Feedback Controller Design Over the Internet of Things

THESIS

Presented in Partial Fulfillment of the Requirements for the Degree Master of Science in the Graduate School of The Ohio State University

By

John J Zakelj

Graduate Program in Electrical and Computer Engineering

The Ohio State University

2015

Master's Examination Committee:

Kevin M Passino, Advisor

Wei Zhang

Abstract

Utilizing non-gradient optimization methods on light control systems over a distributed network, experiments were implemented on many feedback control systems in an effort to show convergence and robustness of possible applications for the Internet of Things. Two non-gradient optimization methods are used to select gains for the feedback control systems. Results show that *all* feedback control systems in the network had desirable responses, which shows a type of robustness. These non-gradient optimization methods will allow control engineers to retrieve robust control designs without the need to model the system. It is hopefull that the distributed non-gradient optimization methods discussed in this paper will have many potential applications on the Internet of Things.

Vita

May 2008 .....Saint Ignatius High School

May 2013 .....B.S. Electrical and Computer Engineering, The Ohio State University

May 2015 .....M.S. Electrical and Computer Engineering, The Ohio State University

2010 to 2015 .....Graduate Teaching Associate, Department of Engineering, The Ohio State University

Fields of Study

Major Field: Electrical and Computer Engineering

Table of Contents

List of Figures

Introduction

The “internet of things" (IoT) is the interaction between devices to sense the environment, interpret the information, and react to real-world events autonomously with or without human intervention over a network [1]. It promotes the idea that an increasing number of devices are being connected to the internet which allows more efficient monitoring and control of these devices [2]. Furthermore, smart systems, like the smart lights studied here, are some of the building blocks for the IoT [3]. With mobile TCP/IP communication in hand, mobile devices such as those in control systems of automobiles, aircraft, and trains can be connected to the IoT to improve various aspects of travel [4]. Automobiles can be connected over the IoT to receive updates to controllers such as ones for climate control or cruise control for optimal performance over all connected vehicles. Also, home appliances, such as refrigerators, ovens, and washing machines, will be able to communicate with each other to reach optimum performance in each home connected to the IoT. There is a vast list of possible applications which will benefit from the IoT, and, with the growing number of devices connected, a distributed optimization method can be used to create and implement control systems for these devices. It is hoped that the non-gradient algorithms tested here can be adapted to control many feedback control systems to produce robust controller designs on large sets of control systems to benefit the IoT for a whole range of applications.

Distributing control systems over a network allows for them to be monitored, accessed, and changed in real-time. In this article, a distributed non-gradient optimization algorithm is operated over a network to tune a large set of feedback controllers in a relatively large scale laboratory experiment of 40 lighting control systems. It is shown that a “robust controller" emerges, one that performs well in all of the feedback control systems. This experimental approach to designing a controller removes the difficulties in modeling of complicated non-linear systems, and, thus, permits minimal system understanding to obtain a good solution. Essentially, the information that is exploited in design is from a large set of operating control systems.

Two standard non-gradient optimization methods, the genetic algorithm (GA) and the set-based stochastic (SBS) optimization method, are tested in the experiment. Based on Darwin’s natural selection theory, the GA is a non-gradient stochastic search method which allows multiple simultaneous search points to reach a global optimum point. The SBS optimization method creates a set of search points centered about the best point of the previous generation to find the global optimum point [5]. Although doubts about performance assurance surfaced in the past, these algorithms have now reached a stage of maturity to solve complex and conflicting problems which used to be considered “deadlocked" [6]. The GA is now being successfully implemented in a variety of engineering applications from the classic inverted pendulum problem to complicated VLSI circuit design problems [7, 8]. Both of these algorithms have been shown to be appropriate methods for feedback control system design, resulting in stable optimal designs [9].

Smart lighting solutions have been an important topic in energy conservation with the potential of reducing energy costs by 50% in existing buildings [10]. Smart lighting has been successfully tested to reach desired light values through an illumination balancing algorithm (IBA) while being influenced by cross-illumination effects and ambient light sources [11]. Furthermore, sensory nodes communicate illumination levels to the controller through a network to provide real-time changes over a large set of lighting systems. Due to the importance, experimental simplicity, and fast response times, smart light experiments provide an ideal system to test the use of distributed optimization algorithms for controller design over a network.

Experiment: Distributed Networked Light Control Systems

A proportional-integral (PI) controller sets the LED voltage, $V_{in} = u(t)$, in order to achieve the desired sensor voltage, $V_{out} = y(t)$, in Figure 1.

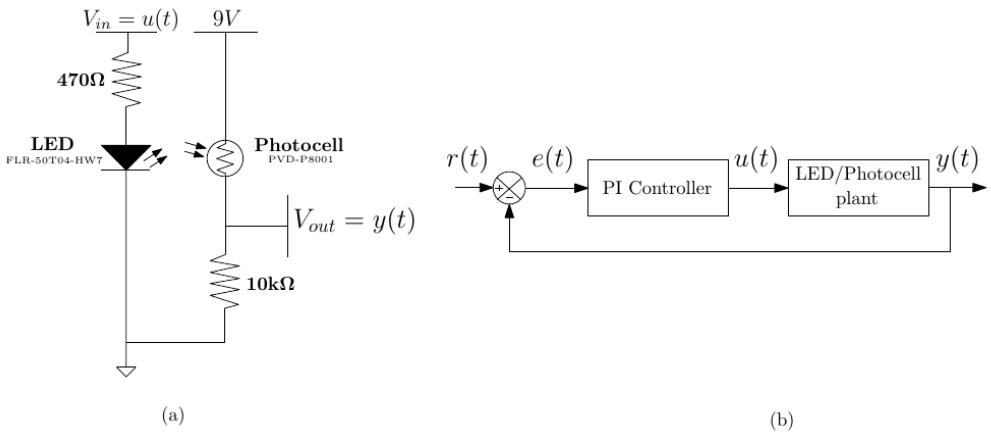

Figure 1: (a) Light sensor circuit design of feedback control system (right side) and light driver (left side). The voltage V_{in} is applied to the LED by the controller and V_{out} is the voltage from the sensor that is proportional to the light level. (b) Feedback control of light level.

Unlike the IBA experiments [11], cross-illumination effects and ambient light sources are minimized by barriers between each light source. Each light controller, $j = 1,2,\ldots,S$, has two gains, K_P^j and K_I^j, which are used to pick $u^j(t)$ via

$$u^j(t) = K_P^j e^j(t) + K_I^j \int_0^t e^j(\tau) d\tau \quad (1)$$

where

$$e^j(t) = r - y^j(t), \tag{2}$$

where e^j is the error input to the controller, r is the reference input, and y^j is the output of the plant.

In order to effectively use the distributed optimization algorithm to improve the feedback gains, a large population of control systems are necessary for the best results. Therefore, a control network was created with eight light control systems on each of five computers to achieve $S = 40$ controllers reporting their performance calculations to the central computer as shown in Figure 2. Using Matlab, the central computer runs the optimization algorithm based on the performance of each individual and returns a new set of gains to the population of feedback controllers through real-time Simulink models using transmission control protocol/internet protocol (TCP/IP) communication methods.

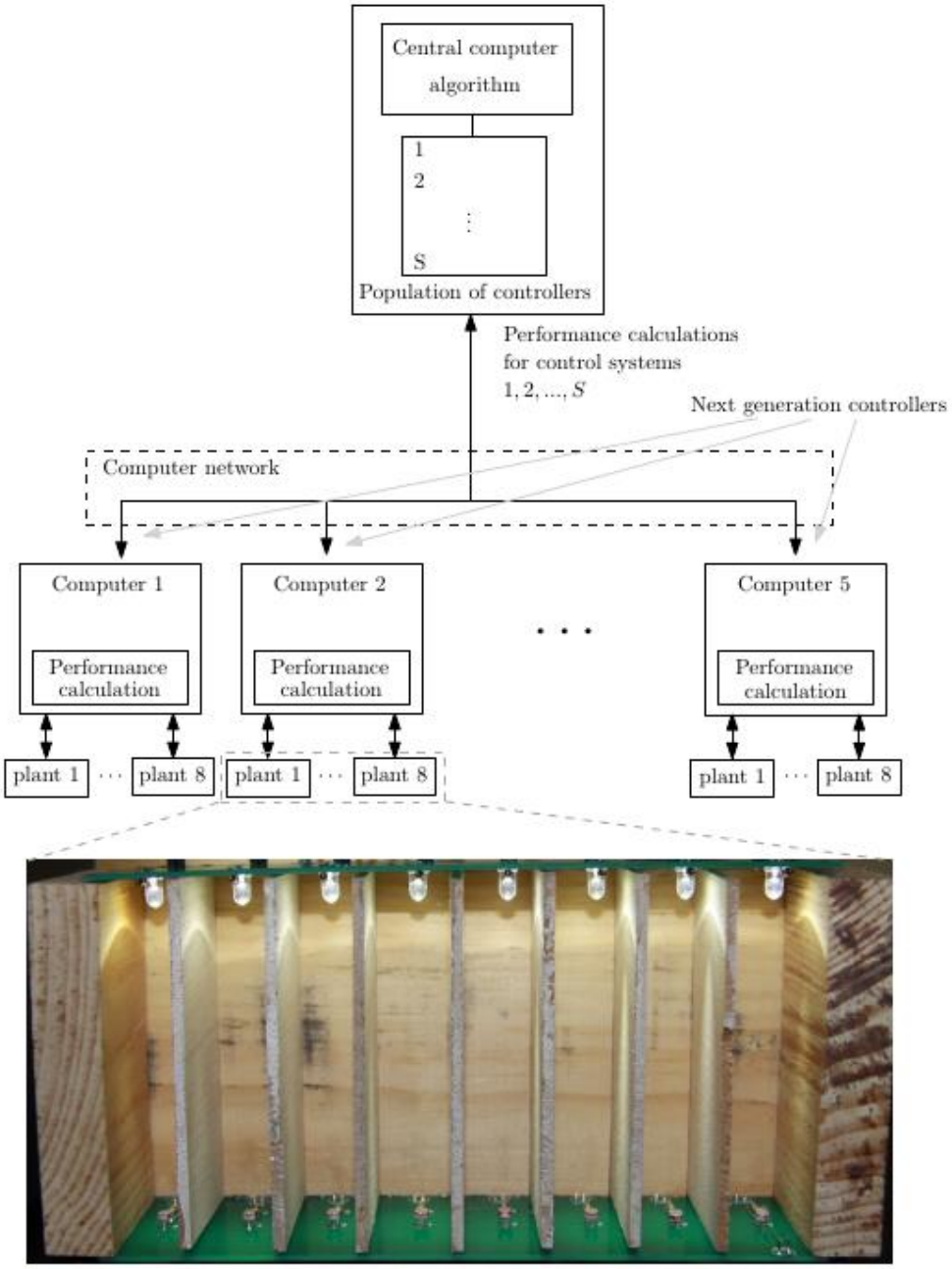

Figure 2: Distributed feedback control systems over a network. Each set of plants has eight individually controlled light systems, which include eight LED-photocell pairs. A single computer's eight physical plants are displayed in the image. The central computer algorithm provides each plant with next generation controllers. The controllers are tested and return their performance calculations to the central algorithm.

Closed-Loop Control System Performance Evaluation

A population of controllers

$$P(k) = \{\theta^j(k): j = 1,2,\dots,S\} \tag{3}$$

where S is the number of individuals in the population and

$$\theta^j(k) = \left[K_P^j, K_I^j\right]. \tag{4}$$

At $t = 0$, each “test" (run of the control system) initializes the LED voltage at 0V and sets the feedback control gains per Equation (4). Each feedback control loop generates a response with a 1ms sampling time for 1s. Once step k has completed, each local computer calculates the performance of its closed-loop system. The performance evaluation for each control system is

$$J^j(k) = w_s J_s^j(k) + w_{os} J_{os}^j(k) + w_{st} J_{st}^j(k) + w_{rt} J_{rt}^j(k) \tag{5}$$

which is a combination of steady state Equation (5), overshoot Equation (6), rise time Equation (7), and settling time Equation (8) values according to

$$J_s^j(k) = \left|\frac{s^j - r}{r}\right|, s^j = \frac{1}{L}\sum_{t=0.8t_f}^{t_f} y^j(t) \tag{6}$$

$$J_{os}^j(k) = \frac{\max_t\{y^j(t)\} - s^j}{s^j} \tag{7}$$

$$J_{rt}^j(k) = \frac{t_r^j}{t_f} \tag{8}$$

$$J_{st}^{j}(k) = \frac{t_s^j}{t_f} \tag{9}$$

Here, L is the number of steps in the last 20% of the signal $y^j(t)$, t_r^j is the 10%-90% rise time, t_s^j is the 2% settling time, and t_f is the total test time of 1s. The performance evaluation parameters were designed to be a percentage of a total. Therefore, the weights of this function were each set to 100 to describe the percentage as can be seen later in the results. These weights may be adapted to place more emphasis on specific performance attributes, but this functionality in the performance calculation is not tested here.

Non-Gradient Optimization Methods

Genetic Algorithm Optimization Method

Following a standard genetic algorithm (GA) optimization approach, the “fitness" (performance value) for each individual controller is gathered and used to form a new generation of individuals. Each individual has an opportunity to pass its traits down to the next generation of controllers through a “mating" process in which individuals are paired based on

$$p^j(k) = 1 - \frac{J^j(k)}{\sum_{i=1}^{S} J^i(k)}.$$

A random pair of individuals mate S times based on their $p^j(k)$ probabilities to form a new generation of S individuals. Furthermore, once two individuals have been paired, “crossover" of traits may only occur at a crossover probability, p_c. Once a pair of individuals is selected and p_c allows for crossover to occur, the algorithm selects a single trait (gain) to be exchanged. Once a trait is selected, a specific gene (single digit of a gain) is selected to be transferred from one individual to the other. All of the subsequent genes in that trait also transfer to the other mate to complete the mating process.

The example in Figure 3 shows the transfer of genes from two individuals, $\theta^5(k) = [001.045,010.012]$ and $\theta^9(k) = [005.984,026.973]$. Notice that each trait always includes six genes with the decimal place between the third and fourth genes. The

decimal location cannot be moved, and is therefore not considered in the mating process. In this example, the crossover occurs at the fourth gene in the first trait to form the first individual of the next generation, $\theta^1(k+1)$.

$\theta^5(k)$	0	0	1	0	4	5	0	1	0	0	1	2
				↓	↓	↓						
$\theta^9(k)$	0	0	5	9	8	4	0	2	6	9	7	3
$\theta^1(k+1)$	0	0	5	**0**	**4**	**5**	0	2	6	9	7	3

Figure 3:GA crossover example. The fourth gene is randomly selected for crossover along with all subsequent genes to produce θ^1 (k+1) for the next generation of controllers

Once the crossover stage is complete, a new generation of individuals, $P(k+1)$, is ready to be tested on the physical plants. However, in order to allow for a more complete search for an optimal solution, a mutation stage is included in the GA optimization method. For some low probability, p_m, an individual can be mutated. When mutation occurs, all genes in that individual are randomly reselected.

Set-Based Stochastic Optimization Method

The SBS optimization method also uses a population of individuals as described earlier in Equations (3) and (4); gathering the performance values of each individual defined in Equations (5)-(9) determines the next generation of individuals. However, this

non-gradient algorithm selects the best designed individual, j^*, from the previous generation, via

$$j^* = \text{argmin}_j J^j(k)$$

and creates a “cloud" of new individuals around the best individual by selecting the next generation of individuals via

$$\theta^j(k+1) = \theta^{j^*}(k) + \beta r^j$$

where r^j is a random number from a normal distribution with zero mean and unit variance and β is a scaling factor.

In this way, a cloud of individuals centered about the best individual is created as the next generation. Furthermore, to allow the algorithm to search randomly, a mutation probability is again added. Given the mutation probability, p_m, a random individual from the previous generation is selected for the next generation by

$$\theta^j(k+1) = \theta^j(k), j \neq j^*$$

However, instead of using a constant mutation probability as done in the GA optimization method, this method adapts p_m as the performance changes according to

$$p_m(k) = \frac{1}{10}\sum_{j=1}^{S} J^j(k)$$

Here, p_m is defined as 10% of the total performance of the previous generation. Due to the narrow search space of the SBS optimization method governed by β, a varying mutation probability is necessary to move the search away from local minima in early generations but decrease the mutation probability in later generations when the cost is converging on an optimal point for all control systems.

Experimental Results

Since each system has unique circuit devices and construction, variances in individual control gain choices are expected to be found. Therefore, the distributed optimization algorithms were tested on the light control systems repeatedly to represent a Monte Carlo method and to enable statistical analyses of the results.

Here, one set of generations is considered a test, and a set of tests is one experiment. A test number is denoted by n, and the total number of tests is N. Each individual initiates the test with a random pair of initial gain values on the set [0, 100]. Once a test completes the fixed amount of generations, the performance and gains are stored for future analyses. As described in Equation (10), matrix M_J^n stores the performance values for each individual in each generation. When the experiment is complete, each test matrix is averaged via Equation (11) to acquire statistical data for each generation. Similarly, the gain matrices are created and averaged to form $\overline{K}_P$ and $\overline{K}_I$ in Equations (12) - (15),

$$M_J^n = \left[J^j(1), J^j(2), \ldots, J^j(k)\right] j = 1,2, \ldots, S \tag{10}$$

$$\bar{J} = \frac{1}{N}\sum_{n=1}^{N} M_J^n \tag{11}$$

$$M_P^n = \left[K_P^j(1), K_P^j(2), \ldots, K_P^j(k)\right] j = 1,2, \ldots, S \tag{12}$$

$$\overline{K}_P = \frac{1}{N}\sum_{n=1}^{N} M_P^n \tag{13}$$

$$M_I^n = \left[K_I^j(1), K_I^j(2), \ldots, K_I^j(k)\right] j = 1,2, \ldots, S \quad (14)$$

$$\overline{K_I} = \frac{1}{N}\sum_{n=1}^{N} M_I^n \quad (15)$$

The plots in Figure 4 show the result of a GA optimization method experiment, which consisted of 25 tests with 30 generations in each test. Figure 4 shows the average performance matrix, $\bar{J}$, as well as the convergence of the gain matrices, $\overline{K_P}$ and $\overline{K_I}$. The algorithm converges to a high K_I gain and a low K_P gain for each individual. The performance converges to a low cost of less than 10% combined error. Given that the photocell device specifications state a typical total rise time of 55ms, the final performance of the population is near optimum based on the response of the system of a sample case shown in Figure 5. Furthermore, the convergence to a thin interquartile range in Figure 4 shows that the experiment was repeatable over the 25 tests.

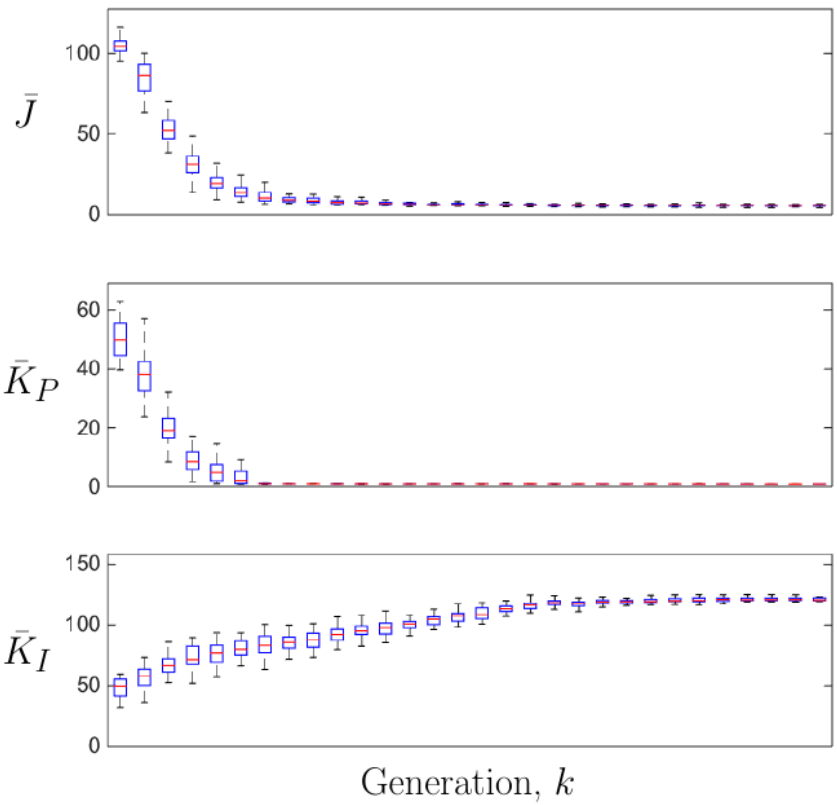

Figure 4: Performance and gain statistics of the GA with $p_c = 80\%$ and $p_m = 0.5\%$. This box plot contains the interquartile range from the first to the third quartile, the centerline of the box indicating the second quartile (median), the extended line representing the data set without the outliers, and the omission of any outliers for clarity.

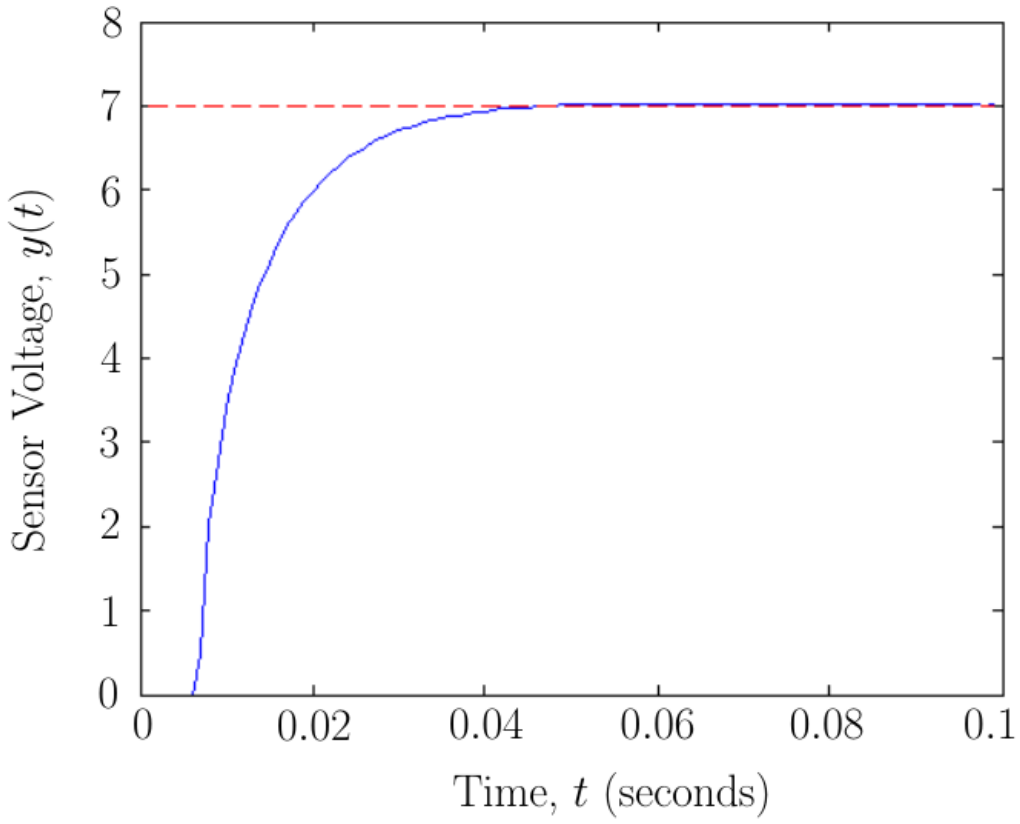

Figure 5: Response of a single control system with $K_P = 0.767$ and $K_I = 120.062$. Response parameters: $J_{rt} = 0.018s, J_{st} = 0.071s, J_{os} = 4.922 \times 10^{-4}, J_{ss} = 4.842 \times 10^{-4}$, resulting in a total $J = 8.998$.

Using the same physical plant in Equations (1) and (2), population in Equations (3) and (4), performance function in Equations (5)-(9), and initial conditions as the GA optimization method experiment, a comparison can be made with the SBS optimization method. The plots in Figure 6 represent the statistics of an experiment with 25 tests and 30 generations. Again, the experiment shows convergence to a low average $\bar{J}$ as well as convergence to a desired set of gains. Also, the thin interquartile range for the performance and gains displays repeatability in all 25 tests. In comparison to Figures 4 and 6, similar but not identical results are obtained.

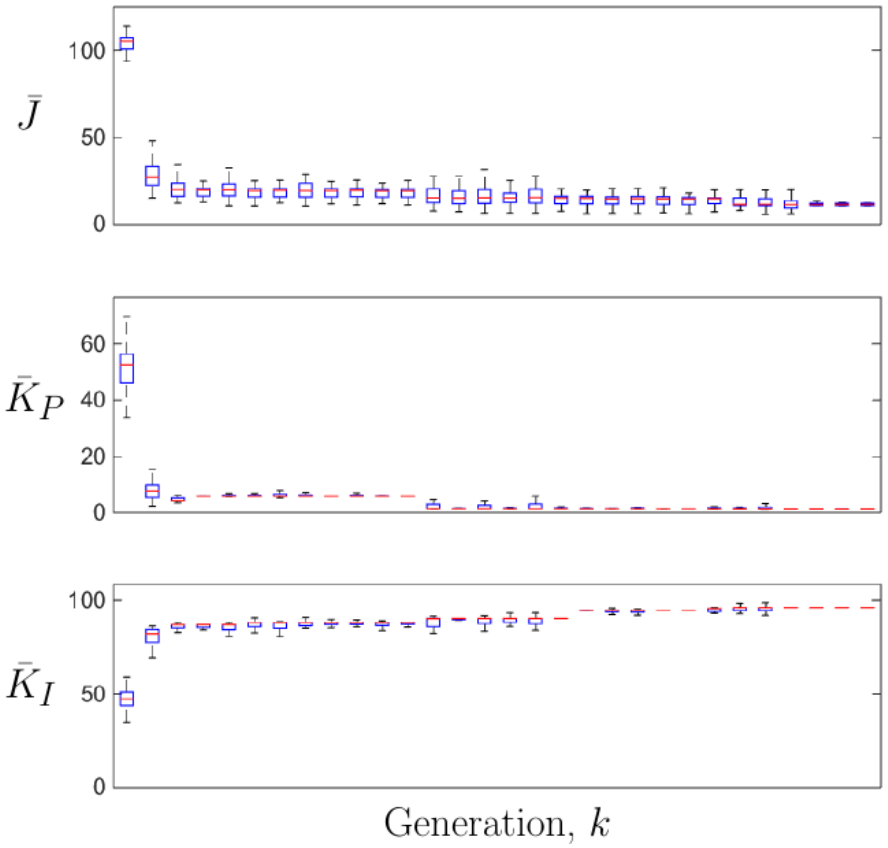

Figure 6: Performance and gain statistics for the SBS optimization method with $\beta = 0.01$. This box plot contains the interquartile range from the first to the third quartile, the centerline of the box indicating the second quartile (median), the extended line representing the data set without the outliers, and the omission of any outliers for clarity.

Conclusion

Utilizing non-gradient optimization methods on light control systems over a distributed network, experiments were implemented on many feedback control systems and have shown convergence and robustness. Both non-gradient optimization methods selected desirable gains for the light control system. Moreover, *all* feedback control systems in the network had desirable responses, which shows a type of robustness. Finally, these non-gradient optimization methods allow control engineers to retrieve robust control designs without the need to model the system. It is hopefull that the distributed non-gradient optimization methods discussed in this paper will have many potential applications on the IoT.

References

[1] O. Vermesan and P. Friess. *Internet of Things: Global Technological and Societal Trends*. River Publishers, Aalborg, 2011.

[2] E. Anzelmo, A. Bassi, D. Caprio, and R. Kranenburg. Internet of Things. *1st Berlin Symposium on Internet and Society (Version electronica), Consultado el*, 20, 2011.

[3] G. Kortuem and F. Kawsar. Smart objects as the building blocks for the internet of things. *IEEE Internet Computing*, 14(1).

[4] T. Ernst and K. Uehara. Connecting automobiles to the internet. *ITST: International Workshop on ITS Telecommunications*, 3, 2002.

[5] K. M. Passino. *Biomimicry for Optimization, Control, and Automation*. Springer, London, 2004.

[6] K. F. Man, K. S. Tang, and S. Kwong. Genetic algorithms: Concepts and applications. *IEEE Trans. on Industrial Electronics*, 43(5).

[7] M.A. Lee and H. Takagi. Intergrating design stage of fuzzy systems using genetic algorithms. *Second IEEE Int. Conf. on Fuzzy Systems*, 1:612–617, 1993.

[8] D. Dasgupta. *Evolutionary Algorithms in Engineering Applications*. Springer, Berlin, 1997.

[9] P.J. Fleming and R.C. Purshouse. Evolutionary algorithms in control systems engineering: A survey. *Control Engineering Practice*, 10(11), 2002.

[10] F. Rubinstein, M. Siminovitch, and R. Verderber. Fifty percent energy savings with automatic lighting controls. *IEEE Trans. Ind. Appl.*, 29(4).

[11] M. T. Koroglu and K. M. Passino. Illumination balancing algorithm for smart lights. *IEEE Trans. on Control Systems Technology*, 22(2).