Evolving Analog Controllers for Correcting Thermoacoustic Instability in Real Hardware

Saranyan A Vigraham
Department of Computer Science and Engineering
Wright State University
Dayton, OH 45435
svigraha@cs.wright.edu

John C Gallagher
Department of Computer Science and Engineering
Wright State University
Dayton, OH 45435
jgallagh@cs.wright.edu

Sanjay K Boddhu
Department of Computer Science and Engineering
Wright State University
Dayton, OH 45435
sboddhu@cs.wright.edu

ABSTRACT

Previous research demonstrated that Evolvable Hardware (EH) techniques can be employed to suppress Thermoacoustic (TA) instability in a computer simulated combustion chamber. Though that work established basic feasibility, there were still significant questions concerning whether those techniques would function in the real world. This paper presents the results of the next incremental step between controlling in pure simulation and controlling a real combustion chamber. In it, we will examine issues involved with using EH methods to learn to control a hardware analog circuit model of a combustion chamber. In so doing, we establish that the basic methods work when interfaced to real hardware and uncover some interesting, potentially critical, differences between simulation and real environments. We will also establish that both the EA methods and the underlying reconfigurable hardware can be expected to learn effectively in noisy control environments and that they are well-suited for upcoming use in a live engine.

Categories and Subject Descriptors

B.m [**Hardware**]: Miscellaneous

General Terms

Algorithms, Design, Experimentation

Keywords

Evolvable Hardware

1. INTRODUCTION

Lean fuel mixtures (low fuel to air concentration) are considered desirable for fueling turbine jet engines because they allow for more complete combustion and emission of fewer pollutants. Unfortunately, the use of lean fuels can give rise to Thermoacoustic (TA) instability. TA instability can be characterized as the exponentially growing pressure oscillations within the combustion chamber leading to loss in fuel efficiency, component failure and even flame-outs [8]. Major traditional control approaches to TA instability problem either involve designing a controller through mathematical stability analysis [9] or using a by using optimization methods to tune generic feedback controllers and/or adaptive filters [5]. With the increasing complexity of the controlled system, these techniques become more difficult to apply. For instance, to design a mathematical controller, one is required to possess a large amount of domain knowledge and possess models of incredible fidelity. This condition becomes increasingly difficult to achieve as the system grows more complex. Tuned feedback systems have also proven problematic in that they have been observed to introduce additional resonance modes in the chamber [1].

In our previous work [6], we applied an Evolvable Hardware (EH) control technique for suppressing TA instability in a simulated combustion chamber. In those studies, our evolved controllers outperformed traditional controllers reported in the literature and were able to do so without introducing additional resonance modes. Though encouraging, these results still left answered the question of whether the EH controllers would function in the real world. Because of the expense and potential danger, it is not desirable to conduct experiments on a real combustion chamber without a strong evidence that the EH control techniques are capable of maintaining the pressure at safe levels when TA instability occurs. However, one may argue that it is impossible to know if the EH control techniques are suitable for such use unless they had been put to test. To help resolve this conflict, we moved the system from simulation to emulation. The EH controller was implemented in a desktop computer and interfaced to real plant through digital to analog converters (DACs) and analog to digital converters (ADCs). The combustion chamber was emulated by a direct analog computer built to model TA instability and feedback phenomena. This allowed us to test live EH hardware in a more real environment. It also allowed us to uncover potential difficulties before moving to a live flame combustion system.

In this work, we present one specific set of experiments in which TA instability is suppressed in an analog circuit

GECCO'05, June 25–29, 2005, Washington, DC, USA.

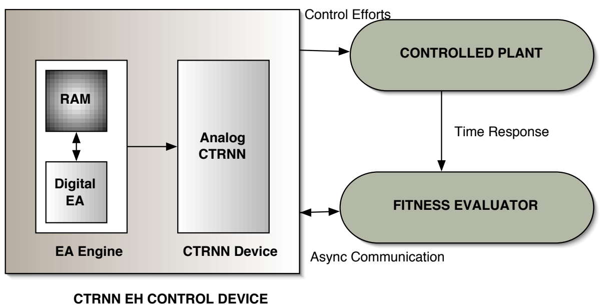

Figure 1: CTRNN-EH Device : Architecture

operating in real time. Subsequent sections of this paper will provide complete details on the chamber emulation circuitry as well as the EAs and reconfigurable control hardware. Because we eventually intend to combine the EA and the control hardware into a unified VLSI device for online use, special care was taken to make choices that will minimize the size of the chip without unduly sacrificing learning or control efficacy. These choices will also be discussed in detail. The paper will conclude with a discussion of results and their implications for future work.

2. EH CONTROL DEVICE : ARCHITECTURE

The EH control device employed in our work consists of an analog Continuous Time Recurrent Neural Network (CTRNN) and a digital EA engine used for configuring the CTRNN. This is illustrated in figure 1.

CTRNN

The CTRNN is a generalization of Hopfield's Neural Network [10]. When the restrictions of zero self connection weight and symmetricity are relaxed from Hopfield networks, they they take the form of CTRNNs. In short, CTRNNs are Hopfield networks with unconstrained weight matrices [2]. It has been demonstrated that CTRNNs possess rich dynamical properties and they are capable of approximating any smooth dynamics when provided with sufficient number of neurons [4]. The mathematical form of a CTRNN is given by

$$\tau dy_i/dt = -y_i + \sum_{j=1}^{N} w_{ji}\sigma(y_i + \theta_i)$$

where y is the state of each neuron, τ is its time constant, w_{ji} is the strength of the connection from j^{th} neuron to ith neuron, θ is a bias term and $\sigma(x) = 1/(1 + e^{-x})$ is the standard logistic activation function. The state, weights, and bias are the parameters of a neuron. The number of parameters in a CTRNN is determined by the size of the network, or the number of neurons present. For instance, a single neuron network has 4 parameters (1 weight, 1 time-constant, 1 bias and 1 external sensory input weight), and a five neuron CTRNN has 40 parameters (25 weights, 5 external sensory input weights, 5 time-constants and 5 biases).

EA Engine

The EA engine is an equally important component of the CTRNN-EH control device. The EA engine is comprised of an evolutionary algorithm and memory to store necessary information like best configurations of CTRNNs, fitness scores and other evolution related details. As mentioned previously, it is desirable that the EA engine can be implemented using digital VLSI techniques without consuming significant area and power. Because the CTRNN is of analog nature, it can be implemented using analog VLSI techniques within the constraints of low power and area. The literature shows some VLSI implementations of CTRNNs [3, 12]. The nature of the EA makes digital VLSI techniques more appropriate than the analog ones. However, most digital VLSI EAs are known for their faster operational speed than for compact and low power implementations. So, it becomes essential for one to use a space saving EA to minimize the area consumption while maintaining a good search efficacy. The latter becomes more important when the environment is noisy. In this work, we explore one such EA capable of dealing with noisy environments effectively and which can also be implemented in hardware without consuming too much area and power. The algorithm will be introduced in the next subsection.

Minipop : EA component of the EA engine

Among the different EAs existing in the literature, Minipop [13] algorithm is chosen for evolving CTRNN controllers. Minipop algorithm is a hardware feasible EA that derives its inspiration from the Micro-GA [11] by maintaining a small population. The small population results in significant space savings. Figure 2 illustrates the standard Minipop algorithm.

The search mechanism of the minipop algorithm is propelled by mutation and hypermutation. When each evaluation completes, the best four members of the evaluation form the population for the next evaluation. There is a hypermutation tournament introduced in each evaluation to navigate the algorithm towards the best possible solution in the entire search space. If the fitness landscape has large areas of flat plateaus and the algorithm gets caught in one such plateu, mutation may not be sufficient to steer the algorithm out. This is overcome by the hypermutation tournament.

Initially, population size is fixed at N (we used N=4). Using a population size of 4 makes a compact hardware implementation possible without losing the algorithm efficacy. This is demonstrated in a later section. The initial population is generated at random (lines 2-5). Fitness scores of all the members of the population is evaluated and stored. Later, all the members of the population are mutated individually and their fitness scores are evaluated (lines 9-10). If the mutant's fitness is better than its parent's fitness, it replaces the parent in the population for the next evaluation (lines 11-14). After this step, a hypermutation tournament is conducted. A completely random individual is generated and evaluated against the worst member in the population. If this hypermutant is better than the worst member in the current population, it replaces the worst member (lines 16-23). The best individual of the final four is returned as the best solution yet seen. The process repeats until the number of evaluations reaches the maximum value.

```
1. Start
    max_evaluations = MAX;
    population_size = N;
2. for i = 1 to N do
3.   Generate RANDOM bitstring for
      pop[i];
4.   Evaluate Fitness for pop[i] and
      store in fitness[i];
5. done
6. while evaluations <= MAX  do
7.  evaluations := evaluations + 1
8.    for i = 1 to N do
9.      mutate pop[i] and store in
         mut[i];
10.     evaluate mut[i] and store in
          mfitness[i];
11.      if mfitness[i] > fitness[i] then
12.        pop[i] = mut[i];
13.        fitness[i] = mfitness[i];
14.       endif
15.  done //for loop ends
16.  generate a completely RANDOM
      bitstring hyper_mutant;
17.  evaluate the fitness of hyper_mutant
      and store in f_hypermutant;
18.  Determine the worst member of
     population and store in worst_member;
19.  Evaluate the score of worst_member
    and store in f_worst_member;
20.  if f_hypermutant > f_worst_member
21.    pop[index_worst] =
                 hyper_mutant;
22.    fitness[index_worst] =
               f_hyper_mutant;
23.   endif
24.  Determine the best member of
      the population;
25.    return pop[best_member_index]
24. done //while loop ends
```

Figure 2: The Minipop algorithm

3. TA INSTABILITY

TA instability can be understood as the phenomenon of exponential pressure growth inside a combustion chamber due to the positive coupling between pressure and heat release rate. Figure 3 is an analog circuit exhibiting TA instability. This has been obtained from [7].

Figure 4 shows the time response of this analog circuit when simulated using MATLAB. The x-axis has units of time-steps and the y-axis has units of volts. In the real combustion chamber, these voltage values correspond to pressure values in Pascal. Each time-step on the x-axis corresponds to 0.5 milliseconds in real time. As it can be seen from the figure, the pressure (voltage) exponentially increases within a very short duration of time. Here, the pressure (voltage) has been clipped after a certain amplitude (0.6 V). This has been done to ensure that the circuit components are kept within safe operational voltage levels. Diodes D1 and D2 in figure 3 help one achieve this. The MATLAB simulation has been presented to indicate the ideal circuit behavior. The same circuit, when operating in real time, has

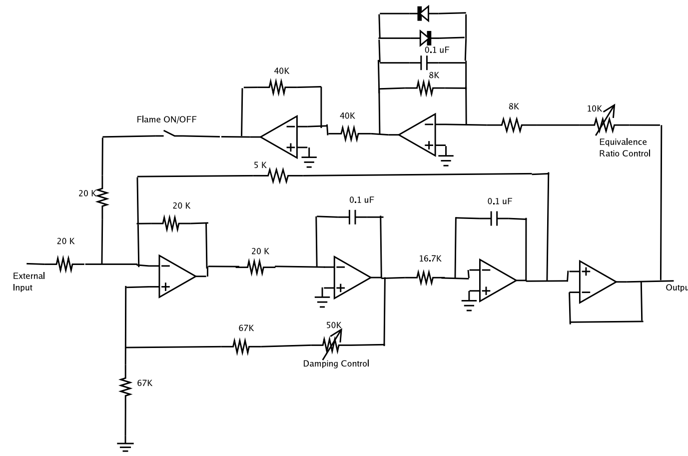

Figure 3: Analog Circuit Exhibiting TA Instability

noisy output as indicated by the live data recording shown in figure 5.

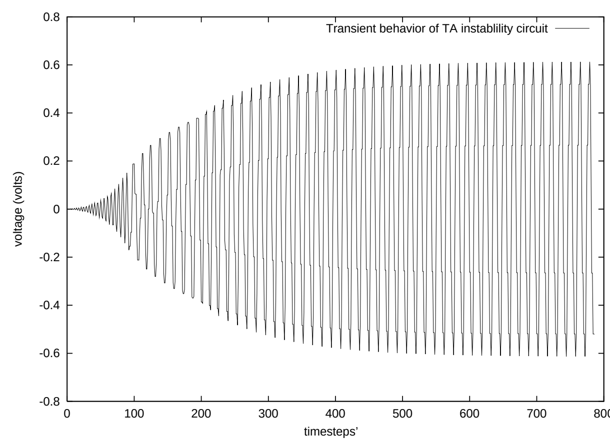

Figure 4: Ideal circuit behavior simulated from Matlab

Figure 6 shows the combined responses of the ideal and the real cases. We can observe from the figure that, there is a 66% increase in the observed amplitude of the oscillations, due to noise. This is a significant change and indicates a high amount to noise. Hence, in addition to controlling the TA instability, it also becomes essential to deal with the noise present in the system.

4. EXPERIMENTAL SETUP

One popular approach for controlling TA instability in real world is to mount a loudspeaker on the combustion chamber [14] and play sounds on the speaker so that the oscillations inside the chamber are dampened. Normally, a filter or a phase shifter is connected in feedback to the loudspeaker integrated combustion chamber. The parameters

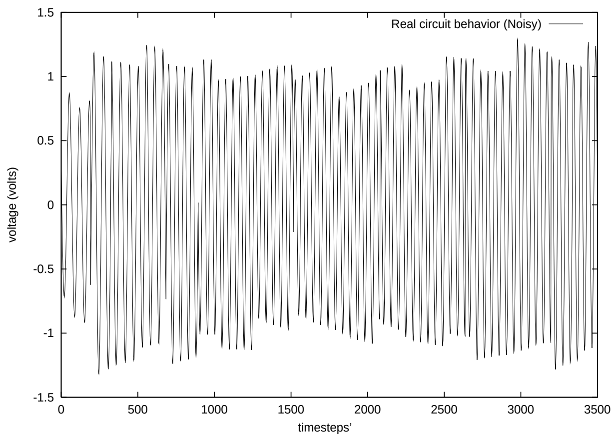

Figure 5: Real circuit behavior recorded from live data

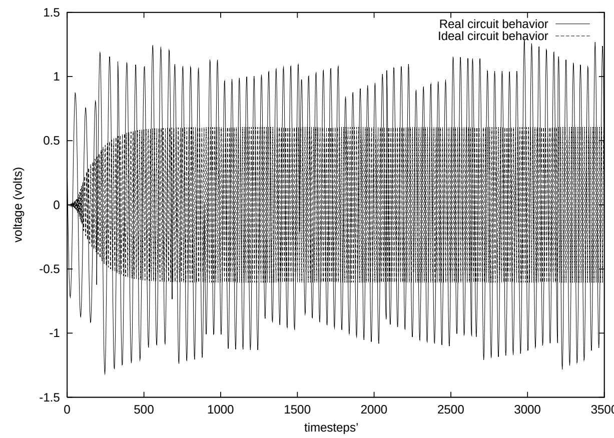

Figure 6: Indication of amount of noise present in the system

of this feedback element are tuned. The feedback element drives a Voltage Controlled Oscillator (VCO) whose output drives the loudspeaker. In our previous work [6], we replaced the traditional feedback element with a CTRNN control device. The rest of the setup remained same. Two neurons of a 5 neuron CTRNN output the amplitude and frequency values for the loudspeaker through the VCO. Figure 9 shows this setup.

In this work, we followed this basic setup and replaced the computer simulated combustion chamber with its analog equivalent model. The VCO was also implemented in analog components. The VCO circuit was designed using SN74L629 IC. Figure 7 illustrates the design. The pins of interest and their connections have been shown in the figure. The output frequency range for the VCO was between 20 and 280 Hz. A voltage follower was connected at the output of the VCO to avoid loading effects. After this stage, the gain and switching circuitry stages were implemented. The VCO output positive voltage values. Because the voltage to be suppressed from the TA instability circuit swung across both positive and negative stages, we biased the VCO output with a simple bias circuit and amplified the output after the bias stage with a variable gain amplifier. The complete bias and amplification circuitry constituted our gain circuit. This amplitude adjustment circuit is shown in the figure 8.

The next stage was the switching circuit implemented with a SPST switch CD4016BCN. This circuit was implemented to switch the engine between stable and unstable configurations. Testing the analog engine model in both stable and unstable modes is very essential for deciding upon the quality of the controllers. It is equally important for a controller keep a stable engine stable as much as it is for suppressing the instability in an unstable engine. Hence, we considered both stable and unstable conditions of the engine when evolving controllers. This will be discussed in detail in the next section.

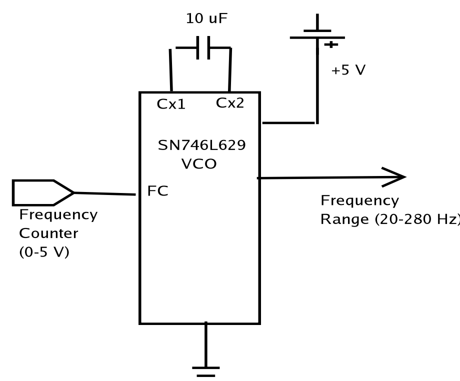

Figure 7: VCO implementation

The EA engine (minipop) was running on a dedicated workstation. The CTRNN controllers were also evolved on this workstation. The CTRNN received inputs from the analog engine model through a National Instruments Data Acquisition Card (DAC) SCB-68, connected to this workstation. The neuron outputs for setting the amplitude and frequency of the control signal are routed to the VCO via the DAC. The complete experimental setup is illustrated in figure 9.

4.1 CTRNN : Experimental Details

In this work, we used an eight neuron CTRNN for controlling TA instability in the analog engine model. In the referred work by Gallagher and Vigraham [6], a five neuron CTRNN had been used for suppressing TA instability in a simulated combustion chamber. Our early experiments suggested that a 5 neuron CTRNN was not sufficient for suppressing TA instability in this analog circuit. While the model employed in [6] seems to have more complexity as indicated by the state equations on which the combustion

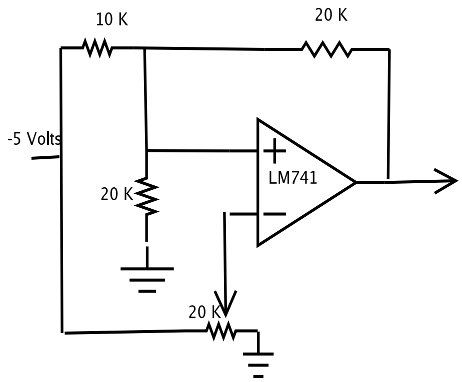

Figure 8: Amplitude Adjustment Circuit

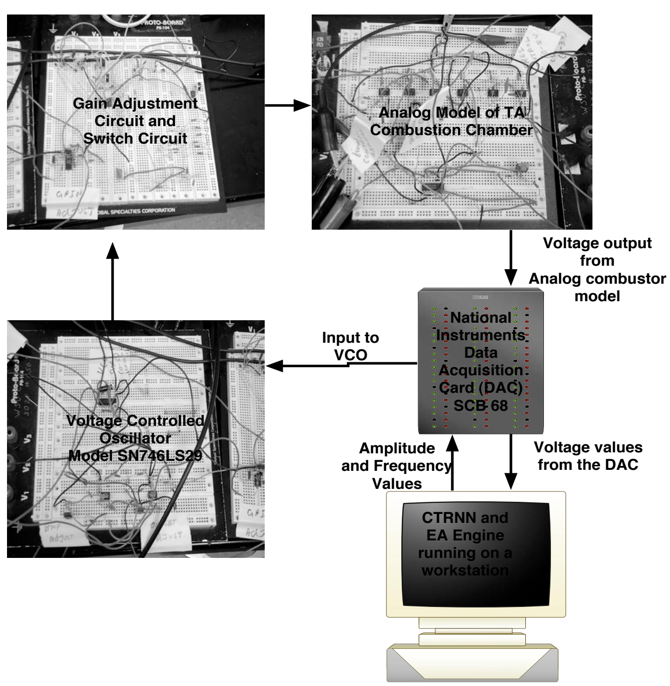

Figure 9: The complete system

chamber was simulated, a five neuron CTRNN was sufficient to suppress TA instability. In this work, we observed that a five neuron CTRNN was not sufficient to control TA instability on real analog hardware. This suggests that the real world problems have more degree of complexity which may not be straightforward to the naked eye. The noise factor discussed in the previous section may be a factor attributing to the increased complexity of this problem. There may be other factors which, at this time, is under investigation.

Each neuron has the parameters described in section 2. In total a 8 neuron CTRNN has 88 parameters. Each parameter is encoded into 8 bits. This determines the genome size of the EA.

4.2 Details about the EA engine : minipop

Weighted Resampling for handling noisy environments

In [13], it was suggested that minipop algorithm worked better in noisy environments when weighted resampling was performed. Resampling is an operation by which a candidate solution is periodically re-evaluated. Such periodic re-evaluation prevents a solution from holding a misleading score . Weighted resampling modifies the resampling operation such that the fitness assigned to an individual is a function of its previous fitness and its resampled fitness. Let F_{old} is the fitness of the individual and F_{new} is the new fitness value returned when the individual is reevaluated. Assign W as the resampling weight operator. The value of W ranges from 0.0 to 1.0. Then the calculation of resampled fitness can be given by the following equation.

$$F_{resampled} = (1 - W) * F_{old} + W * F_{new}$$

The modifications to the minipop algorithm after weighted resampling has been incorporated, can be found from [13]. In this work, we used a value of 0.3 for W.

Parameters of Minipop

For this work, we used a minipop of a population size 4. The genome length was set to 704 to encode all the 88 parameters of the CTRNN. The bitwise mutation rate used was 0.01. The seed for the random number generator was the system clock and the maximum evaluations were set to be 10,000. The run time for one experiment to finish was about 720 minutes. We evolved a total of 25 controllers. It took 13 days to finish evolving all the controllers. The time taken to evolve the controllers can be attributed to the way the fitness of a controller is evaluated. One might argue that if the eventual goal is to suppress TA instability in a real combustion chamber, the time to find a controller is not acceptable. We address this issue in the analysis section of this paper. Till the analysis has been presented, it might be taken with confidence that the time to find a best controller is not a bottleneck as it appears to be.

Fitness Function

For this work, the fitness function for the EA was the area under the curve of the voltage time response curve. Whenever the fitness computation of a genome was requested by the EA, the analog engine was initialized to a known unstable state. The CTRNN controller connected in feedback with the engine was configured to the values represented by the genome. The engine was operated with this setup and the output response was recorded. After 1 sec, the switch in the circuit was flipped to make the engine go to a stable state without disturbing the CTRNN controller. The output response was recorded for 1 sec. The fitness score returned to the EA was the complete area under the curve of the output response of the analog engine for the duration of this 2 seconds. The EA's optimization goal is to find a controller that has the minimum area under the curve. Additionally, if a controller succeeds in making an uncontrolled engine stable but introduces instability in an already stable engine, it is penalized and has a poor fitness score. The advantages of having such a fitness function are :

1. There is a higher chance that controllers evolved are robust i.e., they can recognize the change in the controlled environment and react to it accordingly.
2. If the engine does not encounter TA instability, then it can be assured that the controller will not introduce new instabilities into the engine

5. RESULTS AND ANALYSIS

5.1 Results of the experiments

In total, we evolved 25 controllers using minipop as the EA engine of the CTRNN-EH device. We found that all the 25 controllers were successful in suppressing TA instability in the analog engine circuit, giving a yield of 100 percent. The table 1 summarizes all the interested metrics and their associated values.

Metric	Result
No. of controllers evolved	25
No. of working controllers	25
Evaluations taken to find the first best solution	1325
Final fitness score	7123
Final standard deviation	2030

Table 1: Summary of results : minipop

Of particular interest are the last three metrics listed in the table. Before analyzing these statistics, we will define what an acceptable controller is.

Acceptable Controller

A controller is termed acceptable if it succeeds in suppressing TA instability. We can observe from the figure 4 that the amplitude of the output voltage exponentially grew to a maximum level at which it was held constant for its entire operational time. The maximum voltage at which the circuit oscillated was observed to be 2V. A controller is acceptable when it :

1. It prevents the exponential growth of oscillations.
2. It keeps the oscillations to a minimum.

Because of the nature of the analog engine system, it is impossible to keep the oscillations to zero. Figure 6 contains a matlab simulation demonstrating this statement. Here, the switch in the circuit 3 was opened to remove the positive feedback causing TA instability. In the real world, this is analogous to "somehow" remove the vicious feedback effect caused by the flame dynamics of the chamber. This can only be achieved by removing the flame altogether from the combustion chamber and hence, this method of oscillation elimination is impossible to achieve. However, it provides an excellent benchmark to test the controllers evolved. The amplitude of the minimum oscillations were observed 0.18 V. Because this was an ideal case scenario, the stable amplitude of the system was relaxed to 0.25 V. The area under the curve when the engine was constantly oscillating at this voltage, was calculated to be 8300 units. Any controller having an area under the curve of 8300 or less was categorized to be an acceptable controller. While one might argue that having an error score as the criteria for choosing a controller is misleading, it augurs well to recollect the constraints enforced on the fitness calculations from the previous section. Because, the controllers were penalized whenever their amplitude increased over the simulation time, there is a lesser probability of a controller being incorrectly judged. Additionally, all the 25 controllers were tested for erroneous behavior and none of them responded positively for this test.

Time taken to find the first good solution

This metric is very important for online intrinsic evaluations. When evolving controllers online, it is essential to find solutions faster. For this problem, as mentioned previously, a solution should have had an error score of 8300 or less to be classified as good. In our experiments, we found that, the minipop took on average of 1325 evaluations to find the solution. For the time being, we will consider this result in the literal sense without getting into the details of considering the time taken to perform one evaluation. In the later parts of the section, we will analyze this in detail.

Final fitness score

The average of the optimum fitness scores of all the 25 controllers was computed to be 7123. This suggested that the controllers performed better than the threshold set by us. The fig 10 indicates the trend of the average fitness curve against the number of evaluations. We can see from this curve that the threshold for acceptability is reached quickly and then a local search for a better controller configuration is being performed.

The fig 11 shows the response of the engine while being controlled by the best CTRNN controller. The x-axis is in time-step units instead of real time. The complete simulation window consists of 20,000 time-steps and is equivalent to 2 seconds in real time. The first 10,000 steps or 1 sec is the behavior of the unstable engine under the influence of the CTRNN controller. The second 10,000 steps indicates the period of 1 sec after the switch is flipped to configure the engine circuit to a inherently stable configuration. As seen from the figure, the controller managed to prevent exponential growth of the oscillations and also managed to

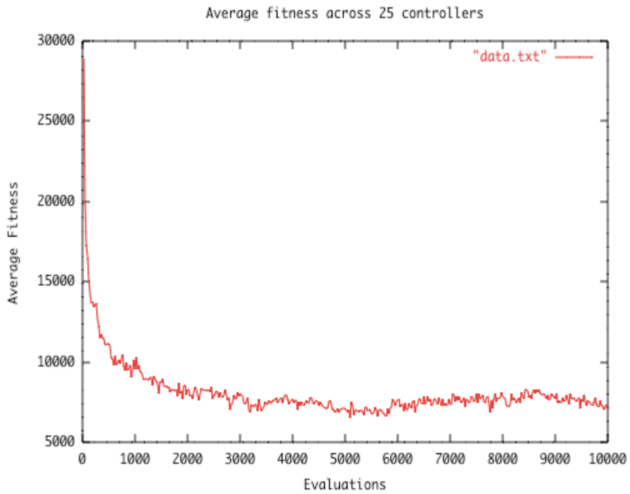

Figure 10: Average Fitness Trace

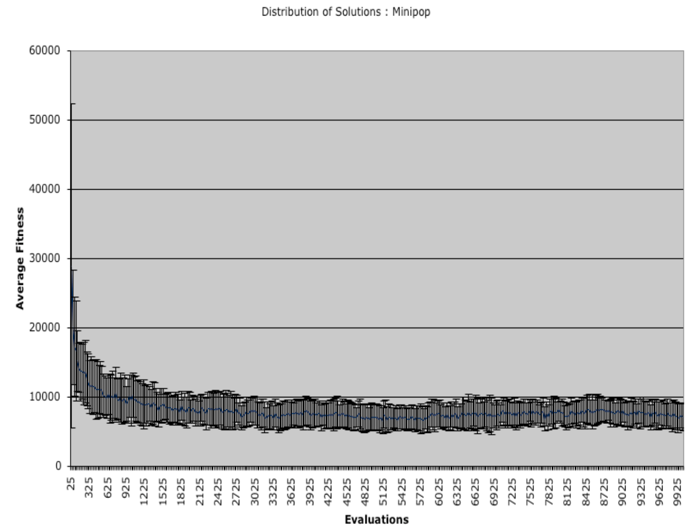

Figure 12: Spread of quality of evolved solutions

effectively keep the amplitude close to the ideal value. All the controllers exhibited this characteristic behavior.

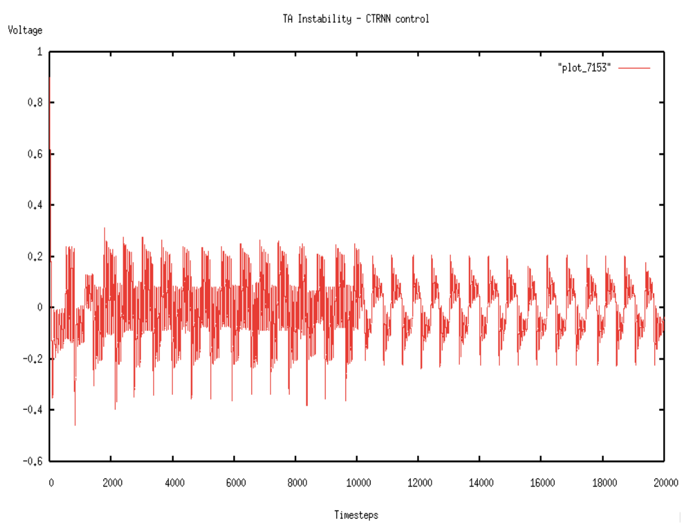

Figure 11: Engine response with CTRNN controller in action

Spread of the solution quality

The third metric of interest from the table 1 is the average standard deviation. It provides information about the clustering of the solutions at an evaluation. It is essential that the solutions are tightly clustered to determine with confidence that the the EA evolves solutions of similar quality every-time. While the standard deviation of the fitness seems to indicate a higher and not-so-attractive value, it is not really the case. It has to be remembered that the fitness score is computed by the area under the curve for the complete simulation time of 2 seconds (or 20,000 steps). If the amplitude of oscillations observed using a controller, say 1, is 0.1V larger than that obtained by using a controller 2, it significantly changes the area under the curve. Hence, the value 2030 for average standard deviation is not large, as it appears to be. The figure 12 shows the spread of solutions along the fitness curve. It can be observed that the solutions become tightly clustered with the increasing number of evaluations.

5.2 Analysis of the observed results

In this section of the paper, we present our detailed interpretation of the results observed. The time taken to find a first acceptable controller is very significant for intrinsic EH applications. Intrinsic EH, where the evolution is made online (in real time), requires the EA to evolve solutions quickly. We found that, it takes 1325 evaluations on average to find the first good solution. Each evaluation takes a minimum of 2 seconds to gather the information required to compute the fitness score. Even if other computational delays are completely ignored, the time taken to perform a fitness evaluation cannot be ignored. Given that the oscillations inside the combustion chamber can get to dangerous levels in the order of milliseconds, one can argue about the benefits gained by such an approach. However, it must be noted that, the bottleneck is just the fitness function and not the control approach. For intrinsic applications, the current fitness function must be changed. Two possible ways of implementing a new fitness function are :

1. Reduce the simulation window size. If the engine goes unstable in the order of milliseconds, the simulation window can be shrunk to monitor the engine behavior for certain millisecond duration rather than a complete second. In the experiments, because we do not have these constraints and also because these experiments were done for checking the feasibility of EH techniques for real time control applications, we did not feel the necessity to make this change.

2. Instead of calculating the area under the curve, the raw

pressure value can be used as the fitness score directly. This requires a controller with a fast response time. CTRNNs have been previously demonstrated to have a fast response time [6].

Other approaches may also exist in addition to the ones described above.

The controlled environment was very noisy. There were power supply and ground rail noises in addition to other environmental noises in the controlled environment. However, minipop algorithm evolved solutions of good quality despite the noise in the environment. We did not evolve controllers using the standard minipop and are hence, not in a position to predict the quality of solutions if weighted resampling was removed. But, from the results published in [13], it appears that there will be a significant difference in the quality of evolved solutions when weighted resampling is removed from the algorithm. Also, all the 25 controllers evolved were qualitatively very similar. This indicates the consistency of the algorithm to find good quality solutions.

6. CONCLUSIONS AND FUTURE WORK

In a previous research, we had demonstrated that EH techniques are better than traditional control techniques for suppressing TA instability in real time [6]. It was shown with evidence that controllers evolved using EH techniques were robust and superior in quality than the traditional controllers [8, 9] for a simulated combustion chamber. At that time, questions were raised over the feasibility of these techniques when experiments were conducted on real time instead of simulation. It was speculated that the EH control technique would be effective in real time as well. However, there were no experimental evidences at that time to support that speculation. In this work, we achieved the following goals :

1. It was demonstrated that the EH control technique is feasible for real time applications.

2. Controllers were evolved for TA instability on a real hardware. The controllers were successful in preventing TA instability and kept the oscillations to a minimum level.

3. A hardware feasible algorithm minipop was introduced and was used as the EA engine to evolve the CTRNN controllers. The minipop algorithm evolved very good controllers despite the noise in the environment.

At this time, we have sufficient results to support our plans for taking EH control of TA Instability to a next stage. A VLSI CTRNN-EH device is going to be built with minipop as the EA engine. This chip is intended to be made generic and hence, will have additional uses apart from suppressing TA instability in real turbine jet engines.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. 0238200.

7. REFERENCES

[1] A. Annaswamy, M. Fleifil, and A. Ghoniem. The origin of secondary peaks with active control of thermoacoustic instability. In *AIAA Joint Propulsion Conference*, 1997.

[2] R. Beer. On the dynamics of small continuous-time recurrent neural networks. *Adaptive Behavior*, 4:469–509, 1995.

[3] B. Brown. An analog vlsi implementation of a continuous time recurrent neural network. Master's thesis, Case western reserve University, 2000.

[4] K. Funahashi and Y. Nakamura. Approximation of dynamical systems by continuous time recurrent neural networks. *Neural networks*, 6:801–806, 1993.

[5] Y. Fung and V. Yang. Active control of nonlinear pressure oscillations in combustion chambers. *Journal of Propulsion and Power*, 1992.

[6] J. C. Gallagher and S. Vigraham. Active control of thermoacoustic instability in a model combustor with neuromorphic evolvable hardware. In *Genetic and Evolutionary Computation Conference*, 2003.

[7] A. Greenwood. Implementation of adaptive filter algorithms for the suppression of thermoacoustic instabilities. Master's thesis, Polytechnic Institute and State University, 2003.

[8] J. Hathout, A. Annaswamy, and A. Ghoniem. Model-based active control design for thermoacoustic instability. *Combustion Science and Technology*, 132:99–138, 1998.

[9] J. Hathout, M. Fleifil, J. Rumsey, A. Annaswamy, and A. Ghoniem. Model-based analysis and design of active control of thermoacoustic instability. In *IEEE Conference on Control Applications*, 1997.

[10] J. J. Hopfiled. Neurons with graded response properties have collective computational properties like those of two-state neurons. In *Proceedings of the National Academy of Sciences*, 1984.

[11] K. Kalamanje. Micro-genetic algorithms for stationary and non-stationary function optimization. In *SPIE Proceedings : Intelligent Control and Adaptive Systems*, 1989.

[12] R. Kier. Reconfigurable pattern generators in analog vlsi. Master's thesis, University of Utah, 2004.

[13] G. R. Kramer and J. C. Gallagher. A mini-population genetic algorithm for intrinsic evolvable hardware control applications in noisy environments. *IEEE Transactions of Evolutionary Computation*, Submitted.

[14] B. Zinn and Y. Neumeier. An overview of active control of combustion instabilities. In *AIAA*, 1997.