Fuzzy Robot Controller Tuning with Biogeography-Based Optimization

George Thomas*, Paul Lozovyy, and Dan Simon

Cleveland State University, Department of Electrical and Computer Engineering
2121 Euclid Avenue, Stillwell Hall Room 332
Cleveland, Ohio 44115
{g.l.thomas71,p.lozovyy89,d.j.simon}@csuohio.edu

Abstract. Biogeography-based optimization (BBO) is an evolutionary algorithm (EA) based upon the models of biogeography, which describe the relationship between habitat suitability and the migration of species across habitats. In this work, we apply BBO to the problem of tuning the fuzzy tracking controller of mobile robots. This is an extension of previous work, in which we used BBO to tune a proportional-derivative (PD) controller for these robots. We show that BBO can successfully tune the shape of membership functions for a fuzzy controller with both simulation and real world experimental results.

Keywords: biogeography-based optimization, fuzzy logic control.

1 Introduction

Optimization is one of the main objectives for almost any engineering project. Iterating through the entire search space of a problem is usually not feasible because of how much CPU time this can require. Engineers have therefore adapted natural processes such as evolution to optimization with great results [1]. Biogeography-based optimization (BBO) [2] belongs to the class of heuristic optimization methods known as evolutionary algorithms (EAs). BBO is based upon the models of biogeography, which describe the migration of species across island habitats [3].

In previous work, we have demonstrated BBO's efficacy at PD controller tuning [4]. The next step, which we take in this paper, is to test BBO's performance at optimizing a more complex controller. We have chosen fuzzy logic control because of its ease of use and increasing ubiquity, combined with the difficulty in tuning it optimally [5].

In the remainder of this section we give an overview of fuzzy logic and BBO. Section 2 describes our robot hardware. Section 3 presents our fuzzy robot control tuning approach, simulation results, and experimental results. Section 3 provides the primary contribution of this research, and reports the first successful use of BBO for fuzzy logic system tuning. Section 4 concludes with some discussion and suggestions for future work.

* Corresponding author.

1.1 History of Fuzzy Set Theory to Fuzzy Logic Control

Fuzzy sets were first introduced in 1965 by Lotfi A. Zadeh [6]. Fuzzy set theory is a generalization of classic (or crisp) set theory, in which elements of a set have a continuous value that represents their membership in a class, generally with a range of [0, 1]. In classic set theory, membership is discrete—an element can only be a member or a nonmember of a set. Fuzzy logic is an extension of fuzzy set theory to the field of logic. Classical logic uses Boolean operators such as AND and OR in reasoning, but fuzzy logic makes these operators ambiguous. The fuzzy generalizations of the operations AND and OR are the t-norm and s-norm respectively. Several functions fit the necessary and sufficient conditions for being a t-norm or an s-norm; the most common t-norms are the min and product functions, and the most common s-norms are max and sum [7].

Zadeh later proposed fuzzy logic as a method of decision making [8]. In this scheme, the domain of each input and output variable is mapped by fuzzy adjectives that describe the value of the variable, such as “hot” or “high.” Each adjective is also associated with a membership function (MF). The adjectives make it possible to form control rules by relating variables together in plain sentences. For instance, “if boiler pressure is *high*, then escape valve should be *slightly open*.” These control rules can then be written in terms of fuzzy set theory with t-norms and s-norms relating variables together.

In order to go from decision making to control, fuzzy outputs must be made crisp, so that they can be used in practice. This process is called defuzzification. The first fuzzy controller was implemented by Mamdani and Assilian for steam engine control in 1975. In this defuzzification scheme, max-min product composition was used to produce crisp outputs from the inputs [9].

The universal approximation theorem states that an arbitrary function composed of a set of sub-functions can approximate any nonlinear function with an arbitrary level of accuracy [10]. A fuzzy logic controller is composed of its rule base and membership functions. These parameters determine the accuracy of the output with respect to the function we wish to approximate. Because our robot is a nonlinear system, a fuzzy logic control is more suitable for this problem than PD control, which is linear.

1.2 BBO

The mathematical models that describe biogeography were first published in the 1960s [11]. These models describe the immigration and emigration of species between habitats, based on the fitness of the habitats. Because BBO is based on biogeography, a lot of its terms are borrowed from biogeography [2]. A solution, or individual, is referred to as a habitat or island. The fitness, or cost, of a habitat is often referred to as habitat suitability index (HSI). Finally the independent variables, or features, of a solution to a problem are referred to as suitability index variables (SIVs). Sometimes the population of candidate solutions is referred to as an archipelago. In BBO, different operators are applied to the population of solutions to accomplish the optimization—migration is one such operator. The probabilities of an SIV immigrating to a habitat, or emigrating from a habitat, are λ and μ, which are determined by the HSI. These probabilities are typically complements of each other (i.e., $\lambda = 1 - \mu$).

In addition to the migration operator, there is also the mutation operator, as with most other EAs. Mutation randomly selects a solution from the population with a specified probability and sets one of its independent variables to a random value. Mutation helps to introduce new features into the population, since in BBO, the population tends to become populated with the same high fitness solution as the number of generations increases. A pseudocode outline of BBO is provided in Algorithm 1.

```
For each candidate solution $H_i$
    For each solution feature $s$
        Select candidate solution $H_i$ for immigration with probability $\lambda_i$
        If candidate solution $H_i$ has been selected, then
            Select $H_j$ for emigration with probability $\mu_j$
            If $H_j$ has been selected, then
                $H_i(s) \leftarrow H_j(s)$
            end
        end if
    next solution feature
    Probabilistically mutate candidate solution $H_i$
next candidate solution
```

Algorithm 1. A pseudocode representation of one BBO generation

BBO has been applied to several real-world problems. In addition to experimental robot control tuning, as discussed in this paper and in [4], BBO has been applied to aircraft engine sensor selection [2], power system optimization [12, 13], groundwater detection [14], mechanical gear train design [15], satellite image classification [16], and neuro-fuzzy system training for biomedical applications [17]. Recent research in the area of BBO has focused on putting it on a firm theoretical foundation, including the derivation of Markov models [18, 19] and dynamic system models [20] that describe its behavior.

2 Hardware

In this research, we use a standard two-wheeled robot design [21]. Our robots were originally assembled in 2007 [22], but we have adapted them to our work [4]. Our robots are designed to be flexible. We can replace any component at any time with a minimum of effort. Solid state components such as a microcontroller, voltage regulators, and H-bridges are mounted and hand soldered on two-layer printed circuit boards (PCBs). The PCBs, sensors, and other electronics are mounted on thin plastic boards separated by aluminum standoffs. Two geared DC motors are attached with brass brackets. Two AA battery packs, which each hold eight rechargeable batteries, supply power to the motors and PCBs separately. Figure 1 depicts one of these robots.

Each robot is equipped with a MaxStream 9Xtend radio, which is used to communicate with the base station. The base station is a PC running a Matlab® graphical user interface (GUI), and it can transmit commands to each robot, update robot control parameters, and receive robot tracking data. The RF device can output power from 1 mW to 1 W and has a maximum outside range of 40 km. It uses a serial interface that can operate at several baud rates. The radio, costing about $200, is the most expensive device on each robot.

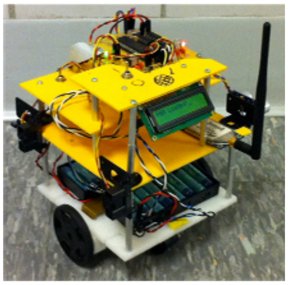
Fig. 1. One of our robots

One of the goals of this work is to improve upon our previous research [4]. We have taken steps towards improving the reliability of our robots by using infrared (IR) range-finding sensors to find distances. The reason for this change from ultrasonic range-finders, which we used in our previous work, is to reduce the effect of noise that came from the motors that made the robot's controller unstable. The sampling rate of the IR sensors is slower than the ultrasonic sensors, so our control code needed modification as well.

Our second upgrade in this research is a new printed circuit board. We have added opto-isolation to the robot PCBs to completely isolate the motor circuit from the logic circuit. Another benefit of opto-isolation is the elimination of the effect of back EMF that is produced by DC motors. Also, the boards have been redesigned for use of the robot's new IR range-finders, and the redesign eliminates capacitance that could be induced between small traces on the PCB. More flexibility also is added to the new boards such as access to every pin of the microcontroller if more components are required.

3 Description of Experiments

In this work, we use tracking error and its derivative for fuzzy logic controller inputs, and a motor voltage correction value for the output. Each of the inputs and outputs are mapped with five sum-normal triangular MFs. Any *n* sum-normal MFs that describe some variable can be described by a set of *n* break points. In the case of our triangular MFs, a break point can also be described as the only point in a MF where it is both non-differentiable and equal to 1. Therefore, the fuzzy MFs for each variable are completely specified by five parameters. Figure 2 illustrates this concept.

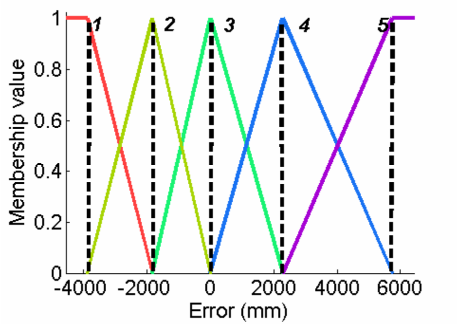

Fig. 2. Five fuzzy membership functions which are triangular and sum-normal. The numbered dashed lines represent break points.

BBO modifies the shape of the MFs by modifying these break points. In addition, we use the Mamdani inference system [7]. Min and max are used for the t-norm and s-norm in our work. We define a rule for each of the intersections between a pair of input MFs. Table 1 represents our rule table, where LP means a large positive, SP means small positive, Z means a value near zero, SN means small negative, and LN means large negative. Table 2 shows the search domain that was used for this work. Note that the zero MF is constrained to have a break point equal to 0. This means that the fuzzy logic system consists of 12 independent variables: four break points for error, four break points for delta-error, and four break points for delta-voltage.

Table 1. Fuzzy rule table, where output corresponds to a change in motor voltage

		Error				
		LN	SN	Z	SP	LP
ΔError	LN	LN	LN	LN	SN	Z
	SN	LN	LN	SN	Z	SP
	Z	LN	SN	Z	SP	LP
	SP	SN	Z	SP	LP	LP
	LP	Z	SP	LP	LP	LP

Table 2. The search space used for each fuzzy membership function break point in the fuzzy controller

		Break points				
		LN	SN	Z	SP	LP
Variable	Error (mm)	[-10000, -2500]	[-2500, 0]	[0, 0]	[0, 2500]	[2500, 10000]
	ΔError (mm/s)	[-750, -100]	[-100, 0]	[0, 0]	[0, 100]	[100, 750]
	ΔMotor Voltage (normalized)	[-1000, -250]	[-250, 0]	[0, 0]	[0, 250]	[250, 1000]

The maximum allowable magnitudes of the large negative and large positive values were chosen to be more than an order of magnitude greater than the values that we typically used in our controller, which gave BBO additional flexibility in its optimization search. The cost function that we used is a weighted sum of the rise time of the robot trajectory to a reference point, and the integral of absolute value of the tracking error:

$$Cost = k_1 \int |e(t)| dt + k_2 r \tag{1}$$

where k_1 and k_2 are weighting constants, $e(t)$ is the tracking error (mm), and r is the rise time, which represents the length of time that the robot takes to reach 95% of the reference tracking distance. For our experiments, k_1 and k_2 were set to 1 and 5 respectively to give approximately equal contribution of each term to the cost function. The time duration of each robot tracking experiment was 20 seconds.

3.1 Simulation Results

We ran some preliminary simulations before we performed experimentation with the robots. We ran 100 Monte Carlo simulations of BBO with a population size of 20, for 50 generations. Also, the mutation probability was set to 10% and a single elite solution was preserved between generations. We used an unusually high mutation rate because our population size was relatively small. Figure 3 shows the cost of Equation (1) decreasing as a function of generation number in BBO.

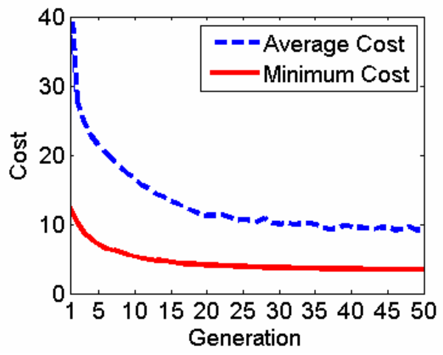

Fig. 3. BBO optimizing the fuzzy controller cost. The traces show the average and minimum cost of the entire population, averaged over 100 Monte Carlo simulations.

3.2 Experimental Results

In our experiment on the real robots, we ran 10 generations of BBO with a population size of 6 and a mutation probability of 20%. One elite solution was kept between generations. Figure 4 shows the cost as BBO optimizes the population each generation. Figure 5 shows the robot paths representing the best solutions of the first and final generations.

The integral of absolute error changed from 1476 mm·s for the first generation to 717 mm·s in the final generation, which is a decrease of 51%. Figure 6 shows the MFs of the best solution produced by BBO.

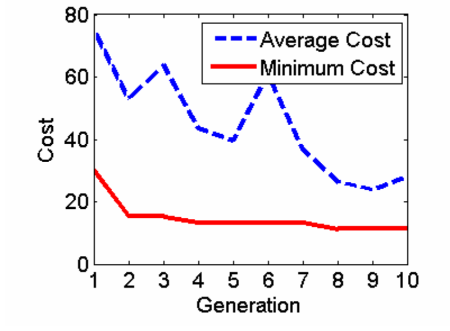

Fig. 4. BBO optimizing the fuzzy controller cost on the real robot

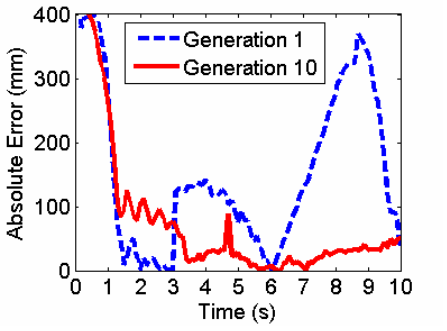

Fig. 5. Tracking error of best solution at the first and last generation

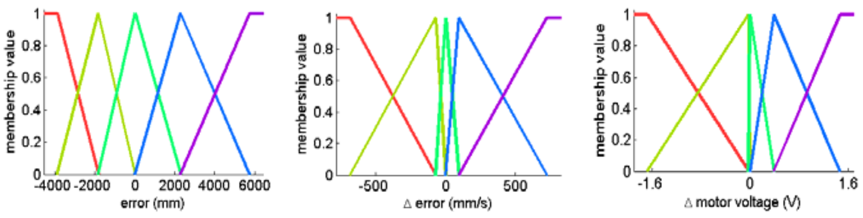

Fig. 6. Fuzzy MFs produced by BBO after 10 generations

4 Conclusion

The difficulty associated with tuning a fuzzy controller makes the application of EAs to this problem very appropriate. We have shown that BBO can perform this task well both in simulations and in a real robotic system. Future work may include performing these experiments with different parameter settings. Changing parameter settings such as population size and mutation probability may not provide much information for qualitative tests like we have performed here, which were designed to test whether or not BBO can optimize the MFs of a fuzzy logic system. However, such tests would be very useful for obtaining more quantitative results. For example, if we wanted to examine how fast BBO could produce a set of membership functions with a cost value below some threshold, it would be important to find optimal BBO parameter settings. Other future work may include using BBO to tune non-triangular MFs, such as MFs with trapezoidal or Gaussian shapes. BBO could also be used to tune the rule base of a fuzzy controller. Also, we can apply BBO to the optimization of other control algorithms, such as artificial neural networks.

It would be very interesting to compare BBO's performance at tuning our fuzzy controller versus BBO's performance at tuning our PD controller [4], but it is difficult to compare these two tasks with the data from our experiments. Although we use the same cost function to evaluate the performance of both controllers, their input parameters are very different. The PD controller's inputs are proportional gain and derivative gain, and the fuzzy controller's inputs are the shapes of its constituent membership functions. These quantities are not comparable in any meaningful way. Therefore, the domains of input values for each problem are not comparable. Future work in this regard would be to redesign our experiments in such a way that we can make meaningful comparisons between BBO's performance at these two problems. Perhaps we can then draw conclusions about tradeoffs between controller robustness and tuning complexity.

Acknowledgment. This material is based upon work supported by the CSU Provost's Office, and by the National Science Foundation under Grant No. 0826124.

References

1. Fogel, D.B.: Evolutionary Computation. John Wiley and Sons, Hoboken (2006)
2. Simon, D.: Biogeography-Based Optimization. IEEE Transactions on Evolutionary Computation 12(6), 702–713 (2008)
3. Lomolino, M.V., Riddle, B.R., Brown, J.H.: Biogeography. Sinauer Associates, Sunderland (2009)
4. Lozovyy, P., Thomas, G., Simon, D.: Biogeography-based optimization for robot controller tuning. In: Igelnik, B. (ed.) Computational Modeling and Simulation of Intellect: Current State and Future Perspectives. IGI Global (in print, 2011)
5. Jantzen, J.: Tuning of fuzzy PID controllers. Denmark Tech. Report no 98- H 871(fpid) 30, 1–22 (1998)
6. Zadeh, L.A.: Fuzzy Sets. Information and Control 8(3), 338–353 (1965)
7. Jang, J.-S.R., Sun, C.-T.: Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence. Prentice Hall, Upper Saddle River (1997)
8. Zadeh, L.A.: The Concept of a Linguistic Variable and its Application to Approximate Reasoning. Inf. Sci. 8, 199–249 (1975)
9. Mamdani, E.H., Assilian, S.: An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller. Int. J. Man-Machine Studies 7(1), 1–13 (1975)
10. Tikk, D., Kóczy, L.T., Gedeon, T.G.: A Survey on Universal Approximation and its Limits in Soft Computing Techniques. Int. J. Approx. Reasoning 33(2), 185–202 (2003)
11. MacArthur, R., Wilson, E.: The Theory of Biogeography. Princeton Univ. Press, Princeton (1967)
12. Rarick, R., Simon, D., Villaseca, F., Vyakaranam, B.: Biogeography-based optimization and the solution of the power flow problem. In: IEEE Conference on Systems, Man, and Cybernetics, pp. 1029–1034 (2009)
13. Roy, P., Ghoshal, S., Thakur, S.: Biogeography-based optimization for economic load dispatch problems. Electric Power Components and Systems (38), 166–181 (2010)
14. Kundra, H., Kaur, A., Panchal, V.: An integrated approach to biogeography based optimization with case based reasoning for retrieving groundwater possibility. In: 8th Annual Asian Conf. and Exhibition on Geospatial Information, Tech. and Applications (2009)
15. Savsani, V., Rao, R., Vakharia, D.: Discrete optimisation of a gear train using biogeography based optimisation technique. Int. J. Design Eng. (2), 205–223 (2009)
16. Panchal, V., Singh, P., Kaur, N., Kundra, H.: Biogeography based satellite image classification. Int. J. of Comp. Sci. and Info. Security (6), 269–274 (2009)

17. Ovreiu, M., Simon, D.: Biogeography-based optimization of neuro-fuzzy system parameters for diagnosis of cardiac disease. In: Genetic and Evolutionary Computation Conference, pp. 1235–1242 (2010)
18. Simon, D., Ergezer, M., Du, D., Rarick, R.: Markov models for biogeography-based optimization. IEEE Transactions on Systems, Man, and Cybernetics (Part B: Cybernetics), 299–306 (2011)
19. Simon, D., Ergezer, M., Du, D.: Population distributions in biogeography-based optimization algorithms with elitism. IEEE Conference on Systems, Man, and Cybernetics, 1017–1022 (2009)
20. Simon, D.: A Dynamic System Model of Biogeography-Based Optimization (2010) (submitted for publication)
21. Jiang, X., Motai, Y., Zhu, X.: Predictive Fuzzy Logic Controller for Trajectory Tracking of a Mobile Robot. In: Proc. IEEE Workshop on Soft Comp. in Ind. Appl., pp. 29–32 (2005)
22. Churavy, C., Baker, M., Mehta, S., Pradhan, I., Scheidegger, N., Shanfelt, S., Rarick, R., Simon, D.: Effective Implementation of a Mapping Swarm of Robots. In: IEEE Potentials, pp. 28–33 (2008)