Evolutionary Design of Fault-Tolerant Analog Control for a Piezoelectric Pipe-Crawling Robot

Geoffrey A. Hollinger
Robotics Institute
Carnegie Mellon University
Pittsburgh, PA USA
01-904-993-1584
gholling@andrew.cmu.edu

David A. Gwaltney
Embedded Control Design Team
NASA Marshall Space Flight Center
Huntsville, AL USA
01-256-544-4231
David.A.Gwaltney@nasa.gov

ABSTRACT
In this paper, a genetic algorithm (GA) is used to design fault-tolerant analog controllers for a piezoelectric micro-robot. First-order and second-order functions are developed to model the robot's piezoelectric actuators, and the GA is used to evolve closed-loop controllers for both models. The GA is first used to assist in traditional PID design and is later used to synthesize variable topology analog controllers. Through the use of a compact circuit representation, runtimes are minimized and controllers are synthesized with minimum population sizes and components. Fault-tolerance is built into the fitness function to facilitate the design of controllers robust to both actuator failure and component failure. The GA is successfully used to design synthetic controllers and to optimize a traditional PID design. This research shows the advantages of GA assisted design when applied to robot-control problems.

Categories and Subject Descriptors
B.8.1 [**Hardware**]: Reliability, Testing, and Fault-Tolerance.

General Terms
Algorithms, Design, Reliability.

Keywords
genetic algorithms, piezoelectric actuators, evolvable hardware, robot control, inspection robots.

1. INTRODUCTION
In electronic applications, particularly in aerospace, fault-tolerance is an important design goal. When electrical components fail, it is desirable for circuits to maintain some of their original functionality. For this reason, redundant circuitry has become commonplace in analog circuit design. Additionally, it is desirable for circuitry to be robust to environmental changes.

When controlling robots, such changes often represent wear-and-tear on the robot or minor failures within the actuators. Often, it is difficult to design controllers that are robust to both internal and external failures.

GECCO'06, July 8–12, 2006, Seattle, Washington, USA.

Evolutionary design using genetic algorithms (GAs) and genetic programming (GP) has the potential to solve many of the problems associated with robust design. For instance, GAs optimize designs by simulating many different designs, keeping the designs that perform favorably, and sharing characteristics between successful designs. When evolving analog circuitry, each circuit can be represented by a string of bits. The GA initializes a population of circuits to a random string of bits and evaluates these circuits based on a specified fitness function. After evaluating all members of the population, the bit strings are modified using genetic operators (crossover and mutation) to generate a new population, which is then stochastically sampled to determine the next generation of circuits. After many generations, the population contains circuits well suited for the task. Genetic programming uses similar genetic operators but, instead, represent circuit components as tree structures or computer programs.

Evolutionary analog circuitry design has been successful at fabricating opamps, filters, and amplifiers. Koza presents many techniques for designing various types of analog components using genetic programming [8]. Using a variable topology representation, he is able to successfully fabricate filters, amplifiers, and robot controllers at the transistor level. While this work is promising, Koza uses large population sizes, and his runtimes restrict evolvable circuitry to multi-computer setups. Additionally, Dastidar, Chakrabarti, and Ray develop a genetic programming solution for synthesizing circuits from circuit "building blocks" [2]. This technique, however, places a good deal of emphasis on the selection of the building blocks and takes freedom away from the evolutionary algorithm.

In [10,11], Lohn and Colombano use GAs to develop a compact circuit representation that decreases the complexity of evolutionary design of analog circuitry. While their representation restricts potential topologies, it decreases runtimes significantly by maximizing the number of useful circuits in the population. Lohn and Colombano successfully fabricated both lowpass filters and amplifiers up to 85 dB using their compact circuit description.

GAs have also been successful when applied to analog control tasks. In [18], Zebelum and Pacheco synthesize analog controllers for three common transfer functions. They present a dynamic fitness function using rise time, overshoot, and mean-squared error along with noise and circuit dissipation. They also add fault-tolerance to their fitness function by removing each component in

their controllers, simulating with the faulty component, and combining the fitness functions of each simulation. Using this technique, they successfully develop controllers capable of losing any one of their components and maintaining reasonable responses. Unlike in [8,10,11], Zebelum and Pacheco pre-specify the number of internal and external circuit nodes prior to synthesis. While successful, Zebelum's technique greatly limits the number of possible topologies during controller synthesis and, thus, excludes potentially useful controllers from entering the population.

Hu and Goodman present further research in developing fault-tolerant analog circuitry using genetic programming [6]. They use a bond tree representation to design analog filters while taking advantage of a hierarchical fair competition algorithm to maximize sustainability. While their method is promising, the bond tree structure limits the freedom of the evolutionary algorithm, and they do not apply their algorithm to control tasks.

Additional work on fault-tolerance using GAs is presented by Keymeulen et al. [7]. They discuss two techniques for developing fault-tolerant transistor array configurations: one extracts from a population of evolved circuit the circuits best able to recover from faults, and one attempts to evolve circuits in the population that are resistant to faults by adding fault tolerance to the fitness function. The latter approach is similar to the one used in [18] and is the approach taken in this paper.

The goal of this paper is to design fault-tolerant analog controllers for the actuators on a piezoelectric pipe-crawling inspection robot for use in microgravity. Two models are developed for the robot actuators, and a GA is used to synthesize controllers for both models. A modified version of the compact circuit representation in [10,11] is used by a GA to evolve variable topology controllers with minimum population sizes and runtimes. Several fitness functions, including one similar to the one in [18], are used to determine fitness. Fault-tolerance is achieved by simulating each of the controllers with components removed and actuator models changed. This facilitates the evolution of controllers robust to both internal circuit failures and external actuator failures. Additionally, a GA is used to optimize the components on a PID controller (using a similar procedure as [17]) to serve as a gold standard against the automatically synthesized variable topology filters.

This paper is organized as follows: Section 2 describes the piezoelectric pipe-crawling robot and its actuators. Section 3 describes the models derived for the piezoelectric actuators, and Section 4 describes the genetic algorithm used. Section 5 presents the GA assisted PID controller design, and Sections 6-8 describe variable topology evolution using both models without fault-tolerance. Finally, Section 9 presents the design of fault-tolerant controllers for the robot.

2. THE PIEZO ROBOT

In previous research, a piezoelectric pipe-crawling robot was designed for use in the inspection of the coolant pipes on the United States Space Shuttle Main Engine [5]. The "piezo robot" has three piezoelectric actuators, one middle actuator, and two front and back clamping actuators. The robot moves similar to an inchworm by clamping and unclamping the front and back actuators and extending the middle actuator. For instance, the robot will unclamp the front actuator, extend the middle actuator, clamp the front actuator, unclamp the back actuator, and then compress the middle actuator. Figure 1 shows a picture of a manufactured prototype of the piezoelectric robot, and Figure 2 shows a simulated design of the initial robot.

The initial piezo robot design was constructed to fit the mechanical specifications from Dynamic Structures and Materials, LLC, and this design was later optimized using computer simulated genetic optimization [5]. The genetic optimization was performed with an embedded GA using the Darwin2k simulation software [9]. The genetic optimization of the robot's design further proves the effectiveness of evolutionary design in robotic applications.

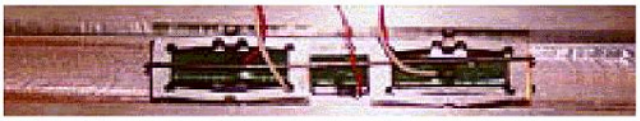

Figure 1. Picture of piezelectric robot prototype

Figure 2. Simulated piezo robot prototype

The front and back actuators on the piezo robot are based on the DSM FPA-900 design. The FPA-900 has a maximum stroke of 900 μm, operates at 150V, and has a resonant frequency of approximately 300 Hz [3]. This paper focuses on the operation of the front and back FPA-900 actuators on the piezo robot. Figure 3 shows pictures of a manufactured piezoelectric actuator, and a picture of the actuator as simulated in [5].

Figure 3. Manufactured FPA-900 actuator (left) and simulated actuator (right)

To drive the piezo robot's actuators, a controller with three 0 to 150 V inputs must be used. Piezoelectric actuators can be driven using both open-loop and closed-loop methods. When driven open-loop, the controller output is a non-sinusoidal wave of voltages applied to the piezo. Due to the piezoelectric effect, the extension of the piezo will be proportional to the input voltage. The drawback of open-loop control is that hysteresis prevents precise positioning of the piezo actuators. Additionally, driving frequencies of open-loop controllers are limited to one-third of the resonant frequency to avoid damaging the piezoelectric actuators [14]. In the case of the FPA-900, open-loop control is limited to driving frequencies around 100 Hz [3].

Options for evolving open-loop controllers for the piezo robot using a GA were explored but later abandoned. Based on previous work [8,10,11,16], it seemed unlikely that a GA could successfully fabricate the high-power amplifiers necessary for open-loop amplification and control. Additionally, work on

charge recovery [12] and complementary waveform behavior [15] would be quite difficult or impossible to replicate using a GA. For these reasons, this paper focuses on closed-loop control of the piezo robot. Closed-loop control also has the potential to drive the actuators near their resonant frequency and increase the speed of the piezo robot beyond that achieved through open-loop control. In the following section, two models of the FPA-900 actuator are developed, and the simulated plant output is used in a closed-loop controller. Strain gauges will need to be placed on the actual robot to supply this information in a real-world environment.

3. PIEZOELECTRIC ACTUATOR MODELING

To simulate the behavior of the piezo robot, models needed to be developed for the piezoelectric actuators. The first model was developed under the assumption that piezo actuators act similar to a capacitor. This assumption comes from the behavior of the actuators when subjected to increasing voltages. As voltage increases, the actuators show a proportional displacement. Ideally, the relationship between voltage and displacement is fully linear and analogous to the charge voltage relationship of a capacitor. In practice, piezoelectric actuators exhibit hysteresis, and their displacement characteristics fail to be linear.

To account for the non-linearity in the displacement characteristic, a resistor was placed in series with the modeling capacitor. The value of the resistor corresponds to the area of the hysteresis loop in the voltage displacement characteristic. The equivalent RC circuit acts similarly to the piezoelectric actuator if the input to the circuit is varying (i.e. sinusoidal). From the FPA-900 datasheet, the value for the capacitive load was determined to be 10 μF, and the resistance to be 10 Ω. For this model, the system output is the voltage across the capacitor. This modeling technique is similar to the one used in [1] for the development of a charge recovery method for driving piezoelectric circuits.

While the leaky capacitor model described above provides a simple method for modeling the piezoelectric actuators, it does not account for the underdamped nature of the piezoelectric system. When placed in a closed loop, piezoelectric actuators are subject to significant ringing, especially when driven near their natural frequencies [14]. To better account for this property, a second-order model was developed based on the frequency characteristic of the actuators.

By fitting the actuator frequency characteristic in [3] to a second-order model, the transfer function in equation (1) was developed.

$$TF = \frac{90{,}000}{s^2 + 30s + 90{,}000} \qquad (1)$$

This transfer function correctly places the natural frequency at 300 Hz and exhibits damping similar to that found in the actual actuator. To maintain unity gain, the voltage to displacement conversion was done after the plant model in the control loop. The transfer function above was implemented using a second-order Sallen-Key lowpass filter with unity gain and equal resistor values. The output of this model was scaled such that it represents the actual displacement of the actuators. Figure 4 gives a graph of the resulting frequency characteristic.

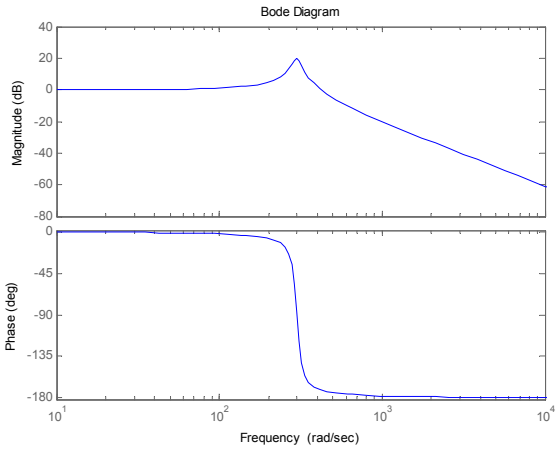

Figure 4. Frequency characteristic of second-order actuator model

4. GENETIC ALGORITHM DESCRIPTION

To implement the GA necessary for circuit evolution, the Genetic Algorithm Utility Library (GAUL) was used. GAUL provides a modular GA evolution environment and allows the user to specify crossover and mutation methods, statistical selection, elitism, and other GA parameters [4]. Based on previous work, the crossover and mutation rates were set to 80% and 20% respectively [8,10,11]. Double point crossover and tournament selection were used, and parents were allowed to survive into subsequent generations. This type of elitism ensured that fit circuits were not sampled out for their inferior children. Population sizes varied on each run, but were set between 100 and 1000 members.

Fitness was determined using the transient analysis mode of the NGSPICE circuit simulator on a single 3.2 GHz P4 Processor running Fedora Linux. NGSPICE is built on the Berkeley SPICE simulator and is written in modern C++ and optimized for use in Linux. NGSPICE was modified slightly from its release to quit after failing to converge after a specified number of iterations. NGSPICE was run as a separate process from GAUL and interfaced using file I/O functions in C++.

Two fitness functions were used in different runs to evaluate the performance of the evolved control systems. In both cases, the goal is to minimize the fitness measure. The first function compares the squared errors between the transient response and an ideal step response. The integral squared error was determined for an entire cycle rather than just a single rise to eliminate circuits that cannot return to zero after the step response. The numerical integral of the squared error was determined for all N transient step points according to equation 2.

$$Fitness = ISE = \sum_{i=0}^{N} \left(V_i - V_{target}\right)^2 * \left(t_i - t_{i-1}\right) \qquad (2)$$

The second fitness function, given in equation 3, was found using a weighted average of the rise time, overshoot, and integral of the squared error (as above) of the step response. After each generation, the weights of each component of the fitness function are adjusted based on the error between the population averages

and a user-specified value as in [18]. This technique provides feedback to the fitness function and determines which metrics should be optimized in the subsequent generation. A comparison of results using the two fitness functions is presented in Section 8.

$$Fitness = w_{rise} \cdot t_{rise} + w_{over} \cdot V_{over} + w_{ise} \cdot ISE \quad (3)$$

5. PID CONTROLLER OPTIMIZATION

To begin evolutionary design, the GA was used to optimize the constants in a fixed topology analog PID controller. A closed loop PID controller was implemented in NGSPICE as in [13]. The proportional stage was implemented using an inverting opamp amplifier configuration, and the integral and derivative stages were implemented using a typical first-order integrator and differentiator circuit. The three components of the controller were then summed together using an opamp summing configuration. The GA was set to modify the resistor values for each section, thus changing the PID constants of the controller. The GA was allowed to vary the proportional component between 0.1 and 1000, the derivative component between 0.0001 and 1, and the integral component between 0.01 and 100 with 15 bit resolution.

Five GA optimization trials were run for the PID controller using the integral squared error fitness function. The three fittest circuits for both actuator models are presented in Tables 1-2 and Figures 5-6. The GA optimized the PID controllers by giving the capacitor model a high proportional gain to reduce steady-state error and a moderate integral gain to speed up the circuit. For the second-order model, the GA increased the derivative gain to smooth the ringing. The GA achieved similar results to those that a human designer would in this situation.

Table 1. Control statistics for optimized PID controllers

Model	Settling Time (μs)	Overshoot (%)	SS Error (%)	Fitness (ISE)
Capacitor	5.58	17.3	1.54	0.0320
	4.94	17.9	1.74	0.0319
	5.16	17.5	1.72	0.0319
2nd Order	455	11.6	1.58	0.0683
	458	12.3	1.51	0.0683
	456	11.9	1.57	0.0683

The best design for the capacitor model had a settling time of 5 μs.[1] If the piezo actuator is driven by a square wave with a period of twice the settling time, this allows for operating frequencies up to 100 kHz. However, since the capacitor model does not account for ringing in the actuators, these speeds are likely not attainable. The best design for the more realistic second-order model had a settling time of 455 μs allowing for operating frequencies up to 1 kHz. This is a magnitude of ten faster than the suggested open-loop operating frequency of 100 Hz and is more than three times the actuator's resonant frequency. This speed increase shows the advantage of closed-loop control for piezoelectric actuators.

[1] The closed-loop capacitor circuit without a controller had a settling time of 116 μs, and the second-order circuit had a settling time of 142 ms.

Table 2. Evolved constants for optimized PID controllers

Model	KP	KI	KD
Capacitor	66.5	0.0103	8.02E-04
	59.6	0.0472	9.54E-04
	60.1	0.0794	9.24E-04
2nd Order	43.0	0.0201	0.0897
	47.0	0.0206	0.0898
	43.4	0.0126	0.0899

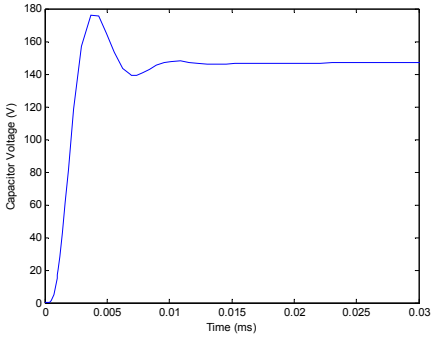

Figure 5. Capacitor model response for evolved PID controller

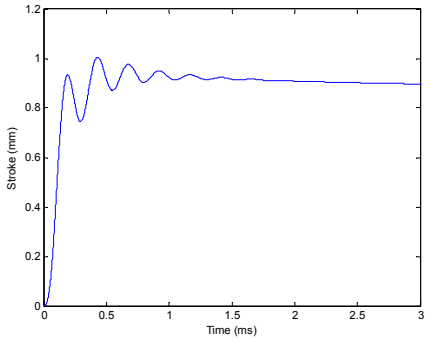

Figure 6. Second-order model response with evolved PID controller

6. CIRCUIT REPRESENTATION FOR AUTOMATED SYNTHESIS

In attempt to implement a fast, compact description of analog circuitry, a circuit representation based on the one in [10,11] was used to describe variable topology controllers. This technique uses an incremental approach to constructing circuits from bit strings with each portion of the bit string representing a component in the circuit. Starting with the circuit input as the active node, the circuit decoder steps through the circuit and places components according to their bit strings. The description implemented for the piezo robot uses fifteen bits for each circuit component. The first five bits determine whether the component is a resistor, capacitor, inductor, or bipolar junction transistor and how that component is placed in the circuit. The five bits allow for six potential placement configurations for each component. The first terminal of the component is fixed to the current active node, and the second terminal is fixed to ground, input, output,

power source, the previous active node, or a new node (which becomes the active node) depending on the bit code.

For transistors, the first five bits only account for two of the terminal connections. The next ten bits determine the connection of the third terminal (also to ground, input, output, power source, previous active, or next active) and the order of connection. Both NPN and PNP transistors were used, and transistors were realistically modeled after the popular m2n2222 (NPN) and m2n2907 (PNP) designs. For RCL components, the last ten bits determine the value of the component. Resistor values were scaled linearly between 10 Ω and 100 kΩ, capacitor values between 1 nF and 10 μF, and inductor values between 1 mH and 0.5 H. Figure 7 shows a diagram of a circuit bit string.

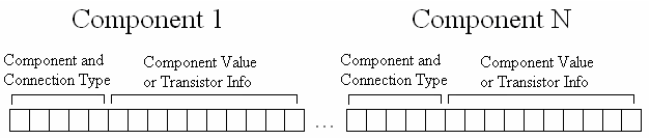

Figure 7. Variable topology instruction set format

The circuit description in this paper differs from the one used in [10,11] in that it is not variable length. A maximum number of elements are specified, and the number of bits in each circuit representation is always fifteen times that number. In some circuits, a stop sequence is reached, and the circuit terminates before placing all components. This maintains full functionality of the variable description method with the simplicity of fixed bit string lengths. In this paper, the maximum number of circuit components was set to ten. This was done to develop compact circuits and minimize GA runtimes.

This incremental approach to circuit description allows for many circuit topologies to be explored by the GA while greatly limiting the number of unsimulatable circuits. It provides a very fast and compact method for evaluating circuit configurations in an evolutionary environment.

7. AUTOMATED CONTROLLER SYNTHESIS USING LEAKY CAPACITOR MODEL

Using the variable topology GA representation and the integral squared error fitness measure, five GA runs were performed to synthesize control circuitry for the capacitor model.[2] Early attempts at synthesizing control circuitry were unsuccessful due to the high steady-state error in the response. To combat this, a proportional gain stage was added prior to the evolved controller. This stage reduces the steady state error and allows the evolved controller to focus on the shape of the response. The best fitness runs using the capacitor model are presented in Table 3, and the circuit diagram and step response for the best circuit are shown in Figures 8-9. The best controller for the capacitor model had a settling time of 10 μs allowing for closed-loop control up to 50 kHz. These results are not as good as those with the optimized PID controller, but they greatly improve on the closed-loop response without the evolved controllers.[3] Unfortunately, due to the inability of the capacitor model to take into account ringing in the actuators, these controllers would likely fail if driven beyond normal open-loop limits.

Table 3. Control statistics for optimized variable topology controllers using capacitor model

Run	Rise Time (μs)	Settling Time (μs)	Overshoot (%)	SS Error (%)	Fitness (ISE)
1	9.66	9.66	0	5.15	0.0682
2	10.4	56.7	16.4	10.5	0.103
3	1.85	9.71	22.1	8.07	0.0748

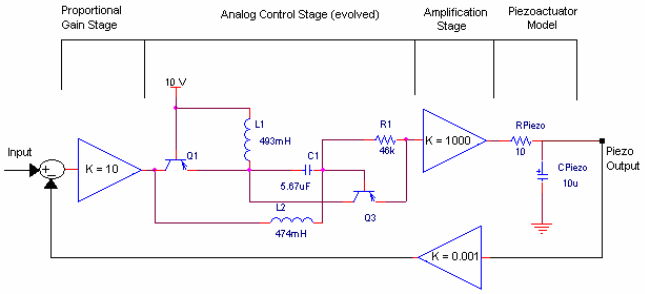

Figure 8. Circuit diagram for best evolved controller using capacitor model

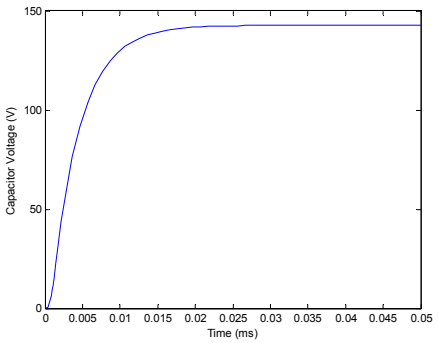

Figure 9. Step response of evolved capacitor model controller

8. AUTOMATED SYNTHESIS USING SECOND-ORDER MODEL

This section describes the evolution of controllers for use with the second-order model of the piezoelectric actuators. Since this model was developed based on the frequency characteristic of the FPA-900 actuators (see Section 3), the controllers developed for it have the potential to operate beyond the normal operating frequency of 100 Hz.

Since controlling the 2nd order transfer function is a more difficult task than controlling the charging of a capacitor, a new fitness function was introduced for this evolution. The weighted fitness function takes a weighted average of the rise time, overshoot, steady-state error, and integral squared error of the step

[2] Population sizes of 1000 were used, and the GA converged after about 10 hours of runtime and 50 generations.

[3] The proportional gain stage improved the settling time of the closed-loop capacitor circuit to 21.8 μs, and the evolved controller improved it to 10 μs.

response and then updates the weights depending on the statistics of the circuits in the population (see Section 4). As with the capacitor model, a proportional gain stage was added before the evolved controller. Due to the difficulty of the task, the gain was increased to 100 for these runs. Five GA runs were done with the weighted fitness function and five with the integral squared error fitness function.[4] The fittest circuits for each model are presented in Table 4, and Figures 10-11 show the circuit diagram and step response of the fittest circuit found using the ISE fitness function. The GA was successful at evolving controllers with settling times as low as 2.48 ms allowing for driving frequencies up to 202 Hz, twice that of the suggested open-loop driving frequency.[5]

Table 4. Control statistics for optimized variable topology controllers using second-order model

Fitness Method	Rise Time (ms)	Settling Time (ms)	Overshoot (%)	SS Error (%)	ISE
ISE	0.555	2.48	26.9	4.44	0.121
Weight	0.543	3.13	3.37	4.44	0.122

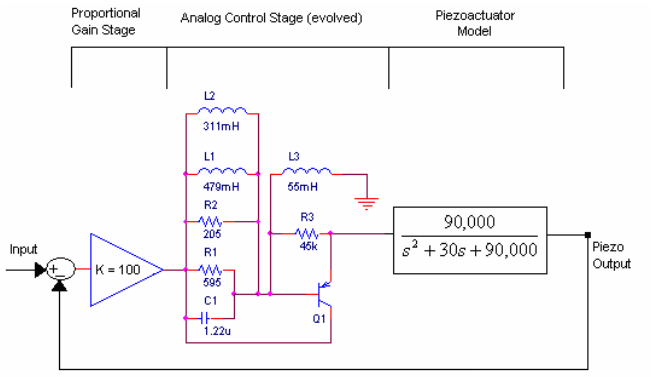

Figure 10. Circuit diagram for evolved controller using second-order model

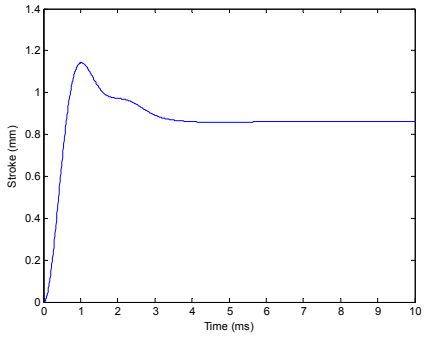

Figure 11. Step response for evolved controller using second-order model

To compare the two fitness functions, the average values of the overshoot, rise time, settling time, SS error, and integral squared error of the three best circuits were determined for each fitness function. As shown in Table 5, the two fitness functions performed comparably at the task of developing evolved controllers for the second-order model. For this reason, both fitness functions were used for the synthesis of fault-tolerant controllers in Section 9.

Table 5. Average control statistics over three runs for optimized variable topology controllers using second-order model

Fitness Method	Rise Time (ms)	Settling Time (ms)	Overshoot (%)	SS Error (%)	ISE
ISE	0.585	2.59	31.9	4.60	0.134
Weight	0.551	3.61	32.9	3.30	0.141

9. FAULT-TOLERANT CONTROLLER SYNTHESIS

This section describes the use of the GA to perform fault-tolerant design of controllers for the piezoelectric actuators. Two types of fault-tolerances were implemented; one for internal component failures and one for external plant failures. Component failure was built into the fitness function by simulating each circuit once with each circuit component removed. The fitness values for each of these runs were then averaged with the original fitness to determine the final fitness function of the circuit. Five GA runs were done with each fitness function, and the fittest two were selected.[6] Table 6 gives the control statistics of the evolved component fault-tolerant controllers, and Figures 12-13 show the circuit diagram and step response for the fittest controller evolved using the ISE fitness function. The statistics show that fault-tolerant controllers can perform comparably to those evolved without fault-tolerance in the fitness function. In this application, the ISE fitness function significantly outperformed the weighted fitness function.

To design controllers robust to both component failure and actuator failure, three more evaluations were added into the average fitness function. These evaluations corresponded to three modifications to the plant transfer function. In the first evaluation, the gain of the plant transfer function was reduced to half its original value. This corresponds to a failure in the amplifier prior to driving the piezoelectric actuator. Additionally, excessive heat can cause the ceramic on the actuators to depole, and humidity entering the actuators can lead to decreased force output. Both of these faults appear as a decrease in the gain of the transfer function. In the second evaluation, the natural frequency of the actuator is reduced to half its original value, and the third evaluation corresponds to a one-third decrease in the damping of the actuator. These two modifications to the transfer function could potentially occur if the spring mechanism in the actuator suffered a failure or the actuator frame fractured.

[4] Population sizes of 100 were used, and the GA converged on a solution after about 10 hours of runtime and 50 generations.

[5] The settling time of the controller using solely the proportional gain stage was 257 ms.

[6] Population sizes of 100 were used. The GA converged on a circuit after about 30 hours of runtime and 50 generations.

Table 6. Control statistics for variable topology controllers with component fault-tolerance[7]

Fitness Method	Rise Time (ms)	Settling Time (ms)	Overshoot (%)	SS Error (%)	ISE
ISE	0.611	1.93	37.1	5.11	0.148
	0.625	2.75	48.0	5.62	0.170
Weight	0.560	3.62	58.2	4.84	0.169
	0.575	7.28	61.2	5.48	0.281

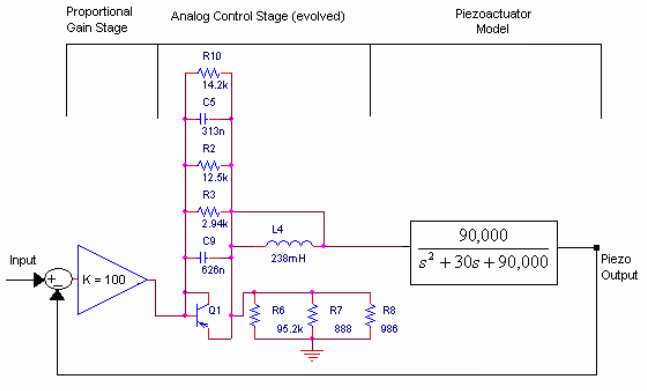

Figure 12. Circuit diagram for component fault-tolerant controller

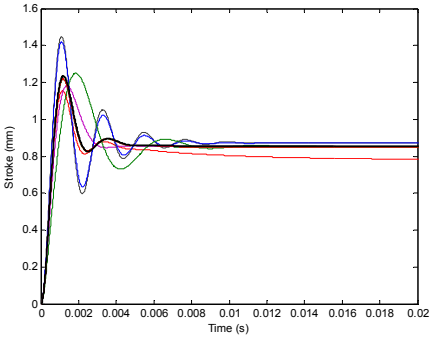

Figure 13. Step response for best evolved fault-tolerant controller (thick black curve shows the response without failure, and other curves show the response with a component removed)

Table 7. Control statistics for optimised variable topology controllers with component and plant fault-tolerance

Fitness Method	Rise Time (ms)	Settling Time (ms)	Overshoot (%)	SS Error (%)	ISE
ISE	0.591	1.87	45.4	6.56	0.155
	0.930	4.90	49.0	7.31	0.257
Weight	0.606	1.95	48.7	5.84	0.160
	0.927	6.71	54.3	7.11	0.305

[7] The first row for each fitness type in Tables VI-VII gives the control statistics without faults, and the second row gives averages over all potential faults.

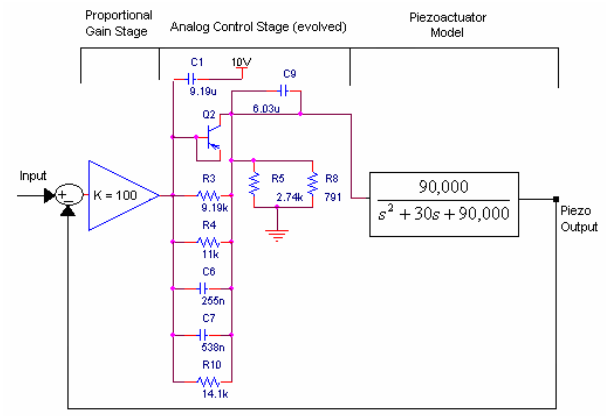

Figure 14. Circuit diagram for plant and component fault-tolerant controller

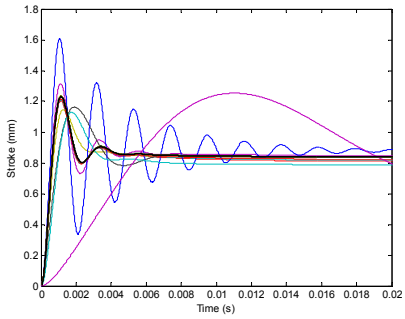

Figure 15. Step response for best evolved fault-tolerant controller (thick black curve shows the response without failure, and other curves show the response with each fault)

Five GA runs were done using each of the fitness functions, and the fittest results are presented in Table 7 and Figures 14-15. The results show that the controllers robust to plant failure do not perform as well as the other evolved controllers. However, the plant and component fault-tolerant controllers still provide adequate control of the actuator model for nearly all of the thirteen potential faults. All component failures in the PID controllers, on the other hand, lead to catastrophic failures.

Both the PID and evolved controllers consistently failed to control the system when the natural frequency was decreased. This critical actuator failure reduced the settling time by a factor of ten (see pink curve in Figure 16). The decreased natural frequency greatly changes the characteristics of the actuators and increases the difficulty of controlling the system. The controllers were unable to respond adequately to this change. The evolved controllers, however, provided better fault-tolerance (with respect to the settling time) for the reduced natural frequency cases and comparable fault-tolerance for the other two actuator faults. Figure 16 gives a comparison of the step responses with actuator faults for the PID controller and the evolved fault-tolerant controller. The graphs are plotted from zero to five times the faultless settling time to show that the evolved controller provides comparable or better actuator fault-tolerance with respect to the settling time. With the current results, the price paid for fault-tolerance is a compromise in the control system performance.

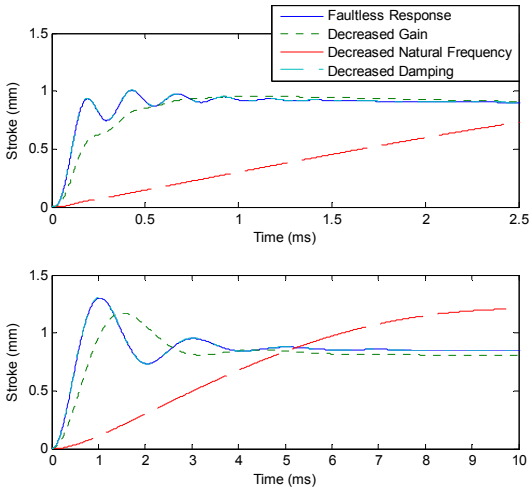

Figure 16. Comparison of step response during actuator faults for PID controller (top) and evolved controller (bottom). Solid curve shows faultless response, and dashed curves show decreased gain, decreased natural frequency, and decreased damping (nearly identical to faultless response) cases.

10. CONCLUSIONS AND FUTURE WORK

Through evolutionary design, fault-tolerant analog controllers were developed for piezoelectric actuators. These controllers were constructed with very few circuit components, small GA population sizes, and a minimum number of generations. Evolved controllers were developed to be robust to both circuit and actuator failures. While the evolved controllers did not perform as well as PID controllers, their fault-tolerant characteristics may be preferable to more traditional controllers in applications where fault-tolerance is mission critical.

The next step is to implement the controllers in hardware and to test their performance on the piezo robot during pipe inspection. Once the performance of these controllers has been verified, the techniques presented in this paper can be applied to other control tasks. Unlike human circuit design, the techniques in this paper may be scalable and generalizable to larger circuits. This claim should be tested in future work. Additional suggestions for further research include tailoring the circuit representation specifically to control tasks. For instance, a study should be made regarding common proportions and layouts of components in analog controllers, and the circuit representation should be biased to create these proportions. This would improve the number of useful circuits and further decrease processing time during GA runs. Finally, further study of fitness functions is suggested. The integral squared error and the weighted fitness functions both performed well at synthesizing controllers, and their strengths and weaknesses should be further studied. Ultimately, this line of research has potential to develop controllers with high levels of fault-tolerance, a crucial characteristic in aerospace applications.

11. ACKNOWLEDGMENTS

Thanks to team members Jeri Briscoe, Ken King and Frank Brannon at NASA Marshall Space Flight Center. Further thanks to Jeff Payne at Dynamic Structures and Materials, LLC.

12. REFERENCES

[1] D. Campolo, M Sitti, and R.S. Fearing, "Efficient Charge Recovery Method for Driving Piezoelectric Actuators with Quasi-Square Waves," *IEEE Trans. on Ultrasonics, Ferroelectrics, and Freq. Control*, vol. 50, no. 1, Jan. 2003.

[2] T. R. Dastidar, P. P. Chakrabarti, and P. Ray, "A Synthesis System for Analog Circuits Based on Evolutionary Search and Topological Reuse." *IEEE Trans. on Evolutionary Computation*, vol. 9, no. 2, April 2005.

[3] "FPA-900 Piezoelectric Actuator Datasheet," tech. memo, Dynamic Structures and Materials, LLC., 2003.

[4] GAUL (Genetic Algorithm Utility Library) Online Documentation, http://gaul.sourceforge.net/.

[5] G. Hollinger and J. Briscoe, "Genetic Optimization and Simulation of a Piezoelectric Pipe-Crawling Inspection Robot," *Proc. IEEE Intl. Conf. Robotics*, 2005.

[6] J. Hu and E. Goodman, "Topological Synthesis of Robust Dynamic Systems by Sustainable GP," In *Genetic Programming: Theory and Practice 2*. Springer, 2005.

[7] D. Keymeulen et al., "Fault-Tolerant Evolvable Hardware Using Field Programmable Transistor Arrays," *IEEE Trans. on Reliability*, vol. 49, no. 3, Sept. 2000.

[8] J.R. Koza et al., *Genetic Programming 4: Routine Human-Competitive Machine Intelligence*, Springer, 2003.

[9] C. Leger, *Darwin2K: An Evolutionary Approach to Automated Design for Robotics*," Kluwer, 2000.

[10] J.D. Lohn et al. "Towards Evolving Electronic Circuits for Autonomous Space Applications," *Proc. IEEE Aerospace Conf.*, 2000.

[11] J.D. Lohn and S.P. Colombano. "A Circuit Representation Technique for Automated Circuit Design," *IEEE Trans. on Evolutionary Computation*, vol. 3, no. 3, Sept. 1999.

[12] J.A. Main et al., "Efficient Power Amplifiers for Piezoelectric Applications," *Smart Mater. Struct.*, vol. 5, 1996, pp. 766-775.

[13] "Opamp PID Controller," http://www.ecircuitcenter.com/.

[14] The PI Polytec Group, "Open and Closed-Loop Operation of Piezo Actuators," http://www.physikinstrumente.com/.

[15] S. Salisbury et al., "Complementary inchworm actuator for high-force, high-precision applications," *IEEE/ASME Transactions on Mechatronics*, in press.

[16] T. Sripramong, C. Toumazou, "The Invention of CMOS Amplifiers using Genetic Programming and Current-Flow Analysis," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 21, no. 11, Nov. 2002.

[17] P. Wang and D. P. Kwok, "Auto-Tuning of Classical PID Controllers using an Advanced Genetic Algorithm," *Proc. Int'l Conf. Industrial Electronics, Control, and Instrumentation*, vol. 3, Nov. 1992.

[18] R.S. Zebulum et al., "Evolvable Hardware: On the Automatic Synthesis of Analog Control Systems," *Proc. IEEE Aerospace Conf.*, IEEE Press, 2000.