Soft Computing-based Design and Control for Mobile Robot Path Tracking

Abdollah Homaifar
NASA ACE Center
Department of Electrical Engineering
North Carolina A&T State University
Greensboro, NC 27411

Daryl Battle
Lucent Technologies
67 Whippany Road
Whippany, NJ 07981

Edward Tunstel
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109

Abstract

A variety of evolutionary algorithms, operating according to Darwinian concepts, have been proposed to approximately solve problems of common engineering applications. Increasingly common applications involve automatic learning of nonlinear mappings that govern the behavior of control systems. In many cases where robot control is of primary concern, the systems used to demonstrate the effectiveness of evolutionary algorithms often do not represent practical robotic systems. In this paper, genetic programming (GP) is the evolutionary strategy of interest. It is applied to learn fuzzy control rules for a practical autonomous vehicle steering control problem, namely, path tracking. GP handles the simultaneous evolution of membership functions and rule bases for the fuzzy path tracker. As a matter of practicality, robustness of the genetically evolved fuzzy controller is demonstrated by examining the effects of sensor measurement noise and an increase in the robot's nominal forward velocity.

1. Introduction

In recent years, increased efforts have been centered on developing intelligent control systems that can perform effectively in real-time. These include the development of non-analytical methods of soft computing such as evolutionary computation and fuzzy logic. These methods have proven to be effective in designing intelligent control systems and handling real-time uncertainty, respectively [1, 2]. In this paper, our efforts are focused on combining these paradigms to develop path tracking controllers for autonomous vehicles such as mobile robots and automated guided vehicles (AGVs). Specifically, we employ genetic programming (GP) for off-line learning of path tracking rules to be implemented in a fuzzy logic controller.

Genetic programming [3] has recently been demonstrated to be a viable approach to learning fuzzy logic rules for mobile robot control and navigation [4, 5]. Herein, we address the simultaneous design of fuzzy logic controllers (FLCs) using GP, i.e. evolution of both the input membership functions and the rule base. In addition, we extend the evolutionary influence of GP by incorporating the random selection of fuzzy logic connectives (t-norms) into the learning process. Finally, we examine the robustness of the evolved controllers by corrupting sensory data used by the path following robot, and by increasing the nominal forward velocity of the vehicle. This provides an indication of how well GP can evolve practical solutions that also retain the tolerance of imprecision and uncertainty characteristic of FLCs.

2. Overviews of Fuzzy Control and GP

A FLC is an intelligent control system that smoothly interpolates between rules, i.e. rules fire to continuous degrees and the multiple resultant actions are combined into an interpolated result. A fuzzy set may be represented by a mathematical formulation known as a membership function. That is, associated with a given linguistic variable (e.g. *speed*) are linguistic values or fuzzy subsets (e.g. *slow*, *fast*, etc.) expressed as membership functions which represent uncertainty, vagueness, or imprecision in values of the linguistic variable. These functions assign a numerical degree of membership to a crisp (precise) number. More precisely, over a given universe of discourse (relevant numerical range) X, the membership function of a fuzzy set, denoted by $\mu(x)$, maps elements $x \in X$ into a numerical value in the closed unit interval, i.e. $\mu(x): X \rightarrow [0, 1]$.

Implementation of a fuzzy controller requires assigning membership functions for inputs and outputs. Inputs to a fuzzy controller are usually measured variables, associated with the state of the controlled plant, that are fuzzified (assigned membership values) before being processed by an inference engine. The heart of the controller inference engine is a set of *if-then* rules whose antecedents and consequences are made up of linguistic variables and associated fuzzy membership functions. Consequences from fired rules are numerically aggregated by fuzzy set union and then collapsed (defuzzified) to yield a single crisp output as the control signal for the plant. For detailed introductions to fuzzy control, fuzzy set operations, and concepts of fuzzification, inference, aggregation, and defuzzification see one of [2, 6].

In the GP paradigm, a population is comprised of computer programs or procedures (individuals) that are candidate solutions to a particular problem. These individuals participate in a simulated evolution process wherein the population evolves over time in response to selective pressure induced by the relative fitness of individuals in the problem domain. In our approach, each program executes condition-action statements, which collectively serve as a rule base to be embedded in a fuzzy controller. To preserve diversity among populations and vital genetic information among individuals, genetic operators are applied to create new individuals for succeeding generations. When the algorithm finally converges or satisfies its termination criteria, it is anticipated that the best (most fit) individual will be representative of an optimum or near optimum solution.

In the next section, we introduce the autonomous vehicle control problem, followed by discussion of FLC design issues to be considered when employing GP.

3. Mobile Robot Path Tracking Problem

The control problem examined in this paper is a path tracking problem, which was formulated by Hemami et al [7, 8] for a class of low speed (less than 2 m/s) tricycle-model vehicles. Essentially, the control objective is to successfully navigate a mobile robot or AGV along a desired path in a two-dimensional environment. We wish to automatically design a fuzzy controller that will achieve this objective. The inputs consist of a measurable position error, ε_d, and a measurable orientation error, ε_θ, associated with path following in the plane (see Figure 1). The output is the steering angle, δ, which is the corrective control action that would cause the errors to approach zero and, thus, force the robot to follow the desired path.

The position error is taken as the deviation of the center of gravity, *C*, or any other desired point of the robot from the nearest point on the path. The orientation error is the angular deviation of the robot from the tangent of the desired path. Hemami et al derived a state-space kinematic model for this robot where the state vector was comprised of the pose errors described. The reader is referred to either of [7] or [8] for details of the model derivation, which culminates in the following:

$$\begin{bmatrix}\dot{\varepsilon}_d \\ \dot{\varepsilon}_\theta\end{bmatrix} = \begin{bmatrix}0 & V_u \\ 0 & 0\end{bmatrix}\begin{bmatrix}\varepsilon_d \\ \varepsilon_\theta\end{bmatrix} + \begin{bmatrix}MC/MP \\ 1/MP\end{bmatrix} V_u \tan\delta \pm \begin{bmatrix}\dot{\eta}_d \\ \dot{\eta}_\theta\end{bmatrix} \quad (1)$$

where V_u is forward linear velocity of the robot, and $\dot{\eta}_d$ and $\dot{\eta}_\theta$ are rates of change of the effects of path curvature. In [8] it is concluded (based on dynamic analysis of the same vehicle) that for small steering angle, δ ($\tan\ \delta \cong \delta$), Equation (1) approximates the slow dynamics of the vehicle when its forward velocity is low. For simulations presented later, we have simplified the robot kinematic model by taking this small steering angle approximation into account. Furthermore, we apply the controller to straight-line path following and, therefore, neglect the model effects of path curvature. Such a simplification does not preclude autonomous tracking of reasonably complicated paths since multi-segment paths can be defined to be piecewise linear.

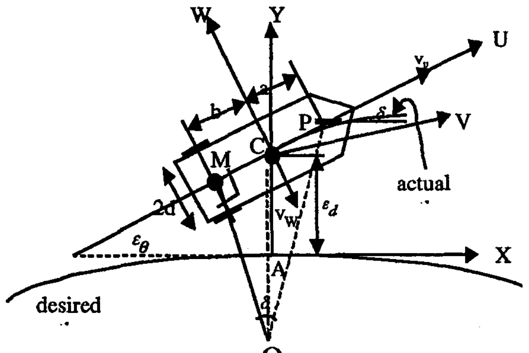

Figure 1. Tracking control and error variables.

For our application, we assume that the robot has dead-reckoning/odometry sensors that provide access to the error states at all times, or permit calculations thereof. This sensory input data is then mapped to control outputs according to the desired control policy. In path following simulations the state vector of the kinematic model is updated using the well-known fourth-order Runge-Kutta numerical integration method.

4. GP Design of Fuzzy Controllers

The path tracker to be learned by GP is a two-input, single-output fuzzy controller that will map the error states into a proper steering angle at each time step. A population of candidate solutions is created from which a solution will emerge. The allowance for rule bases of various sizes enhances the diversity of the population. That is, the GP system creates individuals in the initial population that each have possibly different numbers of rules within a range (15-30) specified before a run. In the process of learning fuzzy control rules and membership functions, GP manipulates the linguistic variables directly associated with the controller. Given a desired motion behavior, the search space is contained in the set of all possible rule bases that can be composed recursively from a set of functions and a set of terminals. The function set consists of membership function definitions (describing controller inputs), components of the generic fuzzy *if-then* rule, and common fuzzy logic connectives. More specifically, these include functions for fuzzy sets, rule antecedents and consequents, fuzzy set intersection and union, and fuzzy inference. The terminal set is made up of the input and output linguistic variables and the corresponding membership functions associated with the problem.

Selection of appropriate t-norms is automated, thereby, giving the GP system greater control of the evolutionary design. That is, the influence of GP is extended to include selection of the type of t-norm employed to compute the conjunction of fuzzy propositions in the antecedent of a rule. The two most commonly used t-norms for fuzzy control are Mamdani's *min* and Larsen's *product* [6]. T-norms for each conjunctive rule are selected at random by GP for rule bases in the initial population, and are carried along based on fitness in successive generations.

To achieve the goal of evolving FLCs, the GP system must conform to strong syntactic constraints when breeding individuals. Special rules of construction were introduced in [4]. In order to accommodate evolution of input membership functions, in addition to the rule base, the allowable syntax was extended in [9] and implemented using a representation proposed in [1]. We refer the reader to [4] and [9] for more detail on the syntactic constraints and other GP implementation issues related to fuzzy controller design.

4.1. Controller Fitness Evaluation

Each rule base in the current population is evaluated to determine its fitness value for steering the robot from initial locations near the desired path to final locations on the path such that steady state and final pose errors are minimized. This evaluation involves frequent simulation of the robot's motion from each of a finite number of initial conditions until either the goal state is achieved or the allotted time expires. The initial conditions are referred to as *fitness cases* in the GP community. For this problem we use eight different initial conditions, which is a logical choice given the pair-wise symmetry of the possible error categories illustrated in Figure 2. Consider error category (d), which represents a case where the robot is located on the left of the desired path with a negative heading orientation. There also exists a symmetric case where the robot is located on the right of the desired path with a positive heading orientation. These symmetric cases are each represented by error category (d). The same holds for category (a), (b) and (c) illustrated in the figure, yielding a total of eight fitness cases that fully describe the possible combinations of errors with respect to the path.

The fitness function is a measure of performance used to rank each individual relative to others in the population. We compute path tracking performance by summing the Euclidean norms (normalized) of the final error states plus the average control effort ($\overline{\delta}$) over all eight fitness cases. Thus, the following fitness function drives the evolution process

$$\textit{Raw Fitness} = \sum_{i=1}^{8} \sqrt{(\varepsilon_d^2 + \varepsilon_\theta^2 + \overline{\delta}^2)_i} \qquad (2)$$

where ε_d and ε_θ are the position error and orientation error existing at the end of each fitness case simulation. The objective of this fitness function is to minimize final path tracking errors as well as the control effort expended. As such, a perfect fitness score is zero and, in general, lower fitness values are associated with better controllers. Simulations thus far showed that including $\overline{\delta}$ as part of the path tracking metric significantly reduces undesired steering oscillations. Fitness functions based solely on final error states sometimes yielded impractical controllers that exhibited rapid oscillations in the steering control signal, which would cause damage to the steering mechanism of a real mobile robot.

The path tracking success of an individual in the population is also based on its ability to minimize tracking errors to within the following specified tolerances, $|\varepsilon_d|<0.15$m and $|\varepsilon_\theta|<0.26$ rad., for each fitness case. A fitness case simulation in which these tolerances are satisfied is considered a hit, or successful trial. Thus, each individual has the potential of receiving a total of eight hits during fitness evaluation.

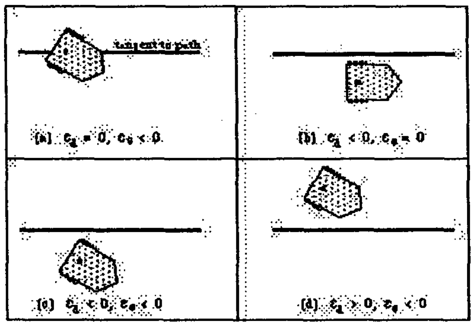
tangent to path
(a) $\varepsilon_d = 0, \varepsilon_\theta < 0$
(b) $\varepsilon_d < 0, \varepsilon_\theta = 0$
(c) $\varepsilon_d < 0, \varepsilon_\theta < 0$
(d) $\varepsilon_d > 0, \varepsilon_\theta < 0$

Figure 2. Error categories for control problem.

5. Path Tracking Results

In this section, we present representative results of simulated path tracking performance for an evolved controller. The simulated robot is based on Hemami's kinematic model with dimensions taken from the Hero-1 mobile robot. The Hero-1 has a tricycle wheel configuration in which the front wheel is driven by a DC motor and steered by a stepper motor. Its two rear wheels are passive. Dimensions employed are 0.3m for the wheelbase, and 0.2m for the offset from the rear axle to the front wheel. These dimensions correspond to the constant lengths *2d* and *MP* of Figure 1, respectively. All simulations were conducted assuming a controller sampling rate of 20 Hz and run for a maximum of ten seconds. In each case, the robot travels at a constant nominal forward speed of 1.5 m/s unless otherwise stated.

The GP system was implemented in the C programming language on a 260 MHz MIPS DECstation. Five consecutive runs (initialized using different random number generator seeds) were executed on a population of 200 individuals for a maximum of 50 generations. About one hour of computation time is required for a run of this magnitude. A rule base of 25 rules emerged as the fittest among all five runs. This rule base used five conjunctive rules, three employing the Mamdani t-norm and two employing the Larsen t-norm. The evolved input membership functions associated with the best rule base are shown in Figure 3 and the rules are listed in Table 1. The notations *NB, NS, Z, PS,* and *PB* represent fuzzy linguistic terms of "negative big", "negative small", "zero", "positive small", and "positive big", respectively. Terms describing the inputs, ε_d and ε_θ, are preceded with the prefix "p" and "o" respectively. The fixed output membership functions are shown in Figure 4, where the linguistic terms are labeled without prefixes.

The evolved controller received a raw fitness of 0.1091 with 8 hits. In [4], an FLC designed manually, through a lengthy process of trial-and-error, was presented which also used 25 rules. Hours of iterative refinement of membership functions and rules were invested before arriving at a suitable design. In comparison, the hand-derived FLC received a comparable raw fitness (0.08 with 8 hits) for the identical tracking problem. Figure 5 shows the temporal responses of position error, orientation error, and control effort for the evolved controller and for the hand-derived controller. This result corresponds to error category (d) of Figure 2, with initial conditions of ε_d = 0.8 m and ε_θ = -0.9 rad. In [8] it was shown that this error category is the most general for studying path tracking by tricycle-type vehicles. It is most general in the sense that in the process of correcting vehicle steering from initial states in all other error categories, the vehicle error status ultimately reduces the category (d) of Figure 2 or its counter-pair. In all fitness cases, the evolved controller achieved comparable response characteristics to those of the hand-derived controller using an equivalent number of rules.

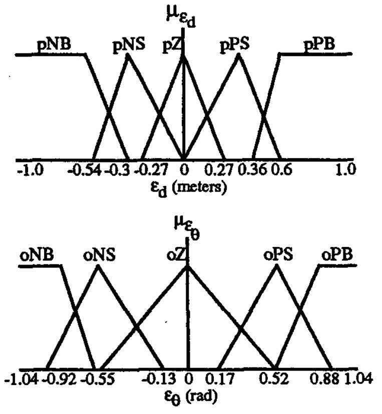

Figure 3. Co-evolved input membership functions.

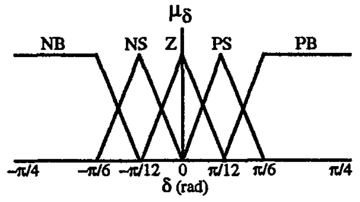

Figure 4. Output membership functions.

TABLE 1. Best Evolved Rule Base

1	IF oZ THEN NS
2	IF pPB THEN Z
3	IF pNB THEN Z
4	IF pPS THEN NB
5	IF pNS and oPS THEN NS (Mamdani's *min*)
6	IF pNB THEN PB
7	IF oNS THEN Z
8	IF oNB THEN PS
9	IF pNS THEN NS
10	IF pNS and oZ THEN PB (Larsen's *prod*)
11	IF oPB THEN NB
12	IF pNS and oPB THEN NB (Larsen's *prod*)
13	IF pPS THEN NS
14	IF oNS THEN PB
15	IF pPB THEN NB
16	IF oZ THEN PS
17	IF oNB THEN PB
18	IF pNS and oNS THEN PB (Mamdani's *min*)
19	IF pNS THEN Z
20	IF oPS THEN NB
21	IF pZ THEN PS
22	IF pPB and oZ THEN Z (Mamdani's *min*)
23	IF pPB THEN PS
24	IF oPS THEN PS
25	IF oNS THEN PS

6. Robustness Characteristics

Given the capability to evolve FLCs that can effectively follow paths, an important next step is to examine their robustness to practical perturbations. To test the noise robustness of the evolved controller, simulations were performed with the imposition of a noise signal upon the sensor measurement related to heading (orientation). We assume that the error states are derived from sensor measurements which, due to their imperfect nature, introduce an additive sinusoidal noise signature of small amplitude and low frequency (relative to the controller sampling frequency) that corrupts the orientation error. For this investigation we impose the sensor noise signal, $n(t) = 0.15\cos(3t)$ with $t = kT$, where k=1,2,3,... is the sampling instant, and T is the sampling period. Thus, the noise amplitude is bounded by 0.15 radians (10 degrees), and at any sampling instant the corrupted orientation error signal lies in the range of ($\varepsilon_\theta \pm 0.15$) radians.

In addition to the additive noise, we also increased the constant nominal forward speed of the robot by 20%, which resulted in a simulated speed of 1.8m/s. A typical result is shown in Figure 6, which illustrates the performance of both the evolved controller and the hand-derived controller when induced with noise and an increased vehicle speed. While the oscillatory effects of the added noise are clearly evident in the steady state response, the controller successfully navigates the robot onto the path and maintains the steady state errors within the tolerances specified earlier. Thus, this evolved fuzzy controller exhibits path tracking robustness to the imposed perturbations. This result is representative of temporal responses for each of the remaining fitness cases. In simulations completed thus far, the most robust fuzzy controllers were those evolved when GP was allowed to randomly select t-norms.

The performance assessment of the evolved controller with regard to robustness is based upon the assumption that low frequency oscillations within the control signal of amplitude less than 0.026 radians (1.5 degrees) are practical. In light of this assumption, the results indicate that the evolved FLC was able to navigate the robot along the desired path with the imposed perturbation of sensor noise and the increase in the robot's nominal speed.

7. Summary and Conclusions

This paper has demonstrated an approach to path tracking controller design based on soft computing methods. GP was successfully applied to discover fuzzy controllers capable of navigating a mobile robot to track straight-line paths in the plane. The performance of the best-evolved FLC was comparable to that of a manually derived FLC, which required a considerably longer design cycle. GP simultaneously evolved membership functions and rules for an FLC that demonstrated satisfactory responsiveness to various initial conditions while utilizing minimal human interface. The speed of evolution alone serves as a strong basis for practical application of GP in the controller design process. The approach enables expeditious design of FLCs that can be directly applied to a physical system. Alternatively, human experts can use the rapidly evolved FLCs as design starting points for further manual refinement. Finally, the evolved FLC was shown to be robust to perturbations of sensor noise and an increase in nominal robot speed. This supports the notion that genetically evolved FLCs can have practical utility.

Acknowledgments
This work is partially funded by grants from NASA Autonomous Control Engineering Center (ACE) at North Carolina A&T SU under grant # NAG2-1196 and NASA Dryden Flight Research Center under grant # NAG4-131. The authors wish to thank the ACE Center and NASA Dryden for their financial support.

References

[1] Homaifar, A. and McCormick, E., "Simultaneous Design of Membership Functions and Rule Sets for Fuzzy Controllers Using Genetic Algorithms", *IEEE Transactions on Fuzzy Systems*, Vol. 3, No. 2, 1995, pp. 129-139.

[2] Jamshidi, M., Vadiee, N. and Ross, T. (Eds.), *Fuzzy Logic and Control: Software and hardware applications*, Prentice-Hall, Englewood Cliffs, NJ, 1993.

[3] Koza, J. R., *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press, Cambridge, MA, 1992.

[4] Tunstel, E. and Jamshidi, M., "On Genetic Programming of Fuzzy Rule-Based Systems for Intelligent Control", *International Journal of Intelligent Automation and Soft Computing*, Vol. 2, No. 3, 1996, pp. 271-284.

[5] Tunstel E., Lippincott, T. and Jamshidi, M., "Behavior Hierarchy for Autonomous Mobile Robots: Fuzzy-behavior modulation and evolution", *International Journal of Intelligent Automation and Soft Computing*, Vol. 3, No. 1, 1997, pp. 37-49.

[6] Lee, C., "Fuzzy Logic in Control Systems: Fuzzy Logic Controller, Part I", *IEEE Transactions on Systems, Man & Cybernetics*, Vol. 20, No. 2, 1990, pp. 404-418.

[7] Hemami, A., "Steering Control Problem Formulation of Low-Speed Tricycle-Model Vehicles", *International Journal of Control*, Vol. 61, No. 4, 1995, pp. 783-790.

[8] Hemami, A., Mehrabi, M., and Cheng, R., "Optimal Kinematic Path Tracking Control of Mobile Robots with Front Steering", *Robotica*, Vol. 12, No, 6, 1994, pp. 563-568.

[9] Battle, D. D., *Implementation of Genetic Programming for Mobile Robot Navigation*, MS Thesis, Dept. of Electrical Engineering, North Carolina A&T State University, Greensboro, NC, 1998.

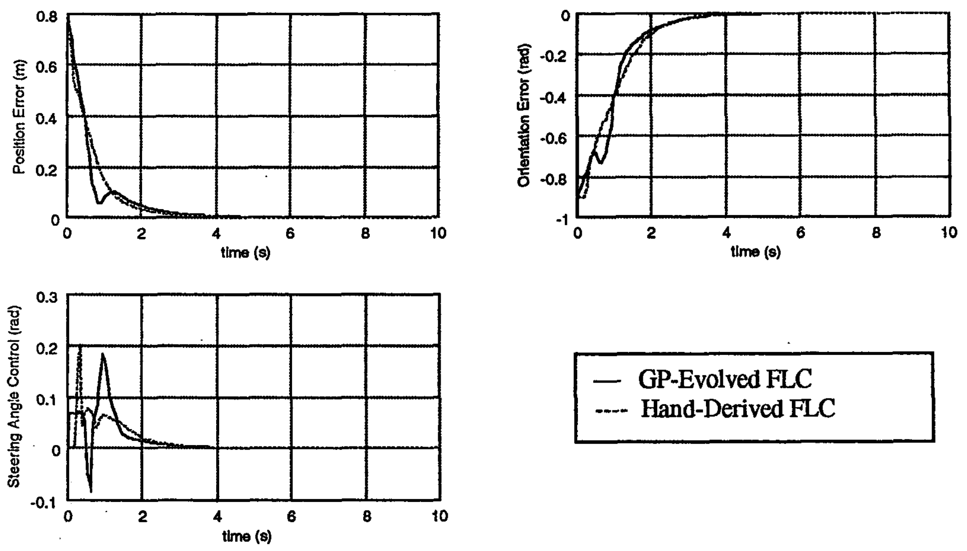

Figure 5. Evolved FLC path tracking performance.

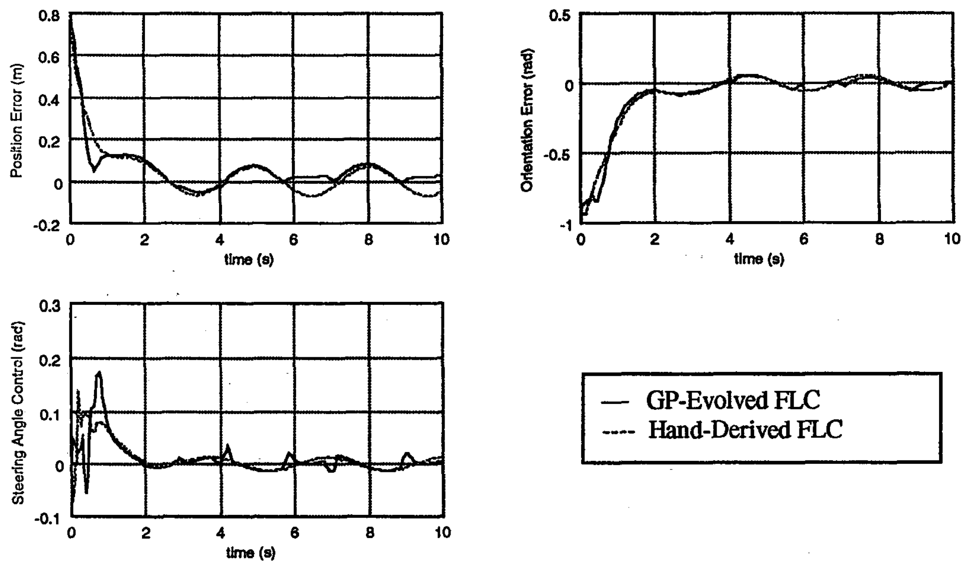

Figure 6. Evolved FLC response to sensor noise and increased forward speed.