TRAJECTORY TRACKING OF A MOBILE ROBOT USING FUZZY LOGIC TUNED BY GENETIC ALGORITHM

H. N. Pishkenari, S. H. Mahboobi and A. Alasty*

Center of Excellence in Design, Robotics and Automation (CEDRA)
Department of Mechanical Engineering, Sharif University of Technology
Tehran, Iran, Tel: +98216165504, Fax: +98216000021
aalasti@sharif.edu

* Corresponding Author

(Received: February 6, 2004 – Accepted in Revised Form: August 9, 2006)

Abstract In recent years, soft computing methods, like fuzzy logic and neural networks have been presented and developed for the purpose of mobile robot trajectory tracking. In this paper we will present a fuzzy approach to the problem of mobile robot path tracking for the CEDRA rescue robot with a complicated kinematical model. After designing the fuzzy tracking controller, the membership functions and rule weights will be optimized by genetic algorithm in order to obtain more acceptable results. Simulation results have demonstrated significant improvements in controller efficacy.

Key Words Fuzzy Control, Trajectory Tracking, Rescue Robot, Genetic Algorithm

1. INTRODUCTION

The problem of trajectory tracking for mobile robots has been an attractive issue in the robotic field during recent years. Our purpose is to control a certain high mobility rover for rescue operations that have a complex kinematical model.

Path tracking has been widely investigated, such as, Dongbing Gu and Huosheng Hu [1] who developed a path tracking scheme for a car-like mobile robot based on neural predictive control; Jacky Baltes and Robin Otte [2] developed a Fuzzy Logic Path Controller for path tracking, Jiri Sika and Joop Pauwelussen [3] used Look-ahead Virtual Point and developed a lateral controller to follow a pre-described path.

Waneck [4] proposed a fuzzy controller for an autonomous boat in absence of a nonlinear dynamic model of the vehicle. Sugeno et al. [5] has designed a fuzzy controller based on the fuzzy modeling of a human operator’s control actions to navigate and park a car. Larkin [6] has proposed a fuzzy controller for aircraft flight control where the fuzzy rules are generated by interrogating an experienced pilot and asking him a number of highly structured questions. The design of a fuzzy logic system (FLS) includes the design of a rule base, input scale factors, output scale factors, and finally the design of the membership functions.

Input scale factors transform the real inputs into normalized values, and output scale factors transform the normalized outputs into real values. Some studies have shown that FLS performance is more dependent on membership function design than rule base design [7]. The tuning of input and output scale factors is known as context adaptation.
Some researchers have studied genetic algorithms for context adaptation [8]. Others have used genetic algorithms to design the rule base and the scale factors when the normalized membership functions are fixed [9]. Some studies used neural networks for context adaptation [10]. A genetic learning process for the membership function design, coupled with a heuristic method for the rule base design, has been proposed in [11].

The subject of this paper is restricted to the tuning of membership functions. Researchers have used many different methods over the past decade to optimize fuzzy membership functions. These methods include genetic algorithms [12], neural networks [13], evolutionary programming [14], geometric methods [15], fuzzy equivalence relations [16], heuristic methods [17], and gradient descent [18].

Design and selection of linguistic variables and rules of a fuzzy controller require expert knowledge of the under control system. This knowledge is usually obtained by trial-and-error or by consulting and observing a human operator controlling a real system.

One of the main factors in the design of efficient and robust fuzzy logic controllers is the selection of parameters of the membership functions. We intended to have a weight to every rule and optimize these weights for finding suitable rule base. The existing approaches for choosing the membership functions are based on trial-and-error process, which mostly lack learning and autonomy. One method of removing the uncertainty associated with the selection of these variables is the use of genetic algorithms (GA). In this trajectory tracking application, the fitness function evaluates the robot’s path, taking into account the distance and orientation error from the desired path. The optimum membership functions and weights of the rules differ from one path to another. Therefore a very complicated desired path has been chosen in the optimization process, to assure the robustness of the optimal fuzzy controller.

2. KINEMATIC MODEL OF THE ROBOT

A top view of CEDRA rescue robot (see Figure 1.) is shown in Figure 2. This mobile robot consists of six wheels. Front and rear ones are steerable and the remaining four wheels are mounted beside the robot. Robot kinematic Equations derivation can be seen in the Equation 1 through 8 and Equations 9, 10 and 11 depict the final relations.

\[
\begin{align*}
\dot{\bar{r}}_c &= \frac{(\dot{r}_f + \dot{r}_r)}{2} \\
\dot{\bar{V}}_c &= \frac{(\dot{V}_f + \dot{V}_r)}{2} \\
\dot{V}_f &= V[\cos(\phi_c + \theta)\bar{i} + \sin(\phi_c + \theta)\bar{j}] 
\end{align*}
\]
Interaction between robot and path curve is shown in Figure 3. Xc and Yc give the coordinates of the robot geometrical center. The orientation of the robot is given by \( \Phi_c \). Coordinates \( X_n \) and \( Y_n \) are the coordinates of the closest point to the robot on the path. The slope of the path at this point is given by \( \Phi_n \). In addition, subscripts \( r \), \( c \) and \( f \) refer to the rear, center and front of the robot respectively.

In this application, the control signals are robot velocity and steering angle, which are determined based on the deviation between the desired and actual position and orientation. It should be mentioned that environment is completely known and path is prescribed and then the optimized fuzzy controller will be proposed for tracking this path. The control function is defined as a function of positional error (d), the orientation error (\( \Delta \Phi \)), and the curvature of the path (R). The positional error is the distance between points \((X_c, Y_c)\) and \((X_n, Y_n)\). The orientation error is defined as \( \Delta \Phi = \Phi_n - \Phi_c \). It is assumed that the path is continuous. However, the curvature and any derivative of the path may be discontinuous.

**3. DESIGN OF THE FUZZY LOGIC CONTROLLER**

The crux of designing an FLC lies in the selection of high-performance membership functions that represent the human expert’s interpretation of linguistic variables, because different membership functions determine the extent to which the rules affect the action and hence the performance [19]. The existing iterative approaches for choosing the membership functions are basically a manual trial-and-error process and lack learning capability or autonomy. Therefore, the more efficient and systematic genetic algorithm which acts on the survival-of-the-fittest has applied to FLC design for searching the poorly understood, irregular and complex membership function space with improved performance.

This section describes the design of Fuzzy Logic Controller for CEDRA rescue robot. It also discusses the heuristic that we apply to determine the number of necessary fuzzy input and output sets. All of the membership functions used in input and output sets are in the form of trapezoid function formulated as in Equation 12 [20].
3.1. Fuzzy Input Sets  Fuzzy Logic controller, the input to the controller (curvature, position error, and orientation error) is converted into a series of fuzzy sets via the singleton fuzzifier. The number and exact shape of these fuzzy sets critically determine the performance of the controller. These fuzzy sets describe a qualitative situation in which the output of the controller is qualitatively different.

In other words, whenever the desired behavior (e.g., change from going straight to turning left or change from fast to medium speed) of the controller changes in an input situation, a fuzzy set is created to represent this case.

Curvature consists of three fuzzy sets; Left Curvature, Straight and Right Curvature. The fuzzification of the positional error includes five sets; NegHighDist (NHD), NegLowDist (NLD), ZeroDist (ZD), PosLowDist (PLD) and PosHighDist (PHD). Clearly it is desirable that the robot be on the line (ZeroDist). Assume that the robot is far away from the path, then the desired behavior is to turn either right/left towards the path, drive straight towards the path, and turn left/right to straighten out. Therefore, we require two extra sets on each side of the path.

Similar reasoning leads to the design of the fuzzy sets for the orientation error. A total of five sets have been used to describe different cases; NegHighAngle (NHA), NegLowAngle (NLA), ZeroAngle (ZA), PosLowAngle (PLA) and PosHighAngle (PHA).

3.2. Fuzzy Output Sets  There are two outputs from fuzzy controller to the robot: (a) speed and (b) Steering Angle. There are four membership functions for describing the speed heuristic variable: Zero, Slow, Medium and Fast. The steering Angle is determined by using five membership functions; SharpLeft (SL), LowLeft (LL), Stright (ST), LowRight (LR) and SharpRight (SR). A crisp output value is then computed from this fuzzy set. In this research, we used the well-known centroid defuzzification method, which uses the center of gravity as the crisp output value.

3.3. Fuzzy Rule Base  Given these fuzzy input sets, a fuzzy controller uses a set of fuzzy rules to specify the desired control behavior through the minimum inference engine. After the design of the fuzzy input and output sets, the design of the fuzzy rules is straightforward. There are a total of 5*5*3=75 possible different input configurations. For each of these input configurations, a rule was specified to indicate the desired speed and directional settings. The controller rule base appears in Table 1.

4. GENETIC ALGORITHM  Algorithms for function optimization are generally limited to convex regular functions. However, many functions are multi-model, discontinuous, and non-differentiable. Genetic algorithms (GAs) are a class of stochastic search techniques, loosely based on ideas from biological evolution, which have been used successfully for a great variety of different problems (e.g., [21-23]).

The GA searches for an optimal solution from a population of candidate solutions according to an objective function, which is used to establish the fitness of each candidate as a solution. The governing process in the search is the application of appropriate breeding operators to candidate solutions in a given generation to form the candidates for the next generation. These operators are designed to preserve the most successful aspects of candidate fitness until the best possible solution is attained.

At each generation, a new set of approximations is created by the process of selecting individuals according to their level of fitness in the problem domain and breeding them together using operators borrowed from natural genetics. This process leads to the evolution of populations of individuals that are better suited to their environment than the individuals that they were created from, just as in natural adaptation.
<table>
<thead>
<tr>
<th>No.</th>
<th>Input</th>
<th>Output</th>
<th>No.</th>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Curvature</td>
<td>d</td>
<td>Δ φ</td>
<td>V</td>
<td>Θ</td>
</tr>
<tr>
<td>1</td>
<td>StraightLine</td>
<td>ZD</td>
<td>ZA</td>
<td>Fast</td>
<td>ST</td>
</tr>
<tr>
<td>2</td>
<td>StraightLine</td>
<td>ZD</td>
<td>PLA</td>
<td>Med</td>
<td>LL</td>
</tr>
<tr>
<td>3</td>
<td>StraightLine</td>
<td>ZD</td>
<td>PHA</td>
<td>Slow</td>
<td>SL</td>
</tr>
<tr>
<td>4</td>
<td>StraightLine</td>
<td>ZD</td>
<td>NLA</td>
<td>Slow</td>
<td>LR</td>
</tr>
<tr>
<td>5</td>
<td>StraightLine</td>
<td>ZD</td>
<td>NHA</td>
<td>Slow</td>
<td>SR</td>
</tr>
<tr>
<td>6</td>
<td>StraightLine</td>
<td>PHD</td>
<td>NHA</td>
<td>Slow</td>
<td>ST</td>
</tr>
<tr>
<td>7</td>
<td>StraightLine</td>
<td>PHD</td>
<td>NLA</td>
<td>Slow</td>
<td>LL</td>
</tr>
<tr>
<td>8</td>
<td>StraightLine</td>
<td>PHD</td>
<td>PHA</td>
<td>Slow</td>
<td>SR</td>
</tr>
<tr>
<td>9</td>
<td>StraightLine</td>
<td>PHD</td>
<td>PLA</td>
<td>Slow</td>
<td>SL</td>
</tr>
<tr>
<td>10</td>
<td>StraightLine</td>
<td>PHD</td>
<td>PLA</td>
<td>Slow</td>
<td>SL</td>
</tr>
<tr>
<td>11</td>
<td>StraightLine</td>
<td>NLD</td>
<td>NLA</td>
<td>Slow</td>
<td>LR</td>
</tr>
<tr>
<td>12</td>
<td>StraightLine</td>
<td>NLD</td>
<td>PHA</td>
<td>Slow</td>
<td>SL</td>
</tr>
<tr>
<td>13</td>
<td>StraightLine</td>
<td>NLD</td>
<td>NHA</td>
<td>Slow</td>
<td>SR</td>
</tr>
<tr>
<td>14</td>
<td>StraightLine</td>
<td>NLD</td>
<td>NLA</td>
<td>Slow</td>
<td>SR</td>
</tr>
<tr>
<td>15</td>
<td>StraightLine</td>
<td>NLD</td>
<td>ZA</td>
<td>Slow</td>
<td>LR</td>
</tr>
<tr>
<td>16</td>
<td>StraightLine</td>
<td>NLD</td>
<td>NLA</td>
<td>Slow</td>
<td>LR</td>
</tr>
<tr>
<td>17</td>
<td>StraightLine</td>
<td>NLD</td>
<td>PHA</td>
<td>Slow</td>
<td>SR</td>
</tr>
<tr>
<td>18</td>
<td>StraightLine</td>
<td>NLA</td>
<td>Med</td>
<td>SR</td>
<td>56</td>
</tr>
<tr>
<td>19</td>
<td>StraightLine</td>
<td>NLA</td>
<td>SLA</td>
<td>Slow</td>
<td>SL</td>
</tr>
<tr>
<td>20</td>
<td>StraightLine</td>
<td>NLD</td>
<td>NHA</td>
<td>Slow</td>
<td>SR</td>
</tr>
<tr>
<td>21</td>
<td>StraightLine</td>
<td>NLD</td>
<td>PLA</td>
<td>Slow</td>
<td>SL</td>
</tr>
<tr>
<td>22</td>
<td>StraightLine</td>
<td>NLD</td>
<td>ZA</td>
<td>Slow</td>
<td>LR</td>
</tr>
<tr>
<td>23</td>
<td>StraightLine</td>
<td>NLD</td>
<td>PLA</td>
<td>Slow</td>
<td>LL</td>
</tr>
<tr>
<td>24</td>
<td>StraightLine</td>
<td>NLD</td>
<td>NLA</td>
<td>Slow</td>
<td>SL</td>
</tr>
<tr>
<td>25</td>
<td>StraightLine</td>
<td>NLD</td>
<td>NHA</td>
<td>Slow</td>
<td>SR</td>
</tr>
<tr>
<td>26</td>
<td>LeftCircle</td>
<td>NHD</td>
<td>NHA</td>
<td>Slow</td>
<td>SR</td>
</tr>
<tr>
<td>27</td>
<td>LeftCircle</td>
<td>NHD</td>
<td>NLA</td>
<td>Slow</td>
<td>LR</td>
</tr>
<tr>
<td>28</td>
<td>LeftCircle</td>
<td>NHD</td>
<td>ZA</td>
<td>Slow</td>
<td>ST</td>
</tr>
<tr>
<td>29</td>
<td>LeftCircle</td>
<td>NHD</td>
<td>PLD</td>
<td>Slow</td>
<td>LL</td>
</tr>
<tr>
<td>30</td>
<td>LeftCircle</td>
<td>NLD</td>
<td>PHA</td>
<td>Slow</td>
<td>SL</td>
</tr>
<tr>
<td>31</td>
<td>LeftCircle</td>
<td>NLD</td>
<td>NHA</td>
<td>Slow</td>
<td>SR</td>
</tr>
<tr>
<td>32</td>
<td>LeftCircle</td>
<td>NLD</td>
<td>NLA</td>
<td>Slow</td>
<td>LR</td>
</tr>
<tr>
<td>33</td>
<td>LeftCircle</td>
<td>NLD</td>
<td>ZA</td>
<td>Slow</td>
<td>ST</td>
</tr>
<tr>
<td>34</td>
<td>LeftCircle</td>
<td>NLD</td>
<td>PLA</td>
<td>Slow</td>
<td>LL</td>
</tr>
<tr>
<td>35</td>
<td>LeftCircle</td>
<td>NLD</td>
<td>PHA</td>
<td>Slow</td>
<td>SL</td>
</tr>
<tr>
<td>36</td>
<td>LeftCircle</td>
<td>ZD</td>
<td>ZA</td>
<td>Slow</td>
<td>LL</td>
</tr>
<tr>
<td>37</td>
<td>LeftCircle</td>
<td>ZD</td>
<td>PLA</td>
<td>Slow</td>
<td>SL</td>
</tr>
<tr>
<td>38</td>
<td>LeftCircle</td>
<td>ZD</td>
<td>PHA</td>
<td>Slow</td>
<td>SL</td>
</tr>
</tbody>
</table>

Individuals or current approximation, are encoded as strings, chromosomes, composed over some alphabet, so that the genotypes (chromosome values) are uniquely mapped onto the decision variable (phenotypic) domain.

Reproduction, mutation and crossover are three basic operations in evolution. In reproduction parents are carried unaltered into the next generation. Information is exchanged around a randomly generated bit position through crossover,
while in mutation a randomly generated bit position is altered to a new value. For each function, the user must define a probability that indicates the effect of that function in the evolution. In each section a random number is needed for use in the algorithm. The evolution is based on the statistical nature of these numbers and functions [24].

The following pseudocode gives an abstract view of genetic algorithm.

begin GA
  g: = 0 { generation counter }
  Initialize population P (g)
  Evaluate population P (g)
  while not done do
    g: = g+1
    Select P(g) from P (g-1)
    Crossover P(g)
    Mutate P (g)
    Evaluate P (g)
  end while
end GA

5. MEMBERSHIP FUNCTION AND WEIGHT OPTIMIZATION THROUGH GA

From the viewpoint of a genetic search the membership functions and weight of the rules can be seen as functions, the parameters of which are necessary to achieve optimization in general terms and independently of sensory application. The goal is to achieve the minimum distance and time in path tracking. Since some rules may be uncertain, the weight of the rules is optimized too. The range of membership function parameters is determined based on robot and path dimensions. Here the range defined for weights is [0. 7, 1]. The objective function in mathematical formulation as applied to this work is:

If penalty function is met

\[ \text{obfun} = \text{factor}_1 \times \sum_{i=1}^{\text{max}} \text{Distance}(i) + \text{factor}_2 \times \text{time} \]  

(13)

Else \[ \text{obfun} = 10^{20} \]  

(14)

Where penalty function is failed if \((\text{max (Distance (i))} > \text{Dmax})\) or \((\text{time} > \text{Tmax})\).

The selected values for crossover probability and mutation are 0.9 and \(14/ \text{L}_{\text{ind}}\), respectively. We have assumed that the number of generations or analyzed cycles of populations is 1000 and at first in every generation all the individuals are substituted to the next generation. The result of GA optimization for some membership functions have been shown in Figure 4.

6. SIMULATION RESULTS

Simulations are presented using a complicated desired path including several break points to show the controller performance. Figures 5 and 6 show the response of initial and optimized FLC, respectively. As can be seen, the deviation from the desired path has been extremely reduced. In next stage, some noise and delay have been added to the controller inputs and outputs in order to simulate the practical conditions in a better manner. So it is better to concern criteria seen frequently in experimental works. At any instant, the position and orientation of robot could be determined by accelerometer and tilt sensor. The amount of uncertainty in input data is in an order of 3cm for position, 5cm/s for speed and 1 degree for orientation and in the form of white noise. Sampling time is considered to be 0.1s. Simulation has been repeated considering noise on input and output signals and the result is shown in Figure 7. Finally Figure 8 illustrates the result of adding a 0.5s transport delay to the disturbances mentioned above. As mentioned earlier, the controller response has been optimized for a certain path. Because of the complexity and variety of curve sections, it is expected that the controller shows an acceptable behavior in case of any arbitrary trajectory. In order to illustrate this feature, the controller has been tested over a different path and its response is shown in Figure 9.

7. CONCLUDING REMARKS

In this paper, a fuzzy logic controller has been developed for the path tracking of a rescue robot. In order to tune the membership functions and the
Figure 4. Optimum Fuzzy Sets: (A) Distance Error (B) Orientation Error And (C) Speed.

rule weights, (to achieve minimum deviation from desired path) we have used Genetic Algorithm method. Finally for approaching to the real conditions first we exert noise in inputs and outputs of the controller. Then we added transport delay to the controller. According to the simulation results, performance of the optimized controller is acceptable even if noise, disturbance and transport delay are added to the system. Also the controller performance remains acceptable in spite of changing the desired path.

8. REFERENCES

Figure 5. Robot Path Tracking Before Optimization.

Figure 6. Robot Path Tracking after Optimization.


Figure 7. Robot Path Tracking After Optimization With Noise In Input & In Output Of FLC.

Figure 8. Robot Path Tracking After Optimization With Noise In Input & In Output Of FLC With Considering Transport Delay.

Figure 9. Testing The Optimized Robot Path Tracking Controller Over A Different Path.