Proceedings of the American Control Conference
Arlington, VA June 25-27, 2001

Robust Tuning of Fixed-Structure Controller for Disk Drives Using Statistical Model and Multi-Objective Genetic Algorithms

Bo Zhu[*], Ho Seong Lee[†], Lin Guo[‡] and Masayoshi Tomizuka[§]
[*,§]Department of Mechanical Engineering
University of California at Berkeley, Berkeley, CA 94720
[†,‡]Advanced Technology Group, Maxtor Corporation
510 Cottonwood Drive, Milpitas, CA 95035

Abstract

This paper proposes a non-gradient based method for the parameter optimization of fixed-structure controllers in hard disk drives (HDDs). Besides satisfying multiple frequency-domain constraints, the primary target of HDD servo design aims at the minimization of position error signal (PES) for a large population of drives. This is made possible by adopting a new statistical disturbance model inside the optimization loop to evaluate the time-domain performance of candidate controllers. The convexity of multi-dimension searching space is lost because the controller structure is fixed. This non-convex multiobjective optimization problem (MOP) is solved by multiobjective genetic algorithms (MOGA), which are genetic algorithms (GA) combined with the concept of Pareto optimality. Multiple optimal solutions with trade-offs are provided to support decision making. A design example of tuning a track following controller is used to demonstrate the effectiveness of the proposed method.

1 Introduction

Given cost-limited hardware and computing resources, designing a positioning system for mass-produced hard disk drives (HDDs) is a challenging work. The controller needs to achieve acceptable performance in the presence of external disturbances and measurement noises. Furthermore, the plant parameters vary from drive to drive due to manufacturing tolerances and also change with time and temperature. The controller therefore needs to provide sufficient robustness against these variations. On the other hand, the usage of economical digital signal processor in servo system requires a low order design of controller. This is a typical optimization problem with multiple non-commensurable objectives and constraints. Recently, linear matrix inequalities(LMIs) have received a great deal of attention in dealing with these kinds of multi-objective optimization problems (MOPs) [1]. LMIs provide a unifying framework to setup multiple *quadratic* performance inequalities, which are successfully solved by convex optimization approaches. It has been shown that a number of multi-objective problems, including the popular mixed H_2/H_∞ optimal control problem [2], can be reduced to solving LMIs. However, for the practical control of systems like HDD, it is rather difficult to fit the problem into a solvable convex form because of numerous design specifications and constraints. Furthermore, an optimal solution from the free-order synthesis is a high order controller, which is not implementable in HDD even with order reduction techniques. In fact, it is more efficient to start with a controller structure that has good nominal properties, and optimize the parameters within that structure. Fixing the structure of controller minimizes the coding difficulties involved in design iterations and enhances the controllability of overall firmware. However, since it also inevitably loses the convexity of the search space, gradient-based methods [3] will have no guarantee of convergence and require lots of auxiliary conditions.

This paper presents a non-gradient based solution to the design problem above by using multi-objective genetic algorithms (MOGA), which are the combinations of genetic algorithms (GA) and Pareto optimization. GAs are non-gradient based optimization methods that imitate the stochastic mechanisms of natural selection and genetic variation. Pareto optimality, which defines the optima based on vectorial performance, eliminates the use of weightings before optimization and provides multiple optimal solutions to decision maker. A new statistical disturbance model is included inside the optimization loop to evaluate the time-domain performance of candidate controllers [4]. This makes it possible to directly tune the controller towards the minimization of

---

[*] (510)642-0935; bozhu@me.berkeley.edu
[†] (408)432-4540; ho_lee@maxtor.com
[‡] (408)432-4529; lin_guo@maxtor.com
[§] (510)642-0870; tomizuka@euler.me.berkeley.edu

0-7803-6495-3/01/$10.00 © 2001 AACC

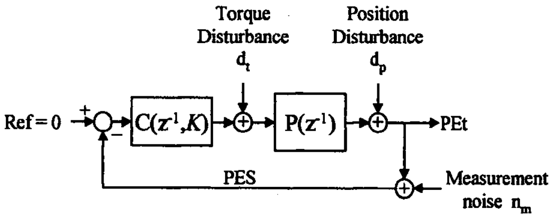

Figure 1: A typical blockdiagram of track following control

position error signal (PES) for a family of drives. Finally, the effectiveness of the proposed method is shown by an example of tuning a track following controller for HDD.

2 Problem Formulation

Figure 1 shows a discrete time model of HDD servo system. $P(z^{-1})$ is the ZOH equivalence of continuous time plant dynamics, including power amplifier (PA), voice coil motor (VCM), and head-disk assembly (HDA). For convenience, disturbances and noises are lumped into three sources depending on their injecting points in the loop: torque disturbance d_t, position disturbance d_p and measurement noise n_m. Note that the true position error PE_t can not be measured directly and PES is always contaminated by measurement noise n_m. $C(z^{-1}, K)$ represents the fixed-structure low-order digital controller with m tunable parameters $K = \{k_1, ..., k_m\} \in \Omega_K \subset R^m$ where Ω_K is the non-convex search space with boundaries defined by designer.

The most important performance index of track following servo is the variance of PES, σ_{PES}. Since we are designing the controller for the mass produced HDDs with plant variations and disturbance uncertainties, σ_{PES} is not a constant but rather a random variable. It is more appropriate to assess the average performance $mean(\sigma_{PES})$ and the performance robustness $var(\sigma_{PES})$. To get a precise and efficient prediction of $mean(\sigma_{PES})$ and $var(\sigma_{PES})$ for a given controller, a statistical model is built based on the PESs measured from a large population of drives [4]. This model not only covers the characteristics of a large population of drives but also requires little computational efforts in calculating $mean(\sigma_{PES})$ and $var(\sigma_{PES})$. The design also needs to satisfy some minimum relative stability margins. In this paper, all constraints are related to the frequency domain requirements, such as the minimum phase margin PM_{MIN} which is a direct safeguard against time delay uncertainty, the minimum gain margin GM_{MIN} which is a direct safeguard against steady-state gain uncertainty, the minimum crossover frequency ω_{oMIN}, and the maximum peak of sensitivity function $S_{\infty MAX}$.

There are couple difficulties in solving this non-convex MOP. First, the most widely used method is to combine all objectives into a single cost function by weightings before optimization. Apparently, the final solution depends on the selection of weightings. However, the best set of weight functions is not known in advance. This dependence will become problematic in practice as there are so many trade-offs among design objectives. The whole design iteration will become inefficient due to tremendous trial-and-errors. Secondly, in designing the controller for HDD, it is desired to provide the decision maker with a number of "trade-off" solutions which reveal the characteristics of solution space before a final solution is chosen. However, the method of using weighting only gives a single solution at each iteration. Finally, due to non-convex nature of this MOP, the gradient-based methods, like hill-climbing, are likely to be trapped in the local minima or fail to converge. In the following section, MOGA is shown to effectively overcome these difficulties.

3 Multi-Objective Genetic Algorithms (MOGA)

GA is a biologically inspired global searching method. The basic concepts were developed by Holland [5], Goldberg [6] and Michalewicz [7]. A simple GA is an iterative procedure starting with a randomly generated population of candidate solutions. For the current generation, each candidate is first evaluated via the fitness which is a real number quantifying its quality with respect to the targeting problem. The Roulette-wheel-like *selection* is more likely to pick up candidates with higher fitness values into a parents pool. The selected solutions (parents) are then processed by applying *crossover* which pairs up the parents and exchanges some of their parts pairwisely. Finally, *Mutation* performs a slight perturbation to the resulting solutions. On the basis of these genetic operators and the evaluations, a better new generation is formed [8]. The convergence of GA can be proved by *Schema Theorem* [6]. To deal with multiple non-commensurable objectives, several multi-objective genetic algorithms (MOGA), which are GAs combined with the concept of Pareto optimality, have been proposed [6] [9]. A good review on MOGA is in [10]. Many researchers [11] [12] [13] have successfully applied MOGA to solve MOPs in control systems design.

The flow chart of the MOGA used in this paper is shown

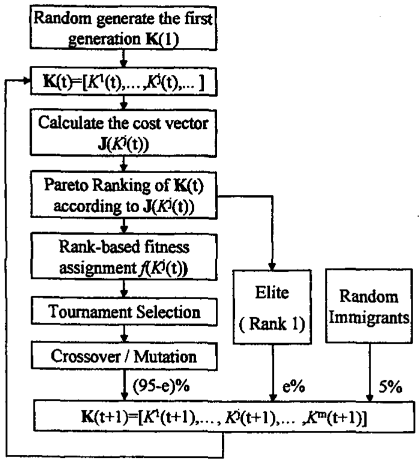
Random generate the first generation K(1)
K(t)=[K¹(t),...,Kʲ(t),...]
Calculate the cost vector J(Kʲ(t))
Pareto Ranking of K(t) according to J(Kʲ(t))
Rank-based fitness assignment f(Kʲ(t))
Tournament Selection
Crossover / Mutation
Elite (Rank 1)
Random Immigrants
(95-e)%
e%
5%
K(t+1)=[K¹(t+1),..., Kʲ(t+1),... ,Kᵐ(t+1)]

Figure 2: The flow chart of MOGA with elitism and random imigrants

in Figure 2. The details are described below. A summary of this MOGA is given at the end of this section.

Real-coded representation In this paper, each candidate solution, or called individual, is represented as a float-point vector $K^j = [k_1^j, ..., k_m^j]$. At $(t)th$ generation, MOGA maintains a population of individuals, $\mathbf{K}(t) = [K^1(t), ..., K^j(t), ..., K^{n_{pop}}(t)]$,with a fixed size n_{pop}. This real-coded (RC) representation has many advantages over the classical binary-coded (BC) representation which uses a bit string to code candidate solution [7]. 1) RC is faster, more consistent from run to run and provides a higher precision than BC, especially with large search domains. 2) RC is intuitively closer to the problem space, so virtually no decoding required. 3) RC avoids Hamming Cliffs effect from which BC suffers.

Cost functions [13] shows a genetic method to solve LMIs. However, limiting the objective functions in quadratic form does not fully use the potential of GA. Since GA is a population-based probabilistic optimization method and does not require the convexity of search space, the cost function can be freely chosen for the convenience of designer. The cost functions associated with the design objectives and constraints are defined as

$$\begin{aligned} J_1(K) &= var(\sigma_{PES}) \qquad (1) \\ J_2(K) &= mean(\sigma_{PES}) \\ J_3(K) &= Q[PM_{MIN} - PM(K)] \\ J_4(K) &= Q[GM_{MIN} - GM(K)] \\ J_5(K) &= Q[\omega_{oMIN} - \omega_o(K)] \\ J_6(K) &= Q[S_\infty(K) - S_{\infty MAX}(K)] \end{aligned}$$

where

$$Q[a] = \begin{cases} a & a > 0 \\ 0 & a \leqslant 0 \end{cases} \qquad (2)$$

and the phase margin $PM(K)$, the gain margin $GM(K)$, the crossover frequency $\omega_o(K)$, and the peak of sensitivity function $S_\infty(K) = \|S(K, j\omega)\|_\infty$ are functions of the tunable parameter set $K \in \Omega_K$. $J_3(K) \sim J_6(K)$ are constraint cost functions which will be zero if the candidate solution K satisfies all constraints. Thus, the original constrained MOP is converted to a unconstrained MOP whose target is to minimize the cost vector $\mathbf{J}(K) = \{J_1(K), ..., J_i(K), ..., J_6(K)\}$ over K. $\mathbf{J}(K)$ is therefore called the *vectorial performance* index.

Pareto optimality and Pareto Ranking The concept of *Pareto optimality* eliminates the use of weightings and makes it possible to provide multiple optimal solutions to support decision making. The vector $\mathbf{J}(K^1) = \{J_1(K^1), ..., J_n(K^1)\}$ is said to *dominate* vector $\mathbf{J}(K^2) = \{J_1(K^2), ..., J_n(K^2)\}$ if and only if $\mathbf{J}(K^1)$ is partially less than $\mathbf{J}(K^2)$, denoted as $\mathbf{J}(K^1) <_p \mathbf{J}(K^2)$, more precisely

$$(\forall i) J_i(K^1) \leqslant J_i(K^2) \wedge (\exists i) J_i(K^1) < J_i(K^2). \qquad (3)$$

A solution K^1 is *Pareto optimal* if and only if there is no $K^2 \in \Omega_K$ such that $\mathbf{J}(K^1) <_p \mathbf{J}(K^2)$. Pareto optimal solutions $\mathbf{K}_p$ are also called non-dominated or non-inferior set, which are a set of K^i such that

$$(\mathbf{J}(K^i) \not<_p \mathbf{J}(K^j)) \wedge (\mathbf{J}(K^j) \not<_p \mathbf{J}(K^i)), \\ K^i \in \mathbf{K}_p, K^j \in \mathbf{K}_p, \forall i \neq j. \qquad (4)$$

All K^i in the Pareto optimal set have the similar vectorial performance and thus are so-called "the equally best solutions" among the current generation $\mathbf{K} = [K^1, ..., K^{n_{pop}}]$. Therefore they are assigned the same rank of 1. The final solution of a MOP depends only on the vectorial performance and on the preferences of the decision maker, and not on any subsequent optimization [14]. Based on (3) and (4), a Pareto ranking [15] is performed as follows

1. Sort $\mathbf{K} = [K^1, ..., K^j, ..., K^{n_{pop}}]$ from the least to the largest according to $\|\mathbf{J}(K^j)\|_1 = \sum_{i=1}^n J_i(K^j)$. The sorted vector is denoted as $\mathbf{K}_s$. Let RNK=1.
2. Use the first entry of $\mathbf{K}_s$, K_s^1, as criterion. Take out any K_s^j from $\mathbf{K}_s$ and put it into a dominated vector Φ_d if $\mathbf{J}(K_s^1) <_p \mathbf{J}(K_s^j)$, $j = 2, ..., length(\mathbf{K}_s)$.
3. Move out K_s^1 from $\mathbf{K}_s$ and put it into a Pareto optimal set $\mathbf{K}_p$.
4. Let the remaining candidate solutions in $\mathbf{K}_s$ to form a new $\mathbf{K}_s$, then repeat 2 to 4 until all dominated solutions are removed.

5. Assign all entries in $\mathbf{K}_p$ with the same rank, RNK. Empty $\mathbf{K}_p$.
6. Replace $\mathbf{K}_s$ with Φ_d, i.e. $\mathbf{K}_s = \Phi_d$.
7. $RNK = RNK+1$ and repeat 2 to 7 until the entire population is ranked,

Rank-based fitness assignment Every candidate solution K is assigned a fitness value $f(K)$ which is the measurement of solution quality. For a candidate solution in MOGA, the smaller the rank number, the better the vectorial performance. By selection, the MOGA is biased to the solution with higher fitness value. Therefore the fitness assignment is such a mapping that maximizing the fitness $f(K)$ is equivalent to minimizing the cost vector $J(K)$, i.e. maximizing the vectorial perform of K. In this paper, a simple exponential mapping is used

$$f'(K) = \frac{1}{rank(K(t))}. \tag{5}$$

Fitness sharing Although all "equally good" solutions are assigned the same fitness, their actual choice to be selected as parents may differ due to the random nature of selection. This imbalance can be accumulated with the evolutions such that the population drift towards an arbitrary region of the trade-off surface, a phenomenon known as *genetic drift* [16]. *Fitness sharing* is a mechanism to counteract the genetic drift by re-distributing the fitness among the candidate solutions with the same rank [14]. In this paper, the sharing function is defined as

$$f(K^j) = \frac{D(K^j)}{\sum_j D(K^j)} \sum_j f'(K^j) \tag{6}$$

where $D(K^j(t))$ is the total mutual geometric distance in search space from $K^j(t)$ to all other individuals with the same rank. This function penalizes the fitness of individuals in popular neighborhoods and is in favor of more remote individuals.

Elitism and random migrants To increase the converging rate of MOGA, the elite (individuals with the Rank 1) of current generation are directly copied to the new generation, which will be $e\%$ of total population. $(95 - e)\%$ individuals are generated from selection/crossover/mutation process. As a complementary mechanism of mutation, the remaining 5% are generated randomly to preserve the population diversity.

The MOGA used in this paper is summarized as follows.

1. Determine the search space of K, $\Omega_K \subset R^m$, which is the range of K that stabilizes the nominal plant $P_0(z^{-1})$ by Jury Criterion [17].

Table 1: MOGA configerations

$n_{pop} = 200$	$Pr(crossover) = 1$	$e\% \leqslant 30\%$
$n_{gen} = 60$	$Pr(mutation) = 0.02$	

Table 2: Design objectives before and after MOGA optimization

	Before Optimization		After Optimization	
	Simulation	Experiment	Simulation	Experiment
$mean(\sigma_{PES})$	5.240	5.219	5.032	5.011
$var(\sigma_{PES})$	2.253	2.311	2.095	2.154
GM	$6.1dB$	$6.3dB$	$6.1dB$	$6.2dB$
PM	42.1°	40.4°	41.0°	39.7°
ω_o	$540Hz$	$544Hz$	$552Hz$	$557Hz$
S_∞	$5.7dB$	$5.5dB$	$5.4dB$	$5.2dB$

2. The MOGA randomly and uniformly generates the first generation with n_{pop} individuals (candidate solutions), $[K^1(1), ..., K^j(1), ..., K^{n_{pop}}(1)] \in \Omega_K$. Each individual is represented by RC.
3. For each individual $K^j(t)$ in the current $(t)th$ generation, calculate the cost function vector $\mathbf{J}(K^j(t)) = \{J_1(K^j(t)), ..., J_i(K^j(t)), ..., J_n(K^j(t))\}$.
4. Pareto ranking of $[K^1(t), ..., K^j(t), ..., K^{n_{pop}}(t)]$ according to $\mathbf{J}(K^j(t))$.
5. Assign fitness to $K^j(t)$ based on its ranking , $f'(K^j(t)) = \frac{1}{rank(K^j(t))}$, and apply the fitness sharing, $f(K^j(t)) = \frac{D(K^j)}{\sum_j D(K^j)} \sum_j f'(K^j(t))$.
6. Directly migrate the elite, individuals with rank 1, to the $(t+1)th$ generation. This makes up $e\%$ of total population n_{pop}, where $e\%$ is up to 30%.
7. Apply the *tournament selection* [6] to generate $n_{pop} \cdot (95 - e)\%$ parents from the $(t)th$ generation. A linear crossover is used to produce $n_{pop} \cdot (95 - e)\%$ new individuals from these parents. Apply *mutation* to these new individuals.
8. Randomly generate 5% of total n_{pop} individuals in the search space Ω_K for the $(t+1)th$ generation.
9. Set $[K^1(t+1), ..., K^j(t+1), ..., K^{n_{pop}}(t+1)]$ as the current generation. Should this new generation achieve the optimization goal, stop the MOGA; otherwise go to step 3.

4 Design Example

The disk drives used in our experiments have a recording density of 19,300 TPI and a sampling rate of 10,800Hz. The track-following controller for these drives is an extended PID controller in the following form

$$C(z^{-1}) = k_0 \cdot \mathbf{F}(z^{-1}; k_1, k_2, k_I) \tag{7}$$

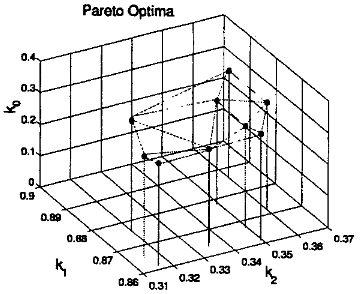

Figure 3: The Pareto optimal solutions by MOGA

where k_0 is the proportional gain, $\mathbf{F}(\cdot)$ is a function of z^{-1}, and constant parameters of k_1, k_2, and k_I. A digital notch filter is added to attenuate the adverse effect due to the suspension resonance at 2700Hz. The integrator gain, K_I, is predetermined and hence not a target of optimization. So the tunable parameter set is $K = [k_0, k_1, k_2]$. The original controller used by these drives was $[k_0, k_1, k_2] = [0.243, 0.850, 0.261]$, which was hand-tuned in the previous design cycle. PES of those drives were collected to build the statistical disturbance model. The search space $\Omega_K = [(0.01 : 0.98), (-0.8 : 0.99), (-0.8 : 0.99)]$ is the range of K that stabilizes the nominal plant. The gradient method used in [3] had always failed to converge in this range. The parameter for MOGA is listed in Table 1. Design constraints are $PM_{MIN} = 36^o$, $GM_{MIN} = 5.5dB$, $\omega_{oMIN} = 500Hz$, and $S_{\infty MAX} = 8dB$.

It took the MOGA 10.5 minutes in a PIII 550 computer to give a Pareto optimal set (Rank 1) with 9 solutions in the $60th$ generation, as shown in Figure 3. One solution, $K = [0.261, 0.864, 0.335]$, which has the minimal $var(\sigma_{PES})$ among the Pareto optimal set, was picked up and loaded into those drives. The performance comparison between this optimized controller and the original one is shown in Table 2. The 4% improvement of $mean(\sigma_{PES})$ and 7% of $var(\sigma_{PES})$ in experiments may be considered moderate because the original controller had been well optimized. It also can be seen that $mean(\sigma_{PES})$ and $var(\sigma_{PES})$ predicted by the model match up reasonably well with the experimental results. The disturbance model was then updated based on new PES measurements, and ready for the next run of optimization. This iterative optimization process can be repeated until satisfactory results are obtained.

5 Conclusion

In this paper, a non-convex MOP of tuning the fixed-structure track following controller for HDDs has been solved by using the MOGA. A disturbance model was used to effectively predict the time-domain performance of candidate solutions for a large population of drives. The controller parameters were then tuned by MOGA, directly towards the minimization of $mean(\sigma_{PES})$ and $var(\sigma_{PES})$ under various frequency-domain constraints. As shown by simulations and experiments, the proposed method was capable of optimizing the controller in a large range in which gradient-based methods generally failed. The proposed method is also very useful for other mass-produced electro-mechanical systems with fixed-structure controllers during the initial development phase.

References

[1] S. Boyd, V. Balakrishnan, E. Feron, and L. E. Ghaoui, "History of linear matrix inequalities in control theory," *Proceedings of the 1994 American Control Conference*, vol. 1, pp. 31–34, July 1994.

[2] K. Zhou, K. Glover, B. Bodenheimer, and J. Doyle, "Mixed h_2 and h_∞ performance objectives," *Proceedings of the 1990 American Control Conference*, pp. 2502–7, May 1990.

[3] H. S. Lee, "Controller optimization for minimum position error signals of hard disk drives," *Proceedings of the 2000 American Control Conference*, pp. 3081–3085, June 2000.

[4] B. Zhu, L. Guo, and M. Tomizuka, "A statistical PES model for direct controller optimization towards minimum PES in hard disk drives.." Maxtor Techanical Report, July 2000.

[5] J. H. Holland, "Outline for a logical theory of adaptive systems," *Journal of the Association for Computing Machinery*, vol. 3, pp. 297–314, 1962.

[6] D. E. Goldberg, *Genetic Algorithms in Search Optimization, and Machine Learning.* Reading, MA: Addison Wesley.

[7] Z. Michalewicz, *Genetic Algorithms + Data Structure = Evolution Programs, Second Extended Edition.* New York: Springer-Verlag Berlin Heidelberg, 1992.

[8] M. Pelikan, D. E. Goldberg, and F. Lodo, "A survey of optimization by building and using probabilistic models," *Proceedings of the 2000 American Control Conference*, pp. 3289–93, June 2000.

[9] J. D. Schaffer, "Multiple objective optimization with vector evaluated genetic algorithms," *Proceeding of 1st International Conference on Genetic Algorithms*, pp. 93–100, 1985.

[10] H. Tamaki, H. Kita, and S. Kobayashi, "Multiobjective optimization by genetic algorithm: A review," *Proceedings of 1996 IEEE International Conference on Evolutionary Computation*, pp. 517–522, May 1996.

[11] B. Chen, Y. Cheng, and C. H. Lee, "A genetic approach to mixed h_2/h_∞ optimal PID control," *IEEE Control Systems*, pp. 51–60, Octobor 1995.

[12] A. J. Chipperfield, N. V. Dakev, P. J. Fleming, and J. F. Whidborne, "Multiobjective robust control using evolutionary algorithms," *IEEE Proc. of the International Conference on Industrial Tech.*, pp. 269–273, 1996.

[13] T. Kawabe and T. Tagami, "A real coded genetic algorithm for matrix inequality design approach of robust PID controller with two degree of freedom," *Proceedings of the 12th IEEE International Symposium on Intelligent Control*, pp. 119–124, July 1997.

[14] C. M. Fonseca and P. J. Fleming, "Multiobjective genetic algorithm made easy: Selection, sharing and mating restriction," *Genetic Algorithm in Engineering Systems: Innovations and Applications*, pp. 45–52, September 1995. Conference Publication No 414.

[15] T. K. Liu, T. Ishihara, and H. Inooka, "Multiobjective control systems design by genetic algorithms," *Proceedings of the 34th SICE Annual Conference*, pp. 1521–26, July 1995.

[16] D. E. Goldberg and P. Segrest, "Finite markov chain analysis of genetic algorithm," *Proceedings of the 2nd International Conference on Genetic Algorithms*, pp. 1–8, 1987.

[17] G. F. Franklin, J. D. Powell, and M. L. Workman, *Digital Control of Dynamic Systems.* Addison-Wesley, 2nd ed., 1990.