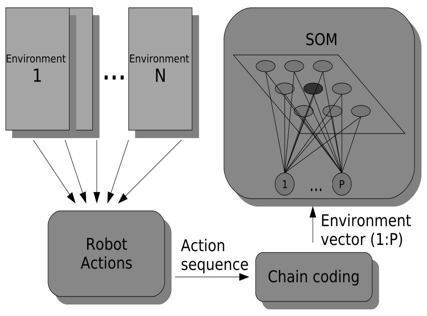

Fig. 5. AEM Overview

For our implementation of SOM we select an input of 400 actions (steps) and a linear output layer of 128 nodes. The actions are obtained by converting each motor command (Motor_L and Motor_R) which is a real value between 0 and 1 into an action (left 30°, right 30°, turn 180°, go straight).

After training SOM we evaluate it by using the same robot and at random select a room for him to navigate in. The SOM network is evaluated by the ability of the robot to recognize which room (r-node) it navigates in. This ability could be fed as the orientation value in the fuzzy fitness calculation.

4 Navigation and Environment Recognition Experimental Results

4.1 Scenario 1: Navigation

We tested behavior based navigation in five different rooms. For comparison and as an example, we selected two sets of fuzzy motivation criteria. This scenario did not include environment recognition (AEM). In order to contrast the effect of curiosity, the first criterion (C, H, O, E) used was (.8, .1, 0, .1) and the second was (.5, .3, 0, .2). We stopped each experiment after 400 steps. As seen in the examples of Fig. 6, during the simulations performed average fuzzy fitness in the first generation was 0.1 and gradually increased. Figures 7 and 8 show some representative results of navigation by various robots after fuzzy fitness optimization.

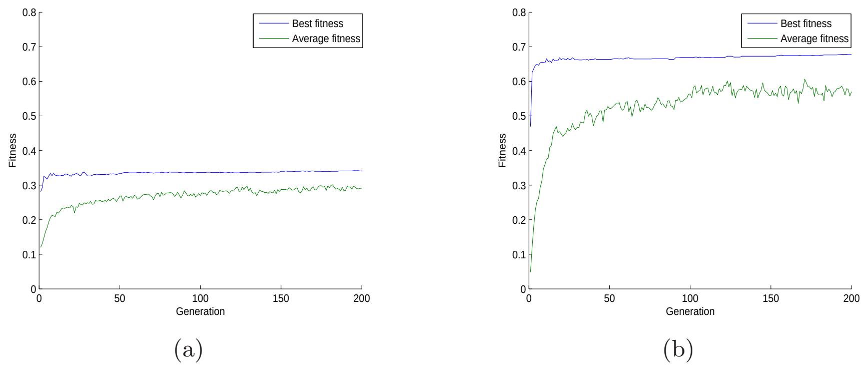

Fig. 6. Fitness Evolution Examples: (a) low curiosity (b) high curiosity

4.2 Scenario 2: Environment Recognition

We tested behavior based environment recognition using five different rooms. For comparison we selected two sets of fuzzy motivation criteria (C, H, O, E), the first criterion used was (.8, .1, 0, .1) and the second was (.5, .3, 0, .2). We stopped each experiment after 400 steps.

During testing, the same robot (neural net) was made to navigate and recognize five rooms with the following shapes: H, T, L, Square and Rectangle. The procedure was repeated 10 times for each room. The threshold for r-node recognition was set at 10 nearest neighbor nodes in the SOM output layer. Environment recognition is the % recognition of which room the robot actually visited. Tables 1 and 2 show our results.

5 Discussion

The results of our experiments are summarized below:

(1) **Navigation:** By changing the different motivation factors the characteristics of robot navigation were changed. With lower curiosity and higher homing the robot conserved its battery (battery usage was around 50%) and the robot

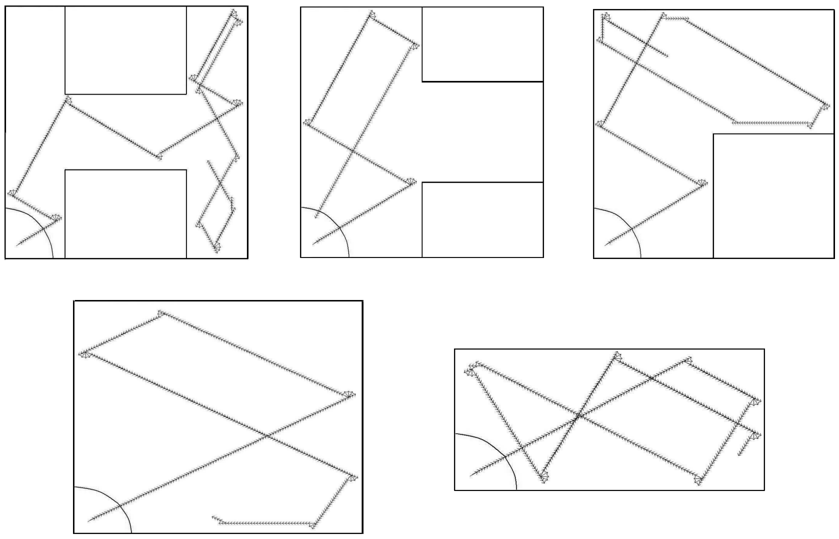

Fig. 7. Navigation with (C, H, O, E) as (.8, .1, 0, .1)

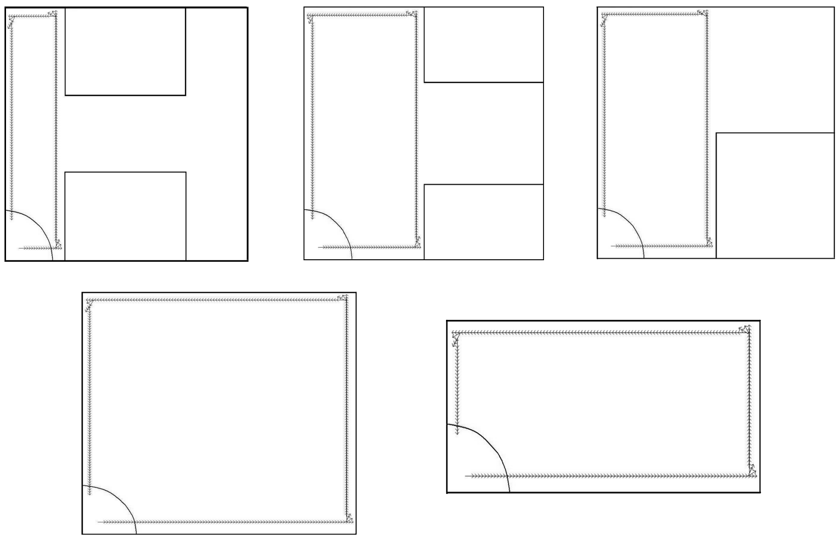

Fig. 8. Navigation with (C, H, O, E) as (.5, .3, 0, .2)

Table 1. Navigation and Environment Recognition with (C, H, O, E) as (.8, .1, 0, .1)

Room shape	Average Fitness	% of Optimal Exploration	% Recognition	% Battery usage
H	0.6421	93.84	80.00	79.19
T	0.6686	91.77	100.00	78.45
L	0.6507	97.04	100.00	79.58
Square	0.6699	99.88	100.00	79.56
Rectangle	0.6377	95.29	100.00	79.88

Table 2. Navigation and Environment Recognition with (C, H, O, E) as (.5, .3, 0, .2)

Room shape	Average fitness	% of Optimal Exploration	% Recognition	% Battery usage
H	0.2465	54.66	20.00	57.90
T	0.2694	57.84	0.00	59.66
L	0.2707	57.93	0.00	59.67
Square	0.3274	80.63	100.00	74.64
Rectangle	0.267	57.70	0.00	59.60

managed to explore only around 50% of the region. With higher curiosity the robot explored more and used a higher amount of battery but it did not return home very often.

(2) **Environmental Recognition:** Different motivation factors affected the capacity of the robot to recognize its environment. With lower curiosity the robot managed to recognize its environment much less than with higher curiosity values. With higher curiosity, the environment vector generated was more representative of the room shape which would explain this result. The orientation efficiency value (l) could be determined by the % recognition of which room the robot actually ran in but was not included in the fitness calculation as the orientation value (O) was set to zero.

6 Conclusions

In general this method was shown as an effective mechanism for generating a variety of robotic behavior within a framework that is simple and intuitive to understand.

Future work includes: the integration of other motivations such as urgency for completing specific missions, integration of % recognition into the fuzzy fitness calculation, the construction of a low cost robot specifically designed to test this method, and the possible real-time extensions to this method.

Acknowledgements

This research was partially funded by Fundación Andes and by the research center (DGIP) of UTFSM. We would also like to thank Dr. Seiji Yamada for providing information about AEM.

References

1. Park, H., Kim, E., Kim, H.: Robot Competition Using Gesture Based Interface. In: Moonis, A. Esposito, F. (eds): Innovations in Applied Artificial Intelligence. Lecture Notes in Artificial Intelligence, Vol. 3353. Springer-Verlag, Berlin (2005) 131-133.
2. Tasaki, T., Matsumoto, S., Ohba, H., Toda, M., Komatani, K., Ogata, T., Okuno, H.: Distance-Based Dynamic Interaction of Humanoid Robot with Multiple People. In: Moonis, A. Esposito, F. (eds): Innovations in Applied Artificial Intelligence. Lecture Notes in Artificial Intelligence, Vol. 3353. Springer-Verlag, Berlin (2005) 111-120.
3. Brooks, R.: A Robust Layered Control System for a Mobile Robot. IEEE Journal of Robotics and Automation, Vol. RA-2, No. 1 (1986) 14-23.
4. Arkin, R.: Behavior-Based Robotics, MIT Press, Cambridge, (1998).
5. YAKS simulator website: http://r2d2.ida.his.se/
6. Yamada, S.: Evolutionary behavior learning for action-based environment modeling by a mobile robot. Applied Soft Computing, 5 (2005) 245-257
7. Jang, J., Chuen-Tsai, S., Mitzutani, E.: Neuro-Fuzzy and Soft Computing, Prentice Hall, NJ, (1997).
8. Konolige, K., Meyers, K., Saffiotti, A.: FLAKEY, an Autonomous Mobile Robot. SRI technical document, July 20 (1992)
9. Goodrige, S., Kay, M., Luo, R.: Multi-Layered Fuzzy Behavior Fusion for Reactive Control of an Autonomous Mobile Robot. Proceedings of the Sixth IEEE International Conference on Fuzzy Systems (July 1997) 573-578
10. Hoffman, F.: Soft computing techniques for the design of mobile robot behaviors. Information Sciences, 122 (2000) 241-258
11. Al-Khatib, M., Saade, J.: An efficient data-driven fuzzy approach to the motion planning problem of a mobile robot. Fuzzy Sets and Systems, 134 (2003) 65-82
12. Izumi, K., Watanabe, K.: Fuzzy behavior-based control trained by module learning to acquire the adaptive behaviors of mobile robots. Mathematics and Computers in Simulation, 51 (2000) 233-243
13. Martnez Barber, H., Gmez Skarmeta, A.: A Framework for Defining and Learning Fuzzy Behaviours for Autonomous Mobile Robots, International Journal of Intelligent Systems, 17(1) , (2002) 1-20
14. Zhou, C. : Robot learning with GA-based fuzzy reinforcement learning agents, Information Sciences, 145 (2002) 45-68
15. Seraji, H., Howard, A.: Behavior-Based Robot Navigation on Challenging Terrain: A Fuzzy Logic Approach, IEEE Trans on Robotics and Automation, V.18, No.3 (June 2002) 308-321
16. Huitt, W.: Motivation to learn: An overview. Educational Psychology Interactive. Valdosta State University.: http://chiron.valdosta.edu/whuitt/col/motivation/-motivate.html, (2001).
17. Jang, J.-S, Sun, C.-T., Sun, Mizutani, E.: Neuro-Fuzzy and Soft Computing: a computational approach to learning and machine intelligence, Prentice Hall, NJ, (1997).
18. Kohonen Teuvo: The self-organizing map. Proceedings of the IEEE, 79(9): 1464-1480, (1990)