PID Control Using Presearched Genetic Algorithms for a MIMO System

Jih-Gau Juang, *Member, IEEE*, Ming-Te Huang, and Wen-Kai Liu

***Abstract*—This correspondence presents a new approach that utilizes evolutionary computation and proportional-integral differential (PID) control to a multi-input multioutput (MIMO) nonlinear system. This approach is demonstrated through a laboratory helicopter called the twin rotor MIMO system (TRMS). The goals of control are to stabilize the TRMS in significant cross-couplings, reach a desired position, and track a specified trajectory efficiently. The proposed control scheme includes four PID controllers with independent input. In order to reduce total error and control energy, all parameters of the controller are obtained by a real-value-type genetic algorithm (RGA) with a system performance index as the fitness function. The system performance index was applied to the integral of time multiplied by the square error criterion to build a suitable fitness function in the RGA. We also investigated a new method for the RGA to solve more than ten parameters in the control scheme. The initial search range of the RGA was obtained by a nonlinear control design (NCD) technique. The NCD provided a narrow initial search range for the RGA. This new method led chromosomes to converge to optimal solutions more quickly in a complicated hyperplane. Computer simulations show that the proposed control scheme conquers system nonlinearities and influence between two rotors successfully.**

***Index Terms*—Genetic algorithms, multi-input multioutput (MIMO) system, proportional-integral differential (PID) control.**

I. Introduction

In this correspondence, we investigate a control problem involving an experimental propeller setup called the twin rotor multi-input multioutput system (TRMS) [1]. The TRMS is a laboratory setup designed for control experiments. In certain aspects, its behavior resembles that of a helicopter. From the control point of view, it exemplifies a high-order nonlinear system with significant cross-couplings. An estimated model has been obtained by using radial basis function networks [2] and a black-box system identification technique [3]. Here, we assume that the dynamics of the TRMS are known. The control objective is to stabilize the system in a coupled condition and make the beam of the TRMS move quickly and accurately to track a trajectory or reach specified positions in 2 DOF. The control scheme is based on the PID control technique, which applies a signal to the process that is proportional to the actuating signal in addition to adding integral and derivative of the actuating signal. PID is the control algorithm most often used in industrial control [4]. It is implemented in industrial single-loop controllers, distributed control systems, and programmable logic controllers. There are two reasons why it is the most used in the industrial process. The first reason is that its simple structure and the well-known Ziegler and Nichols tuning algorithms have been developed [5], [6] and successfully used for years. The second reason is that the controlled processes in industrial plants can almost be controlled through the PID controller [7], [8]. The drawback is that the parameters of the PID controller are partially tuned by a trial-and-error process, which makes it less intelligent. Su *et al.* [9] proposed a terminal sliding mode control to the TRMS and obtained good tracking performance. But the settling time of the time response is too long. Islam *et al.* [10] designed a fuzzy logic controller for the TRMS. Trajectory tracking performance is superior to the system performance with a conventional PID or a linear quadratic controller (LQR) controller. The drawback is that the system performance depends on the number of the fuzzy membership functions. In their simulations, the tracking error of the sine wave response cannot be eliminated. In this correspondence, parameters of the proposed controller are obtained by a modified method of a real-type genetic algorithm (RGA) with a performance index as the fitness function, which can reduce control energy and total error simultaneously.

Huang and Juang [11] investigated the effect of the binary genetic algorithm (BGA) on controller tuning for improving the system performance. The gains of the PID controller are tuned by a BGA. The controller is better than what was designed by the Gauss–Seidel minimization procedure. Tsai *et al.* [12] applied the RGA on PID tuning to the TRMS control in the vertical and horizontal axes separately. In order to design a 1 DOF PID controller, the system is decoupled into two parts, that is, vertical and horizontal. They restrict the connective beam between the main rotor and the tail rotor to move only in the vertical or horizontal plane. The impact on these two rotors is ignored. The parameters of the PID controller are tuned by the RGA to reduce the total error and control energy. Fan and Juang [13] compared different kinds of fitness functions in an RGA via trial and error for tuning an optimal PID controller on the TRMS. The computer simulations show that whether the fitness function best suits the objective or not can be checked by the control response. Although these PID controllers [12], [13] can work in 2 DOF, they are still weak in tracking a desired trajectory and in reaching a specific attitude. It is difficult to stabilize the influence between two axes and a nonminimum phase in a vertical plane. Liu *et al.* [14] quoted the system performance index as a part of the fitness function in an RGA. The system performance index deals with a modification of the known integral of time multiplied by square error criterion (ITSE). It is more efficient in finding the parameters of the four PID controllers. Although these controllers can reduce control force, and the total error is less than before, the trajectory tracking of a desired path in 2 DOF oscillates for several seconds. Above all, the main purpose in the control of the TRMS focuses on designing controllers to stabilize the impact between two rotors and track a desired path and specific attitude in 2 DOF efficiently.

The performance of the PID control depends on the gains of the controller. Proper gains can be obtained by some optimization techniques. Among them, the genetic algorithm (GA) has recently been used very often. The GA was first proposed by Holland [15] at the University of Michigan in 1975. The searching process, based on the Darwinian "survival of the fittest" principle, allows the solution to evolve into a superior solution. It has been successfully applied to various optimization problems [16]–[19]. The GA is easily combined with intelligent control theory (mixed or hybrid) [20]–[24] and solves problems in the traditional control field. However, the disadvantage of the GA is that the searching process is time-consuming for multiple parameters with a wide search range. To overcome this problem, we propose a nonlinear control design (NCD) technique that uses a quasi-Newton method with cubic/quadratic interpolation for unconstrained conditions and a sequential quadratic programming method for constrained conditions. The NCD can provide a narrow search range to the parameter searching process of the GA. In this correspondence, we investigate a cross-coupled PID controller on the TRMS through a modified method in

Manuscript received February 14, 2007; revised June 27, 2007, September 29, 2007, and December 14, 2007. This work was supported in part by the National Science Council, Taiwan, under Grant NSC 93-2213-E-019-007. This correspondence was recommended by Associate Editor J. Lazansky.

J.-G. Juang is with the Department of Communications and Guidance Engineering, National Taiwan Ocean University, 20224 Keelung, Taiwan, R.O.C. (e-mail: jgjuang@mail.ntou.edu.tw).

M.-T. Huang is with Chi Mei Optoelecronics Corporation, 74147 Tainan, Taiwan, R.O.C. (e-mail: mingtehuang@cmo.com.tw).

W.-K. Liu is with the Institute of Applied Mechanics, National Taiwan University, Taipei, 106 Taiwan, R.O.C. (e-mail: d94543001@ntu.edu.tw).

Color versions of one or more of the figures in this correspondence are available online at http://ieeexplore.org.

Digital Object Identifier 10.1109/TSMCC.2008.923890

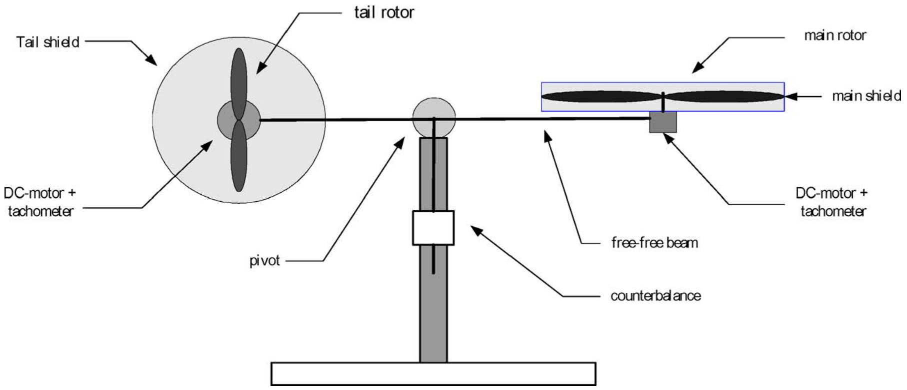


Fig. 1. Laboratory setup TRMS.

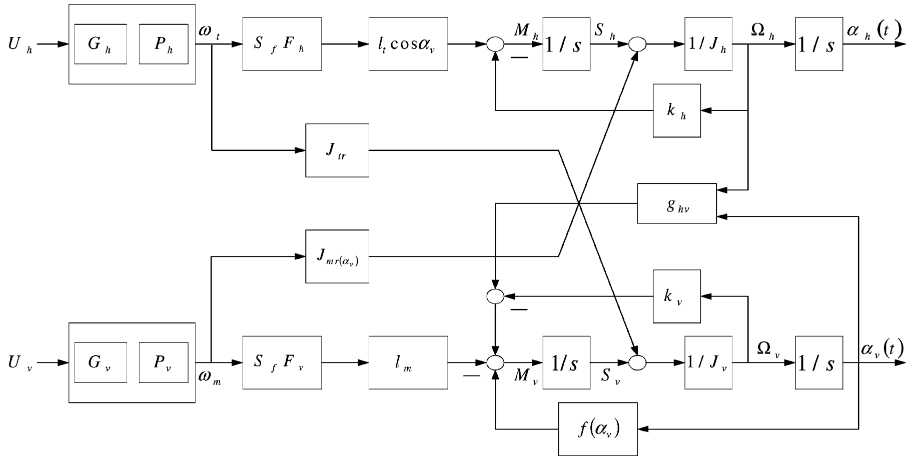


Fig. 2. Block diagram of TRMS model.

an RGA. This approach can optimize more than ten parameters in hyperplane simultaneously and converge to a better solution. Simulations show that the performance of the proposed controller is much better than in previous works.

II. System Description

The TRMS, as shown in Fig. 1, is characterized by complex and highly nonlinear functions with some inaccessible parameters for measurements, and therefore, can be considered a challenging engineering problem [1]. Approaching control problems connected with the TRMS proposed in this correspondence involves some theoretical knowledge of the laws of physics and some heuristic dependencies that are difficult to express in analytical form. A schematic diagram of the laboratory setup is shown in Fig. 1. The TRMS consists of a beam pivoted on its base in such a way that it can rotate freely both in the horizontal and vertical planes. At both ends of the beam, the rotors (the main and tail rotors) are driven by dc motors. A counterbalance arm with a weight at its end is fixed to the beam at the pivot. The state of the beam is described by four process variables: horizontal and vertical angles measured by position sensors fitted at the pivot, and two corresponding angular velocities. Two additional state variables are the angular velocity of the rotors, measured by tachogenerators coupled with the driving dc motors. In a normal helicopter, the aerodynamic force is controlled by changing the angle of attack. The laboratory setup from Fig. 1 is so constructed that the angle of attack is fixed. The aerodynamic force is controlled by varying the speed of the rotors. Therefore, the control inputs are the supply voltage of the dc motors. A change in the voltage value results in a change in the rotation speed of the propeller. This further results in a change of the corresponding position of the beam.

A block diagram of the TRMS model is shown in Fig. 2. It is suitable for use in the SIMULINK environment. The model of the dc motor with propeller is composed of a linear dynamic system followed by a static nonlinearity. The linear part is in the form of first-order transfer functions $G_h = 1/(T_h S + 1)$ and $G_v = 1/(T_v S + 1)$. The nonlinear functions $P_h(U_h)$ and $P_v(U_v)$ are static characteristics of the dc motors with propellers. The nonlinear relations between the rotor's velocity and the resulting aerodynamic force can be approximated by the quadratic function

$$F_h = \operatorname{sign}(\omega_t)\, k_h \omega_t^2 \tag{1}$$

$$F_v = \operatorname{sign}(\omega_m)\, k_v \omega_m^2 . \tag{2}$$

Rotation of the propeller produces an angular momentum, which, according to the law of conservation of angular momentum, must be compensated for by the remaining body of the TRMS beam. This results in the interaction between two transfer functions, represented by the moment of inertia of the motor with propellers J_{tr} and J_{mr} in Fig. 2. This interaction directly influences the velocity of the beam in both planes. The forces F_h and F_v multiplied by the length $l_h(\alpha_v)$ and l_v are equal to the torques acting on the arm. Parameters in Fig. 2 are defined as follows.

α_h, α_v Horizontal and vertical positions of TRMS beam.

Ω_h, Ω_v Horizontal and vertical angular velocities of TRMS beam.

U_h, U_v	Horizontal and vertical dc motor voltage control inputs.
G_h, G_v	Linear transfer functions of tail rotor and main rotor dc motor.
P_h, P_v	Nonlinear parts of dc motor with tail rotor and main rotor.
ω_t, ω_m	Rotational speeds of tail rotor and main rotor.
F_h, F_v	Nonlinear functions of aerodynamic force from tail rotor and main rotor.
l_t, l_m	Effective arms of aerodynamic force from tail rotor and main rotor.
J_h, J_v	Nonlinear functions of moment of inertia with respect to horizontal and vertical axis.
J_{tr}, J_{mr}	Moments of inertia in dc motor tail propeller subsystem and main propeller subsystem.
S_h, S_v	Angular momentums in horizontal and vertical planes for the beam.
S_f	Constant scaling factor.
M_h, M_v	Horizontal and vertical tuning torques.
k_h, k_v	Positive constants.
f_h, f_v	Moments of friction force in horizontal and vertical.
f_m	Vertical tuning moment from counterbalance.
g_{hv}	Nonlinear function of reaction tuning moment.

The mathematical model of the main rotor is

$$\frac{dS_v}{dt} = l_m S_f F_v(\omega_m) - \Omega_v k_v + g((A-B)\cos\alpha_v - C\sin\alpha_v) - \frac{1}{2}\Omega_h^2(A+B+C)\sin 2\alpha_v \tag{3}$$

$$\frac{d\alpha_v}{dt} = \Omega_v \tag{4}$$

$$\Omega_v = \frac{S_v + J_{tr}\omega_t}{J_v} \tag{5}$$

$$u_{vv} = \frac{U_v}{T_{mr}s+1} \tag{6}$$

$$\frac{du_{vv}}{dt} = \frac{1}{T_{mr}}(-u_{vv}+U_v) \tag{7}$$

$$\omega_m = P_v(u_{vv}) \tag{8}$$

where

$$A = \left(\frac{m_t}{2} + m_{tr} + m_{ts}\right) l_t$$

$$B = \left(\frac{m_m}{2} + m_{mr} + m_{ms}\right) l_m$$

$$C = \left(\frac{m_b}{2} l_b + m_{cb} l_{cb}\right)$$

m_{mr} is the mass of the main dc motor with the main rotor, m_m is the mass of the main part of the beam, m_{tr} is the mass of the tail motor with the tail rotor, m_t is the mass of the tail part of the beam, m_{cb} is the mass of the counterweight, m_b is the mass of the counterweight beam, m_{ms} is the mass of the main shield, m_{ts} is the mass of the tail shield, T_{mr} is the time constant of the main motor-propeller system, l_b is the length of the counterweight beam, l_{cb} is the distance between the counterweight and the joint, u_{vv} is the output of the vertical dc motor G_v, and g is the gravitational acceleration. Following are the values of the model parameters:

$$l_t = 0.25\text{ m} \quad l_m = 0.24\text{ m} \quad l_b = 0.26\text{ m} \quad l_{cb} = 0.13\text{ m}$$

$$r_{ms} = 0.155\text{ m} \quad r_{ts} = 0.10\text{ m} \quad m_{tr} = 0.206\text{ kg}$$

$$m_{mr} = 0.228\text{ kg} \quad m_{cb} = 0.068\text{ kg} \quad m_t = 0.0155\text{ kg}$$

$$m_m = 0.0145\text{ kg} \quad m_b = 0.022\text{ kg} \quad m_{ts} = 0.165\text{ kg}$$

$$m_{ms} = 0.225\text{ kg}.$$

Assuming that the main rotor is an independent system, then (3)–(8) can be written as

$$\frac{dS_v}{dt} = l_m S_f F_v(\omega_m) - \Omega_v k_v + g((A-B)\cos\alpha_v - C\sin\alpha_v) \tag{9}$$

$$\frac{d\alpha_v}{dt} = \Omega_v = 9.1 S_v \tag{10}$$

$$F_v = -3.48\times10^{-12}\omega_m^5 + 1.09\times10^{-9}\omega_m^4 + 4.123\times10^{-6}\omega_m^3 - 1.632\times10^{-4}\omega_m^2 + 9.544\times10^{-2}\omega_m \tag{11}$$

$$\omega_m = 90.99u_{vv}^6 + 599.73u_{vv}^5 - 129.26u_{vv}^4 - 1238.64u_{vv}^3 + 63.45u_{vv}^2 + 1283.4u_{vv}. \tag{12}$$

The mathematical model of the tail rotor is

$$\frac{dS_h}{dt} = l_t S_f F_h(\omega_t)\cos\alpha_v - \Omega_h k_h \tag{13}$$

$$\frac{d\alpha_h}{dt} = \Omega_h = \frac{S_h + J_{mr}\omega_m\cos\alpha_v}{J_h} = \frac{S_h + J_{mr}\omega_m\cos\alpha_v}{D\sin^2\alpha_v + E\cos^2\alpha_v + F} \tag{14}$$

$$u_{hh} = \frac{U_h}{T_{tr}s+1} \tag{15}$$

$$\frac{du_{hh}}{dt} = \frac{1}{T_{tr}}(-u_{hh}+U_h) \tag{16}$$

$$\omega_t = P_h(u_{hh}) \tag{17}$$

where

$$D = \frac{m_b}{3} l_b^2 + m_{cb} l_{cb}^2$$

$$E = \left(\frac{m_m}{3} + m_{mr} + m_{ms}\right) l_m^2 + \left(\frac{m_t}{3} + m_{tr} + m_{ts}\right) l_t^2$$

$$F = m_{ms} r_{ms}^2 + \frac{m_{ts}}{2} r_{ts}^2$$

where r_{ms} is the radius of the main shield, r_{ts} is the radius of the tail shield, and u_{hh} is the output of the horizontal dc motor. Assuming that the tail rotor is an independent system, then (13)–(17) can be written as

$$\frac{dS_h}{dt} = l_t S_f F_h(\omega_t) - \Omega_h k_h \tag{18}$$

$$\frac{d\alpha_h}{dt} = \Omega_h \tag{19}$$

$$\Omega_h = 90 S_h \tag{20}$$

$$F_h = -3\times10^{-14}\omega_t^5 - 1.595\times10^{-11}\omega_t^4 + 2.511\times10^{-7}\omega_t^3 - 1.808\times10^{-4}\omega_t^2 + 0.0801\omega_t \tag{21}$$

$$\omega_t = 2020u_{hh}^5 - 194.69u_{hh}^4 - 4283.15u_{hh}^3 + 262.27u_{hh}^2 + 3796.83u_{hh}. \tag{22}$$

A detailed description can be found in [25].

III. RGA and System Performance Index

The GA proposed by Holland is commonly called the simple genetic algorithm (SGA). It will generate a (generally very high) number of chromosome strings at random, each representing an individual in the initial or parent population, to which the evolutionary principles of selection and mutation are applied. For the selection mechanism, the user has to provide the criterion for determining the relative fitness of every individual. This can be done by providing a fitness function that allows us to classify each individual in relation to the average fitness or error of the population and then decide which one is better. The algorithm will then favor individuals with higher fitness and selects them to be propagated into the next generation. Typically, this evaluation process consumes most of the execution time—no matter whether the fitness is determined by calculation or by experiment. The new generation is produced by applying genetic operators to the selected individuals. Some basic operators are reproduction, crossover, and mutation, which are used during the computation course. These operations and propagation are repeated until an individual matches the termination criterion. It is necessary to stress the actual solution of the problem instead of sampling solutions. However, it is important to make a fitness function more adequate by letting the population converge to a solution quickly.

This correspondence uses real-value encodings of chromosomes rather than the more common binary encodings. There is no fixed rule for when a chromosome should be encoded as a bit string or as a real value. Though fixed-length, fixed-order binary encodings are the most common and best explained by the existing theoretical literature, "binary encodings are unnatural and unwieldy for many problems ..., and they are prone to rather arbitrary orderings." As Mitchell [26] said, "for many applications, it is most natural to use an alphabet of many characters or real numbers to form chromosomes." The use of real-value encodings allows the development of a GA based on fitness in relation to a target chromosome. That is, instead of searching the fitness-landscape for an unknown solution, the GA is given possible solution from the beginning. The fitness of each individual is measured in comparison to the cost value of any chromosome in the context of the given problem, the goal provided at initialization. As should be apparent, this is a trivial problem in comparison to typical GA applications, where the answer is not (and sometimes cannot be) known in advance. Rather, here the answer is not of interest; what is of interest is the path that each population takes toward the known destination. The starting point is a population of chromosomes randomly generated with alleles from the specific parameters of the predefined chromosome.

A. Procedure of RGA

1) Population: The RGA does not work on a single individual but on a population $\mathbf{P}$ with p individuals that undergoes an evolutionary process starting with the initial population $\mathbf{P}_0$. The simplest way to create $\mathbf{P}_0$ is to generate p strings or chromosomes randomly. Each gene in the chromosome represents a specific parameter of the PID controller. The population can be expressed as

$$\mathbf{P} = \left(\mathbf{X}_1, \ldots, \mathbf{X}_p\right). \tag{23}$$

In the encoding process, we assign values for each gene within a specific range, which will be described in Section IV. The resolution of each gene is 0.0001 in both the encoding and decoding processes.

2) Selection and Reproduction: The selection operator S selects an individual of the given population according to its fitness value. In this correspondence, we choose the Roulette wheel selection as the selection process [25]. This method is based on the survival-of-the-fittest mechanism; the individuals with higher fitness values have higher probabilities of producing offspring. The probability is

$$PS_i = \frac{\varsigma_i}{\sum_{i=1}^{\lambda} \varsigma_i} \tag{24}$$

where ς_i is the fitness value of each individual and λ is the population size. This operator reproduces the selected individuals to the mating pool

$$S' = R \times S \tag{25}$$

where S' is new individuals and S is selected individuals in parent generation. R is the reproduction operator.

3) Crossover: This operator exchanges the chromosome string of two selected individuals starting from a random index. Here, we use a modified crossover rule [27] for crossover process. It is explained as follows:

$$\begin{cases} x_{o1} = (1-\beta)\, x_{pm}^{\alpha} + \beta x_{pn}^{\alpha} \\ x_{o2} = \beta x_{pm}^{\alpha} + (1-\beta)\, x_{pn}^{\alpha} \end{cases} \tag{26}$$

where the two selected genes x_{pm}^{α} and x_{pn}^{α} are genes of the parents $\mathbf{X}_{pm} = \left[x_{pm}^{1}, \ldots, x_{pm}^{\alpha}, \ldots, x_{pm}^{\mu}\right]$ and $\mathbf{X}_{pn} = \left[x_{pn}^{1}, \ldots, x_{pn}^{\alpha}, \ldots, x_{pn}^{\mu}\right]$, respectively, α is the crossover position, which is an integer randomly generated within the range [1, μ], μ is the number of genes in an individual, and β is a real number randomly generated within the range [0, 1]. To form the offspring chromosomes, x_{o1} and x_{o2} replace x_{pm}^{α} and x_{pn}^{α}, respectively, then the genes on the left side of the crossover position x_{oi} are invariable, and the genes on the right side of the crossover position exchange with each other.

4) Mutation: This mutation operator creates one new offspring individual for the new population by randomly mutating a randomly chosen gene of the selected individual. For example, there are μ genes that represent d parameters in an individual $\mathbf{X} = [x_1, \ldots, x_k, \ldots, x_\mu]$, and the kth parameter is in the mutation position within the range $[\,\mathrm{UB}_k \quad \mathrm{LB}_k\,]$. In (27), the new gene is

$$x_{\mathrm{knew}} = \mathrm{LB}_k + r\left(\mathrm{UB}_k - \mathrm{LB}_k\right) \tag{27}$$

where r is a real number within [0, 1]. The individual after mutation is $\mathbf{X}_{\mathrm{new}} = [x_1, \ldots, x_{\mathrm{knew}}, \ldots, x_\mu]$.

5) Fitness Function: The fitness function F is used to guide the evolution in a certain direction. In fact, the genetic algorithm is merely a method of approximating the global maximum of F in the search space of chromosome strings. The actual interpretation of this search space is packed into f and does not concern the algorithm itself. In this correspondence, the fitness function illustrates the system performance, which will be described later.

In the beginning of the searching procedure of the RGA, an initial set of solutions, called the initial population, is generated randomly. Each individual in the population is called a chromosome, which is a string composed of genes that represents encoded parameters of the solution. The chromosomes evolve through successive iterations, called generations. The first step in each generation is to decode and evaluate current chromosomes through a predefined fitness function. The offspring is a new chromosome of the next generation and is formed through the following procedures: selection, reproduction, crossover, and mutation. The chromosomes of the new generation are decoded and evaluated by the corresponding fitness function. The earlier steps are repeated for several generations, then the evolution process converges

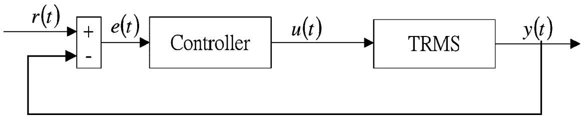


Fig. 3. TRMS control system.

to a chromosome that has the highest fitness value, and the optimal or near-optimal solution to the problem can be obtained.

B. System Performance Index

This section presents an optimization criterion for parameter tuning of the control system. It deals with a modification of the known ITSE. In order to influence a characteristic of the signal, it is unhandy to add a special term to the ITSE for increasing the selection pressure in evolution strategies. An evident possibility is to divide the integral criterion in special error sections for each characteristic value. The addition of the integral section in the time horizon is straightforward and always has the same unit, because one characteristic value stands on a part of the error signal.

The aim of control optimization is to determine the parameters of the controller in order to minimize the summation of the absolute value of error $|e(t)|$ and the squared control force $u^2(t)$, as shown in Fig. 3. For demonstration ease, a subsystem of the TRMS is presented and all explanations are given for a single PID. The general form and an often used performance index are

$$I = \int_0^T \psi\,(\,r\,(t)\,,\quad e\,(t)\,,\quad u\,(t)\,,\quad y\,(t)\,,t\,)\;dt \tag{28}$$

$$\psi = \psi_{\text{ITSE}}\,(\,e\,(t)\,,\quad u\,(t)\,,t\,) = T^n\,\left(|e(t)| + k_u u^2\,(t)\right)$$
$$n > 1, \qquad k_u > 0 \tag{29}$$

where n is the power of the index T and k_u is a constant weighting factor to the control force. In this correspondence, a straightforward valuation of a signal $e\,(t)$ has to divide the signal space into time sections within which the characteristic values are defined. The more characteristic values that are considered, the more sections have to be divided. There are four sections or characteristic values that are used in this correspondence. The following equation represents the system performance index:

$$I_{\text{PITSE}} = T^n \begin{bmatrix} \text{Overshoot I} \\ +2(\text{Overshoot II}) \\ +r\text{Time} \\ +ss\text{Time} \end{bmatrix} + \text{control}. \tag{30}$$

Fig. 4 shows the definition of different sections. The rTime is evaluated as the area between 0.9 $r\,(t)$ and 0 from $t = 0$ to t_r, where t_r is the time when $y(t)$ is 0.9 $r(t)$. The section may also include the undershoot area. Overshoot I and Overshoot II are the areas between the overshoot of $y(t)$ and $1.1r(t)$ and the overshoot of $y(t)$ and $1.5r(t)$, respectively. The Overshoot II section is double evaluated in order to give more selection pressure, which can smooth the fitness landscape. This leads to a faster convergence by increasing the section pressure or weighting value. The *ss*Time is the sum of the areas between $y(t)$ and $r(t)$ in the steady-state time section. The *control* is the integral of the squared control force to the TRMS. The following equation represents the system performance index shown in Fig. 4:

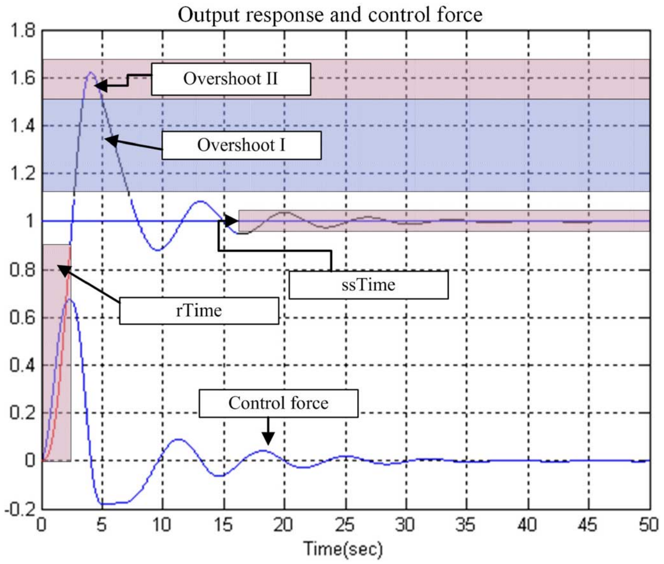


Fig. 4. Definition of performance index.

$$I_{\text{PITSE}}$$
$$= T^2 \left(\begin{array}{c} \displaystyle\int_{t_1}^{t_2} (y\,(t) - 1.1r\,(t))dt\,\Big|_{M_p\,\geq y(t)\geq 1.1r(t)} \\ \displaystyle +2\int_{t_3}^{t_4} (y\,(t) - 1.5r(t))dt\,\Big|_{M_p\,\geq\,y(t)\,\geq 1.5r(t)} \\ \displaystyle +\int_0^{t_r} 0.9r\,(t)\,dt\,\Big|_{0.9r(t)\,\geq\,y(t)} \\ \displaystyle +\int_{t_{ss}}^T |y\,(t) - r\,(t)|dt\,\Big|_{t\,\geq\,\text{steady-state time},|e(t)|\leq 0.05r(t)} \end{array} \right)$$
$$+\int_0^T u^2\,(t)dt \tag{31}$$

where t_1 and t_3 are the times when $y(t)$ first reaches $1.1r(t)$ and $1.5r(t)$, respectively, and t_2 and t_4 are the second times when $y(t)$ reaches $1.1r(t)$ and $1.5r(t)$, respectively; t_r is the rise time; t_{ss} is the steady-state time; M_p is the maximal overshoot of $y(t)$; and T is the total running time. The fitness function in RGA is

$$F = \frac{A}{I_{\text{PITSE}}} \tag{32}$$

where A is a constant. In order to enhance the differences between chromosomes, A is usually assigned to a large positive value. Scaling mechanism or rank-based fitness scheme may also be used to select the parameter A.

IV. Determination of Initial Search Range for GA

In the GA process, the optimal searched result depends on the initial search range. Without any constraint, the search range of the PID control parameters is infinity. The number of combinations of the controller's parameters is also innumerable. For a wide range with multiple parameters, the searching process is time-consuming, which is the disadvantage of the GA application. In this correspondence, we utilize an NCD technique to do the presearch. The NCD can provide a narrow initial search range for the RGA. The NCD is a kind of design work for a control system whose controlled plants may be linear or nonlinear, and the controller may be made up of linear, nonlinear, or other nonlinear components. A control system can be represented at the time domain, such as the time curve of a motor's running speed. Therefore, it is very common to observe the system's response from the time domain. It can be performed or realized after the completion of the design and analyses. Software simulation assisted by the computer is an indispensability step of the design process. Since the TRMS final result is represented at the environment of the time domain, we use the boundary range to surround the expected response and expect that some signal will be varying at the special time domain as per the designer's expectation.

The most used specifications of time-domain response are rise time, settling time, peak time, maximum overshoot, and steady-state error. The NCD optimization uses the method of optimization from math to search the result that could be the local optimum or the global optimum. The optimization procedure is designed according to the time-domain specification boundary condition, which is prescribed by the designer. The procedure searches whether or not the values of those listed tunable variables are in the boundary. Even though it is not complete in the boundary, it does not mean that the result cannot be accepted; the key is how the design is selected. Once the NCD technique in the system's model is added, the time-domain specification can be put in the time-domain response curve. After adding the tunable variables and uncertainty variables, we can execute the NCD optimization process.

A general problem of a parametric optimization is as follows [28]:

$$\begin{aligned}&\text{minimize } f(x), \quad x \in \Re^n\\&\phi_i(x) = 0, \quad i = 1, \ldots, m_e\\&\text{subject to: } \phi_i(x) \le 0, \qquad i = m_e + 1, \ldots, m\\&x \in [x_l, x_u]\end{aligned} \tag{33}$$

where x is the design parameter vector, $f(x)$ is the objective function, $\phi(x)$ is the constraint function, and x_l and x_u are the lower boundary and the upper boundary of $\boldsymbol{x}$, respectively. Besides the number of constraints and design variable of the problem, the property of the objective function will also influence the efficiency and accuracy of the solution. When the objective function and constraint function are both linear, the optimization problem is called a linear programming (LP) problem. Quadratic programming (QP) is used to denote the quadratic object function of minimization or maximization and the linear constraint function. When the objective function and constraint function are both nonlinear, this problem is called nonlinear programming (NP), and the solution process is more difficult than the two mentioned before. Usually, we must use the iterative procedure to find the searching direction. It is usually managed through solving LP, QP, or the subsystem of unconstraint function.

A. Unconstrained Optimization

For the unconstraint condition, methods for optimization can be categorized in terms of the derivative information that is, or is not, used. When solving the optimization problem that is nonlinear or discontinuous, function evaluations are preferable search methods. For solving the minimum value of a function that is continuous in its first derivative, gradient methods are most used. Gradient methods use the function's slope to decide the searching direction for optimal value. Among them, the simplest method is the steepest descent method whose searching direction is $-\nabla f(x)$, which is the gradient of the objective function. The disadvantages of this method are difficulty in configuring the step size, risk of locking in a local optimum point, and possible unstable behavior.

Among the methods that need the gradient information for solving the optimization problem, quasi-Newton methods are most preferable. These methods generate curvature information in the process of each iteration and formulate a quadratic model problem as follows:

$$\min_x \frac{1}{2} x^T H x + c^T x + b \tag{34}$$

where H is the Hessian matrix, c is a constant vector, and b is a constant scalar. The optimal solution for this problem occurs at the position where its first derivative with respect to x is zero

$$\nabla f(x^*) = Hx^* + c = 0. \tag{35}$$

And the optimal solution x^* can be written as

$$x^* = -H^{-1}c. \tag{36}$$

The updating rule for the Hessian matrix is as follows:

$$H_{k+1} = H_k + \frac{q_k q_k^T}{q_k^T s_k} - \frac{H_k^T s_k^T s_k H_k}{s_k^T H_k s_k} \tag{37}$$

where $s_k = x_{k+1} - x_k$ and $q_k = \nabla f(x_{k+1}) - \nabla f(x_k)$.

For the initial condition, H_0 can be set to the identity matrix I. At the kth iteration, its searching direction is

$$d = -H_k^{-1} \nabla f(x_k). \tag{38}$$

The quasi-Newton method has two steps: determination of the search direction and line procedure. The first step has been given earlier. In the second step, if we can obtain the gradient information, the presetting condition adopts the cubic interpolation method; if the gradient information cannot be obtained, it adopts the quadratic interpolation method. To minimize the function $f(x)$, it must satisfy the following condition at each iteration:

$$f(x_{k+1}) < f(x_k) \tag{39}$$

and

$$x_{k+1} = x_k + \alpha_k d \tag{40}$$

where α_k is a scalar step length parameter at the kth iteration.

If the step length does not satisfy (39) then the step can be reduced by half to form a new α_{k+1} until it satisfies (39). But the disadvantage is that the procedure is very slow. To overcome this problem, we use the gradient information and function value combined with cubic interpolation to estimate the step length. Cubic interpolation is used when gradient information or more than three function evaluations have been obtained. For any two points x_1 and x_2 whose gradients are $\nabla f(x_1)$ and $\nabla f(x_2)$, and function values are $f(x_1)$ and $f(x_2)$, respectively, the minimum can be obtained by

$$x_{k+1} = x_2 - (x_2 - x_1) \frac{\nabla f(x_2) + \beta_2 - \beta_1}{\nabla f(x_2) - \nabla f(x_1) + 2\beta_2} \tag{41}$$

where $\beta_1 = \nabla f(x_1) + \nabla f(x_2) - 3\,[f(x_1) - f(x_2)]/(x_1 - x_2)$ and $\beta_2 = \sqrt{\beta_1^2 - \nabla f(x_1) \nabla f(x_2)}$.

If the gradient information cannot be obtained, then the quadratic interpolation method is applied. If the function values of three different points $\{x_1, x_2, x_3\}$ are $\{f(x_1), f(x_2), f(x_3)\}$, it satisfies $f(x_2) < f(x_1)$ and $f(x_2) < f(x_3)$. Then, the minimum can be obtained as

$$x_{k+1} = \frac{1}{2} \frac{\beta_{23} f(x_1) + \beta_{31} f(x_2) + \beta_{12} f(x_3)}{\gamma_{23} f(x_1) + \gamma_{31} f(x_2) + \gamma_{12} f(x_3)} \tag{42}$$

where $\beta_{ij} = x_i^2 - x_j^2$ and $\gamma_{ij} = x_i - x_j$.

B. Constrained Optimization

For the optimization problem with the constraint condition, the most common method is to transform the problem into an easier subproblem that can then be solved and used as the basis of an iterative process, but this method is inefficient. Kuhn–Tucker (KT) equations are the type used for the optimization problem with the constraint condition. The KT equations are described as follows:

$$\begin{aligned} f(x^*) + \sum_{i=1}^{m} \lambda_i^* \nabla\phi(x^*) &= 0 \\ \nabla\phi_i(x^*) = 0, &\qquad i = 1,\ldots,m_e \\ \lambda_i^* \geq 0, &\qquad i = m_e + 1,\ldots,m \end{aligned} \tag{43}$$

where $f\ (x)$ and $\phi_i(x)$ are convex functions and λ_i are the Lagrange multipliers. The Lagrange multipliers must be selected carefully to agree with the earlier equation. The solution for KT equations is the basis of many nonlinear algorithms; it also calculates the Lagrange multipliers' value directly. Adopting the updating program can ensure the convergence of the solution of the KT equations. This method is called sequential quadratic programming (SQP). The SQP method is widely used in many optimization problems. In each iterative process, it adopts the similar quasi-Newton method to update the Hessian matrix of the Lagrangian function and uses H to produce the QP subproblem to get the searching direction. The QP subproblem can be obtained by linearizing the constraints.

$$\begin{aligned} &\min_{d \in R^n} \frac{1}{2} d^T H_k d + \nabla f(x_k)^T d \\ &\nabla\phi_i(x_k)^T d + \phi_i(x_k) = 0, \qquad i = 1,\ldots,m_e \\ &\nabla\phi_i(x_k)^T d + \phi_i(x_k) \leq 0, \qquad i = m_e + 1,\ldots,m. \end{aligned} \tag{44}$$

The updating rule for the Hessian matrix is described as follows:

$$\begin{aligned} H_{k+1} &= H_k + \frac{q_k q_k^T}{q_k^T s_k} - \frac{H_k^T H_k}{s_k^T H_k s_k} \\ s_k &= x_{k+1} - x_k \\ q_k &= \nabla f\left(x_{k+1}\right) + \sum_{i=1}^{n} \lambda_i \nabla\phi_i\left(x_{k+1}\right) \\ &\quad - \left(\nabla f\left(x_k\right) + \sum_{i=1}^{n} \lambda_i \nabla\phi_i\left(x_k\right)\right). \end{aligned} \tag{45}$$

If the initial value of H is positive definite and $q_k^T s_k$ is positive, the Hessian matrix could remain positive definite. If $q_k^T s_k$ is not positive, then q_k should be modified by reducing the most negative element of $q_k^T s_k$ to half of its value till $q_k^T s_k \geq$1E-5. If it also cannot put $q_k^T s_k > 0$, then apply the following steps:

$$\begin{aligned} q_k &= q_k + w_v \\ v_i &= \nabla\phi_i(x_{k+1})\phi_i(x_{k+1}) - \nabla\phi_i(x_k)\phi_i(x_k), \\ &\quad \text{if } \left(q_k\right)_i w < 0 \quad \text{and} \quad \left(q_k\right)_i \left(s_k\right)_i < 0 \\ v_i &= 0, \qquad \text{otherwise} \end{aligned} \tag{46}$$

where v is a constant vector and w is a constant scalar. The direction d of search for the SQP is obtained by QP. The vector d_k, which is produced during the solving of the QP subproblem, are used to carry on the next iterative process, which is described as

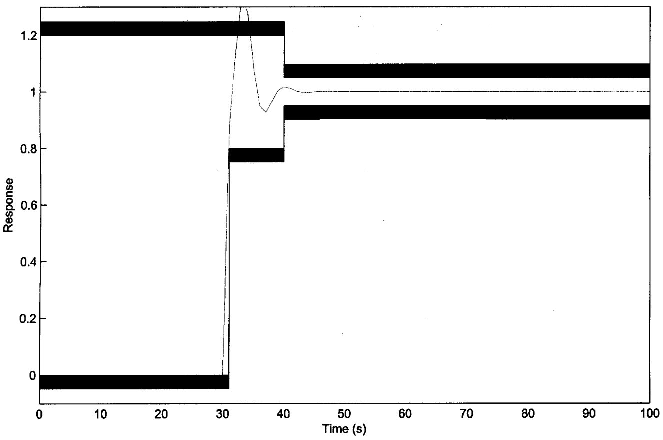


Fig. 5. Position response and specification boundary of the horizontal part.

$$x_{k+1} = x_k + \alpha_k d_k \tag{47}$$

where α_k is the step value that satisfies for decreasing a merit function. The merit function is defined as

$$\Gamma\left(x\right) = f\left(x\right) + \sum_{i=1}^{m_e} r_i \phi_i\left(x\right) + \sum_{i=m_e+1}^{m} r_i \max\left\{0, \phi_i\left(x\right)\right\}. \tag{48}$$

The penalty parameter is defined as

$$r_i = \left(r_{k+1}\right)_i = \max_i \left\{\lambda_i, \frac{1}{2}\left(\left(r_k\right)_i + \lambda_i\right)\right\}, \quad i = 1,\ldots,m. \tag{49}$$

The initial value is set as

$$r_k = \frac{\left\|\nabla f\left(x\right)\right\|}{\left\|\nabla\phi_i\left(x\right)\right\|}. \tag{50}$$

C. NCD Simulation Procedures

Add position error and control output of the NCD outport blockset to the 1 DOF PID control system. The 1 DOF PID control system represents the horizontal part of the NCD system or the vertical part of the NCD system. The optimization steps for the PID control system in the horizontal direction are as follows.

1) Put the time-domain specifications into the time-domain response curve. The time-domain specifications include position response and PID control output.

   Position response, described as thick lines in Fig. 5:

   a) overshoot less than 20%;
   b) rise time less than 1 s;
   c) settling time less than 10 s.

   Response of PID control output, described as thick lines in Fig. 6:

   a) overshoot less than 40%;
   b) rise time less than 1 s;
   c) settling time less than 10 s.

   Note that this preliminary stage of the algorithm only decides the range of the parameters, so, if large overshoots are allowed, large ranges could be selected and an appropriate setting of search space could be provided for the genetic procedure.

2) Input the initial design parameter vector x, which is the vector of the control gains K_{Phh}, K_{Ihh}, and K_{Dhh} of the PID controller.

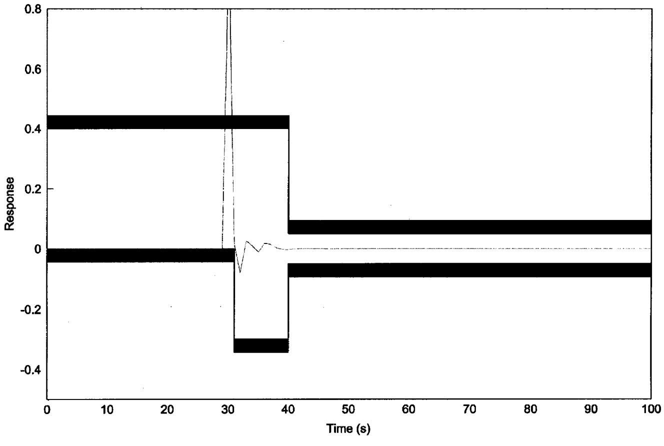


Fig. 6. PID control output and specification boundary of the horizontal part.

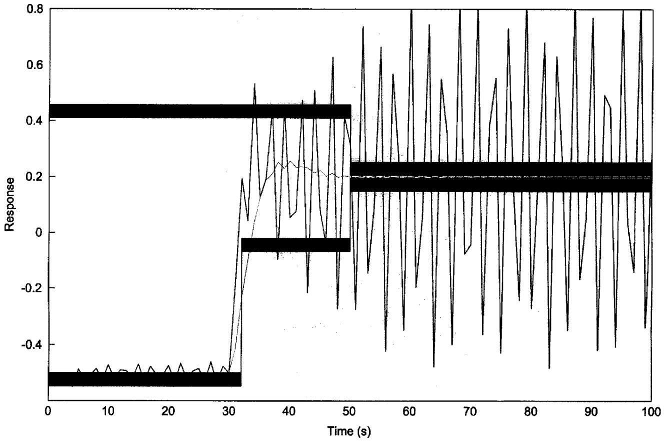


Fig. 7. Position response and specification boundary of the vertical part.

3) Start the NCD optimization procedure and minimize the objective function$f(x)$.
4) Display the optimization result in the time-domain response curve and output the optimal solution x^*. Figs. 5 and 6 denote the responses of the initial settings.

The optimization steps for the PID control system of the vertical direction are as follows.

1) Put the time-domain specifications into the time-domain response curve. The specifications include

   Position response, described as thick lines in Fig. 7:

   a) overshoot less than 30%;
   b) rise time less than 2 s;
   c) settling time less than 20 s.

   Response of PID control output, described as thick lines in Fig. 8:

   a) overshoot less than 66%;
   b) rise time less than 2 s;
   c) settling time less than 30 s.
2) Input the initial design parameter vector x, which is the vector of the control gains K_{Pvv}, K_{Ivv}, and K_{Dvv} of the PID controller.
3) Start the NCD optimization procedure and minimize the objective function$f(x)$.
4) Display the optimization result in the time-domain response curve and output the optimal solution x^*. In Figs. 7 and 8, the oscillation curves that cross over boundary lines are the responses of the initial settings. The curves that are within the boundary lines are the final searched results.

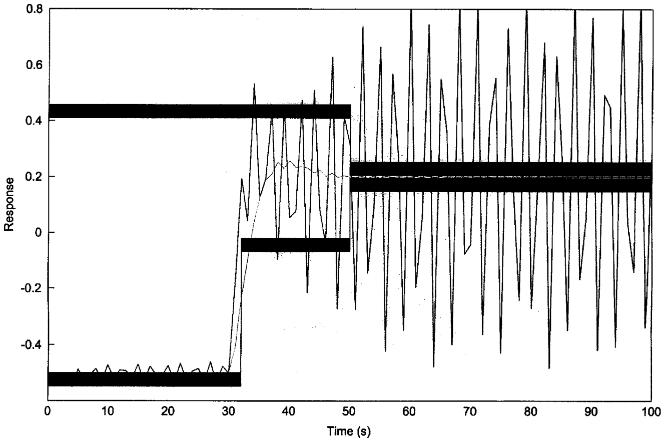


Fig. 8. PID control output and specification boundary of the vertical part.

The setting for the searching range is very important for GA applications. The controlled plant of this research is a nonlinear MIMO system. The control gains of the PID controller have infinite combinations and the searching range is also infinite. In this correspondence, we use the PID control gains from the Gauss–Seidel minimization [1] as a reference. Then, we set the initial range to [0, 11] for every control parameter. Through the NCD presearching procedure, the optimal K_P, K_I, and K_D are around 1. Thus, the range [0, 2] is chosen as the initial range for GA process. The PID control parameters K_P, K_I, and K_D are then fine-tuned by the proposed RGA.

V. Parameters Optimization Using RGA for TRMS Control

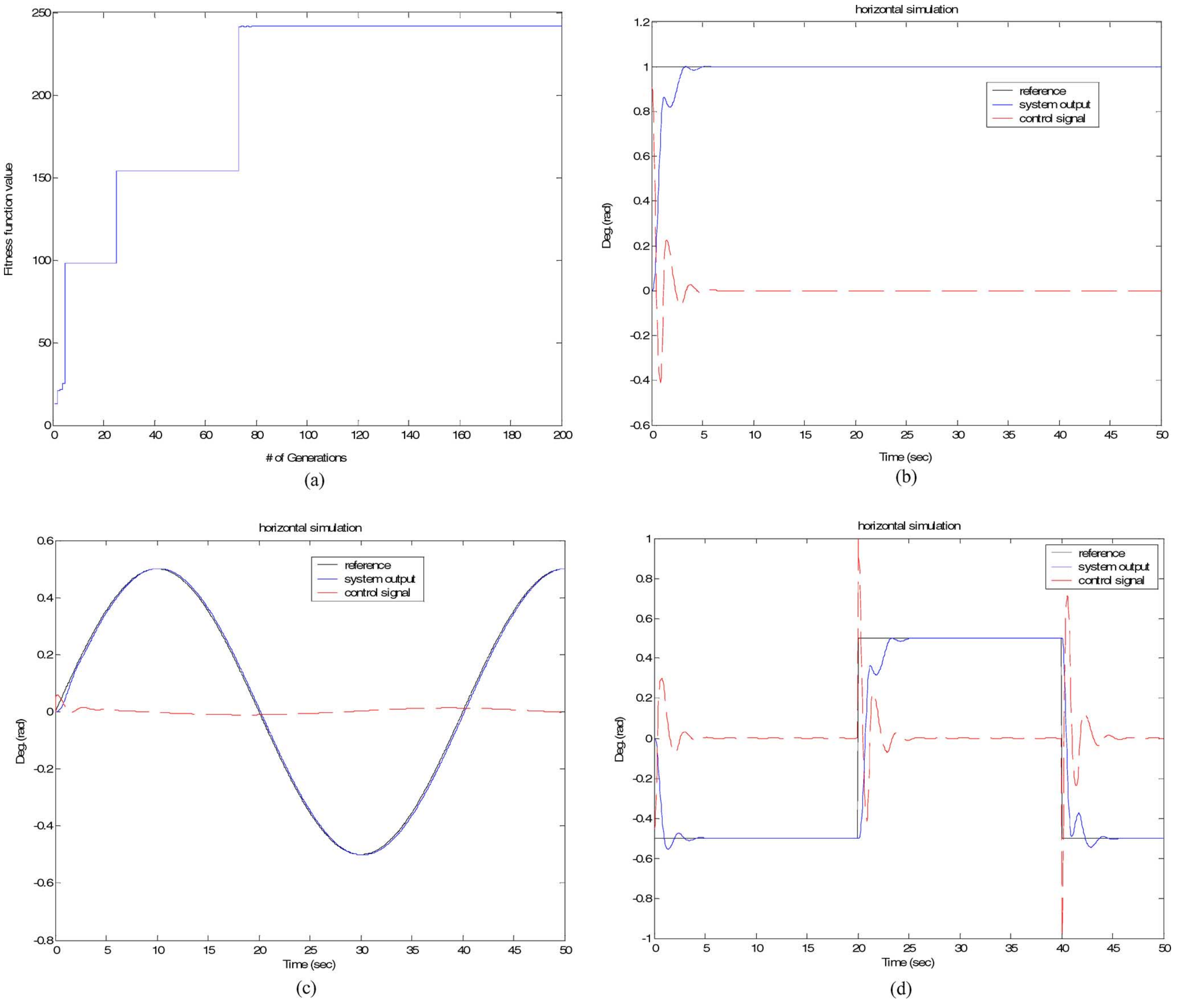
In this section, all parameters of the PID controllers will be optimized by the RGA with the system performance index. The final purpose is to build a control structure in 2 DOF. There are 12 parameters that need to be searched. In order to reach this goal, the parameters in the decoupled system are tuned first and then the results are combined for the 2 DOF system.

A. *Parameters Optimizing for PID Controller in the Horizontal Plane*

The control system contains a PID controller and a horizontal part of the TRMS. There are ten individuals each with three genes in the initial population. The crossover rate is 0.8 and the mutation rate is 0.025. The parameters to be tuned are K_{Phh}, K_{Ihh}, and K_{Dhh}, within the range [0, 2], and the number of generations is 200. The computer simulation is from 0 to 50 s, with a sampling time of 0.05 s. For step response, the reference input is 1.0 rad, with the initial condition of 0.0 rad. The fitness function is

$$F = \frac{30\,000}{I_{\mathrm{PITSE}}}. \tag{51}$$

The simulations of tracking the desired trajectories are given by two different reference inputs: sine wave with amplitude 0.5 rad and square wave with amplitude 0.5 rad. Fifty runs are made and the solutions are very similar. One set of simulations is shown in Fig. 9. Table I shows the comparison of conventional PID, conventional RGA (C-RGA), and the modified RGA (M-RGA) of this correspondence. The *error* (total absolute value of errors from 0 to 50 s) of the step response in [12] is



(a)

(b)

(c)

(d)

Fig. 9. Fitness value and system outputs of 1 DOF control by M-RGA in horizontal plane. (a) Fitness value in horizontal plane. (b) Step response in horizontal plane. (c) Sine wave response in horizontal plane. (d) Square wave response in horizontal plane.

TABLE I
SIMULATION RESULTS OF 1 DOF CONTROL IN HORIZONTAL PLANE

reference	*error*			*control*		
	PID	C-RGA	M-RGA	PID	C-RGA	M-RGA
Step	26.21	20.13	20.28	17.83	15.98	14.84
Sine	11.58	14.46	8.29	12.47	12.29	7.62
Square	86.36	63.11	38.07	78.66	58.83	37.71

20.1336, and the *control* (total absolute value of control forces from 0 to 50 s) is 15.9887, whereas the *error* and the *control* from using the conventional PID are 26.21 and 17.83, respectively [11]. Compared with the conventional RGA technique (C-RGA) [12], the *error* is almost the same but the *control* reduces about 7%.

B. Parameters Optimizing for PID Controller in the Vertical Plane

In the initial population, there are ten individuals with three genes each. The crossover rate is 0.8, while the mutation rate is 0.025. The parameters to be tuned are K_{Pvv}, K_{Ivv}, and K_{Dvv}, within the range

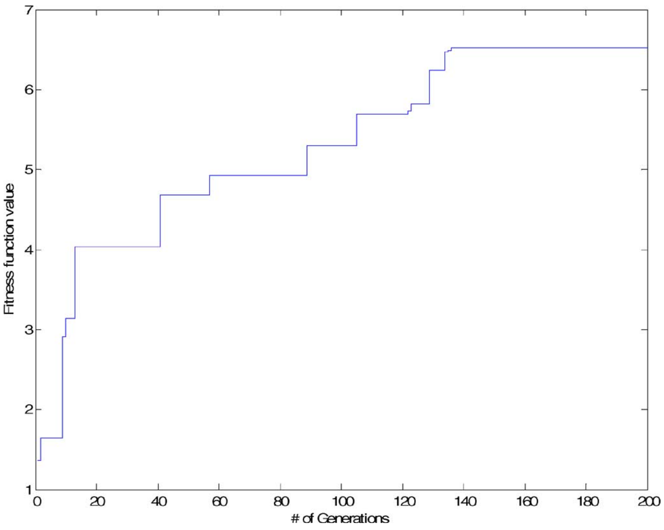


Fig. 10. Fitness value in vertical plane.

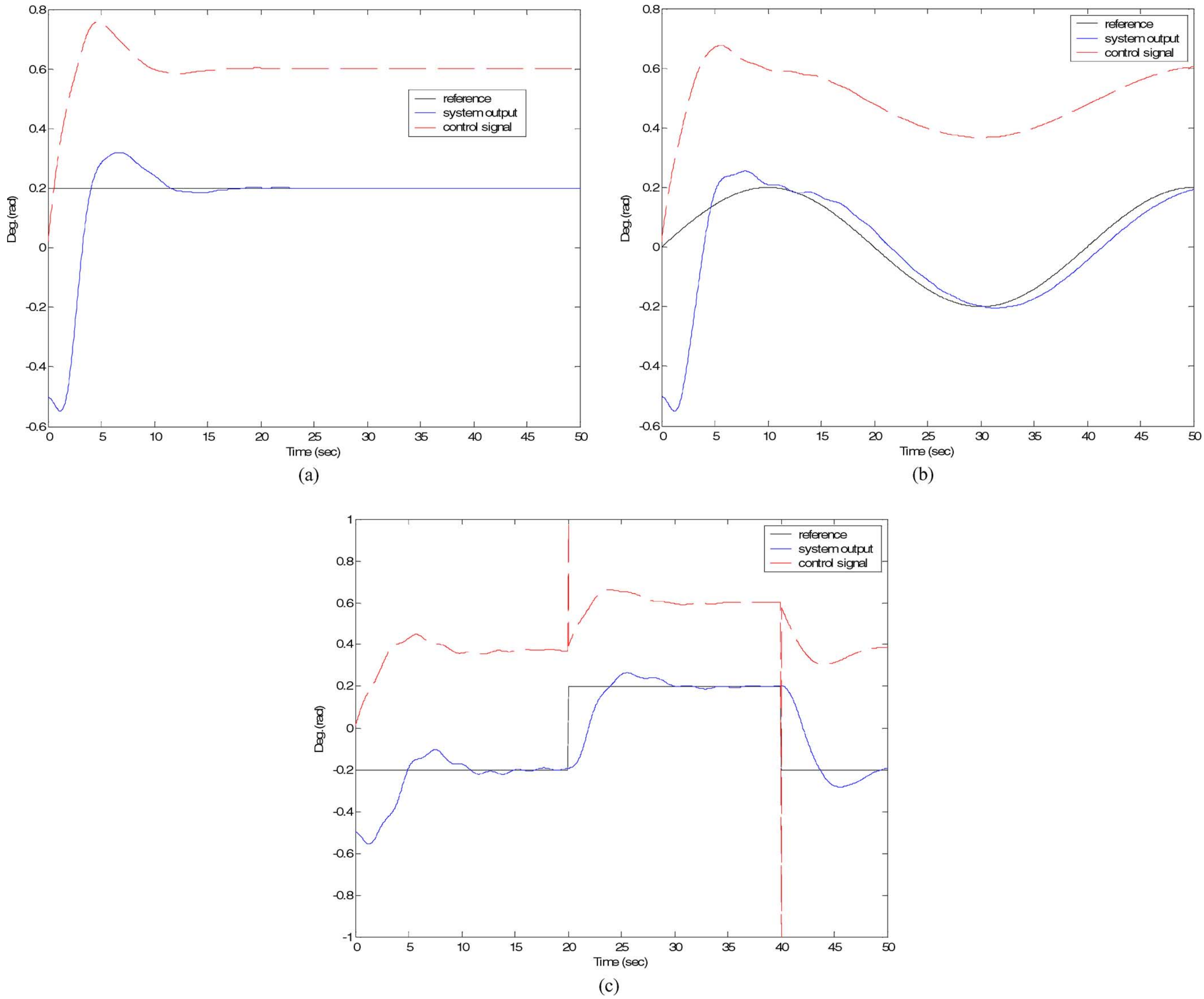


Fig. 11. System outputs of 1 DOF PID control by M-RGA in vertical plane. (a) Step response in vertical plane. (b) Sine wave response in vertical plane. (c) Square wave response in vertical plane.

TABLE II
SIMULATION RESULTS OF 1 DOF CONTROL IN VERTICAL PLANE

reference	*error*			*control*		
	PID	C-RGA	M-RGA	PID	C-RGA	M-RGA
Step	67.84	50.07	53.60	687.53	605.95	599.49
Sine	72.69	72.43	66.47	1185.74	1178.00	492.77
Square	137.27	106.91	75.40	1234.81	1213.9	457.34

[0, 2], and the number of generations is 200. For the step response, the reference input is 0.2 rad with an initial condition of −0.5 rad for gravity compensation. The simulations in tracking the specific paths are given with two different reference inputs: a sine wave with amplitude 0.2 rad and a square wave with the same amplitude and frequency of 0.025 Hz. The simulations are shown in Figs. 10 and 11, and Table II.

The simulation results in [12] show that the step input *error* is 50.07 and the *control* is 605.95, whereas the *error* and the *control* from using the conventional PID are 67.84 and 687.53, respectively [11]. Compared with the step response in [12], as shown in Fig. 12, the output oscillation is eliminated and the maximum overshoot and control force are reduced. In the experiments of tracking a desired trajectory, the proposed controller reduced both the *error* and the oscillation.

C. Parameters Optimizing for PID Controller in Cross-Coupled Condition

The control system has four PID controllers in the cross-coupled condition. There are 12 parameters in the controller structure that need to be optimized, shown in Table III. In order to test the setpoint and tracking control, the computer simulation is given with three different reference signals:

1) step input with 1 rad in the horizontal plane and step input with 0.2 rad in the vertical plane;
2) sine wave with amplitude 0.5 rad in 0.025 Hz for the horizontal plane and sine wave with amplitude 0.2 rad in 0.025 Hz for the vertical plane;
3) square wave with amplitude 0.5 rad in 0.025 Hz for the horizontal plane and square wave with amplitude 0.2 rad in 0.025 Hz for the vertical plane.

In the procedure of the RGA, the crossover rate is 0.8 and the mutation rate is 0.025 with 500 generations. There are 12 genes in an

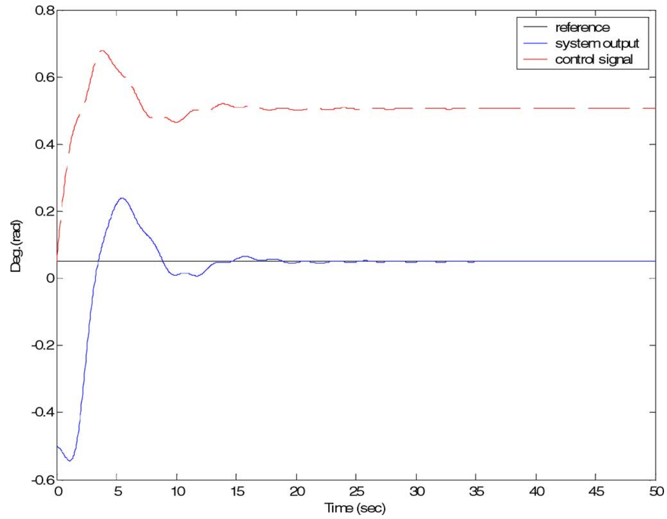


Fig. 12. Step response of vertical plane in [12].

TABLE III
PARAMETERS IN CROSS-COUPLED CONTROL

Horizontal to Horizontal		
K_{Phh}	K_{Ihh}	K_{Dhh}
Horizontal to Vertical		
K_{Phv}	K_{Ihv}	K_{Dhv}
Vertical to Horizontal		
K_{Pvh}	K_{Ivh}	K_{Dvh}
Vertical to Vertical		
K_{Pvv}	K_{Ivv}	K_{Dvv}

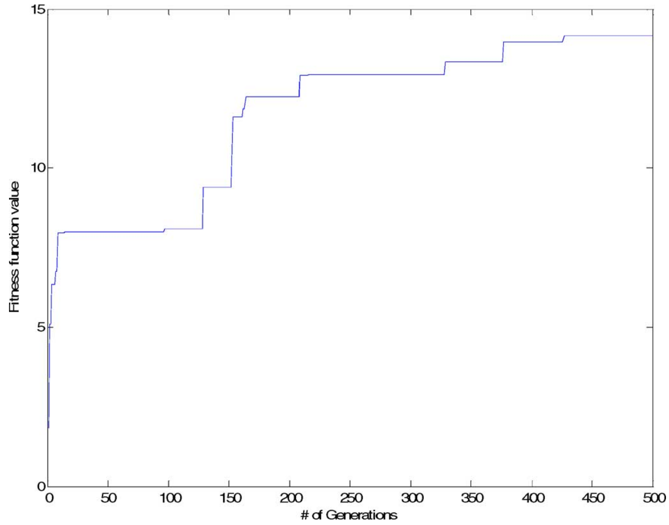


Fig. 13. Fitness value of cross-coupled M-RGA PID control.

individual generated within the range [0, 2]. Each gene represents one control parameter. The fitness function is shown in (51). The computer simulation results are shown in Figs. 13–16. For the sine wave response, the settling time is 10 s in both horizontal and vertical planes, while the settling time is 15 s by using the sliding mode control [9].

Table IV shows the results of the cross-coupled control of the TRMS. The *error* of the step response in [12] is 69.09 in the horizontal plane and 34.92 in the vertical plane. The proposed controller reduces the *error* by 21% in the horizontal plane and 20% in the vertical plane and improves the output performance in 2 DOF. Fig. 17 shows the step response in [12].

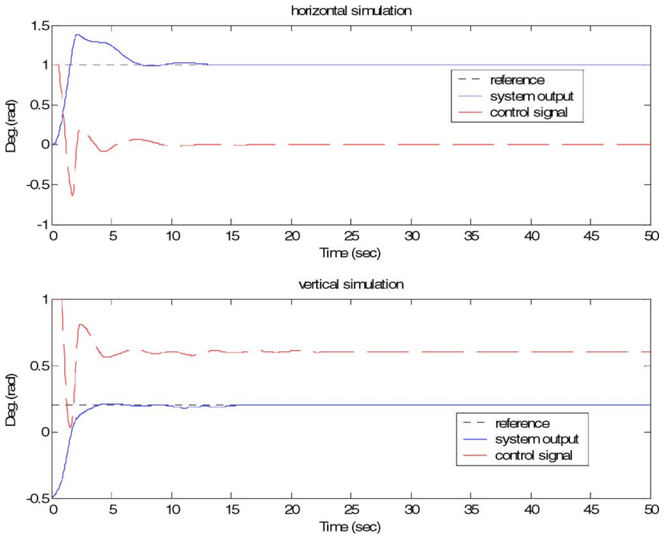


Fig. 14. Step response of cross-coupled control.

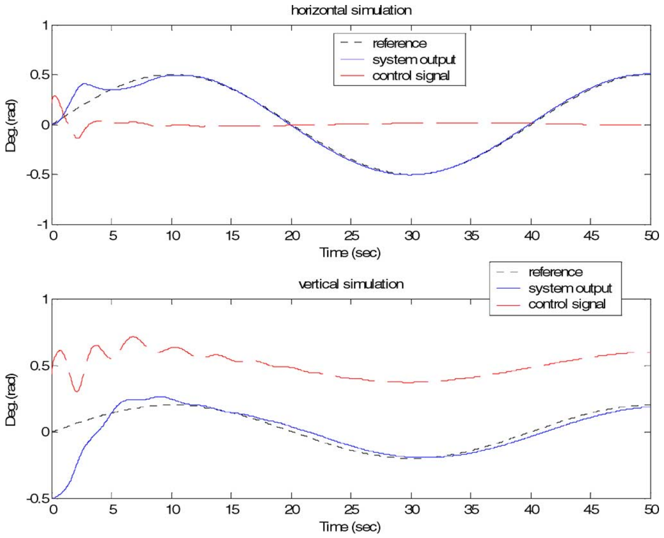


Fig. 15. Sine wave response of cross-coupled control.

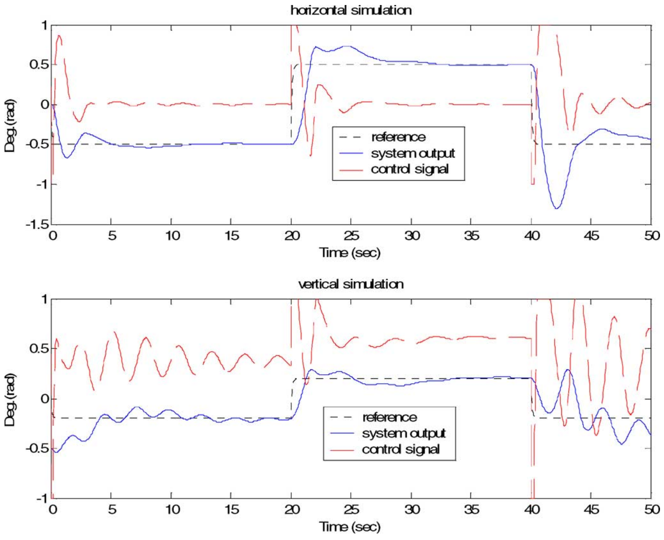


Fig. 16. Square wave response of cross-coupled control.

TABLE IV
SIMULATION RESULTS OF CROSS-COUPLED CONTROL

Reference		*error*			*control*		
		PID	C-RGA	M-RGA	PID	C-RGA	M-RGA
Step	H	81.2	69.09	54.52	76.71	51.34	40.47
	V	40.11	34.92	27.46	812.36	701.23	617.10
Sine	H	23.21	19.33	20.92	27.41	20.12	18.93
	V	65.74	51.78	52.61	611.70	500.2	501.78
Square	H	150.22	141.52	134.03	202	171.28	165.32
	V	112.85	96.36	90.21	656.37	591.65	551.59

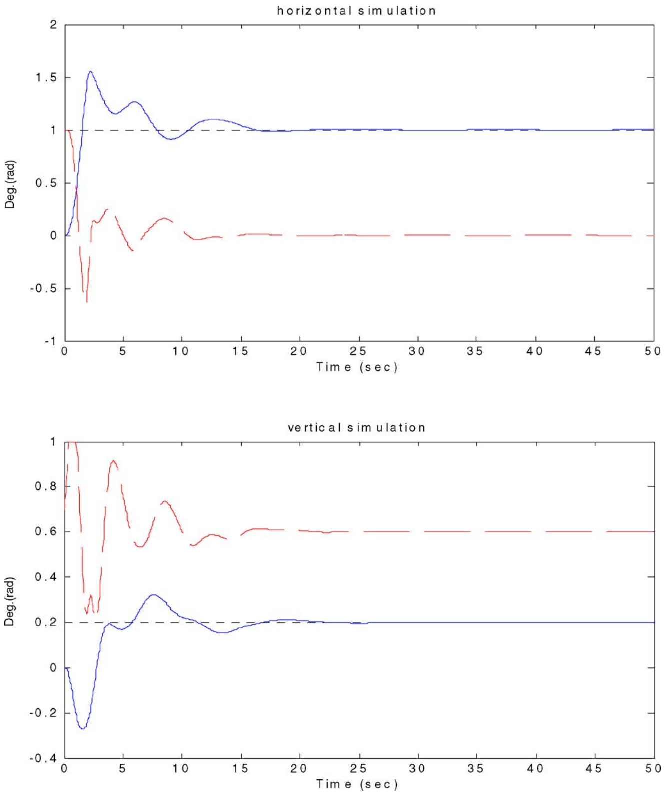


Fig. 17. Step response in [12].

VI. CONCLUSION

In order to control the TRMS in 2 DOF, we employ a modified RGA for tuning the parameters of the PID controller. The simulation results show that the new control scheme can make the TRMS reach a specific position and track a desired path more efficiently. The total error in setpoint control and trajectory tracking of the sine wave and square wave are both reduced. Simulations are done by an Intel P4/3.0 personal computer. The average computational cost of the modified RGA for cross-coupled simulation is 51 s, over 50 runs, and each run has 500 generations. The searched solution of the RGA is related to initial search range, initial population, fitness function, and other factors. In this correspondence, we develop a new procedure for setting the initial search range, and propose a fitness function with a system performance index in solving large number of parameters simultaneously. The control parameters in the cross-coupled PID controller converge to satisfactory solutions quickly. Simulations show that the proposed PID control using NCD and RGA has better performance than conventional PID and GA-PID controllers.

REFERENCES

[1] *TRMS 33-220 User Manual*, 3-000M5, Feedback Company, E. Sussex, U.K. 1998.

[2] S. M. Ahmad, M. H. Shaheed, A. J. Chipperfield, and M. O. Tokhi, "Nonlinear modelling of a TRMS using radial basis function networks," in *Proc. IEEE Nat. Aerosp. Electron. Conf.*, Dayton, OH, 2000, pp. 313–320.

[3] S. M. Ahmad, A. J. Chipperfield, and M. O. Tokhi, "Dynamic modeling and optimal control of a TRMS," in *Proc. IEEE Nat. Aerosp. Electron. Conf.*, Dayton, OH, 2000, pp. 391–398.

[4] A. Odwyer, *Handbook of PI and PID Controller Tuning Rules*. London, U.K.: Imperial College Press, 2003.

[5] B. C. Kuo, *Automatic Control Systems*, 6th ed. Englewood Cliffs, NJ: Prentice-Hall, 1995.

[6] J. G. Ziegler and N. B. Nichols, "Optimum settings for automatic controllers," *Trans. ASME*, vol. 64, pp. 759–768, Nov. 1942.

[7] R. A. Krohling, H. Jaschek, and J. P. Rey, "Designing PI/PID controllers for a motion control system based on genetic algorithms," in *Proc. 12th IEEE Int. Symp. Intell. Control*, Istanbul, Turkey, 1997, pp. 125–130.

[8] R. A. Krohling and J. P. Rey, "Design of optimal disturbance rejection PID controllers using genetic algorithms," *IEEE Trans. Evol.*, vol. 5, no. 1, pp. 78–82, Feb. 2001.

[9] J. P. Su, C. Y. Liang, and H. M. Chen, "Robust control of a class of nonlinear systems and its application to a TRMS," in *Proc. IEEE Int. Conf. Ind. Technol.*, Bangkok, Thailand, 2002, pp. 1272–1277.

[10] B. U. Islam, N. Ahmed, D. L. Bhatti, and S. Khan, "Controller design using fuzzy logic for a TRMS," in *Proc. IEEE Int. Multi Topic Conf.*, Islamabad, Pakistan, 2003, pp. 264–268.

[11] M. T. Huang and J. G. Juang, "Application of GA and PID control to nonlinear TRMS," in *Proc. Artif. Intell. Appl. Conf.*, Taichung, Taiwan, 2002, pp. 734–739.

[12] C. Y. Tsai, M. T. Huang, and J. G. Juang, "Application of real-type GA to TRMS position control," in *Proc. Autom. Control Conf.*, Jung-Li, Taiwan, 2003, pp. 1267–1273.

[13] J. H. Fang and J. G. Juang, "Analysis of optimal fitness function for TRMS parameter searching and its implementation on FPGA," presened at the Artif. Intell. Appl. Conf., Taipei, Taiwan, 2004, Paper FP4-1.

[14] W. K. Liu, J. H. Fan, and J. G. Juang, "Application of system-performance-index based genetic algorithm to PID controller," presened at the Artif. Intell. Appl. Conf., Taipei, Taiwan, 2004, Paper FP4-3.

[15] J. H. Holland, *Adaption in Nature and Artificial System*. Ann Arbor, MI: Univ. of Michigan Press, 1975.

[16] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA: Addison & Wesley, 1989.

[17] L. Davis, Ed., *Handbook of Genetic Algorithms*. New York, Boston: Van Nostrand Reinhold/Kluwer, 1991, pp. 299–305.

[18] K. F. Man, K. S. Tang, and S. Kwong, *Genetic Algorithms*. London, U.K.: Springer-Verlag, 1999.

[19] R. Subbu, K. Goebel, and D. K. Frederick, "Evolutionary design and optimization of aircraft engine controllers," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 35, no. 4, pp. 554–565, Nov. 2005.

[20] G. Di Fatta, F. Hoffmann, G. Lo Re, and A. Urso, "A genetic algorithm for the design of a fuzzy controller for active queue management," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 33, no. 3, pp. 313–324, Aug. 2003.

[21] H. J. Cho, K. B. Cho, and B. I. Wang, "Automatic rule generation using genetic algorithms for fuzzy-PID hybrid control," in *Proc. IEEE Int. Symp. Intell. Control*, Dearborn, MI, 1996, pp. 271–276.

[22] B. Hu, G. K. I. Mann, and G. Raymond, "New methodology for analytical and optimal design of fuzzy PID controllers," *IEEE Trans. Fuzzy Syst.*, vol. 7, no. 5, pp. 521–539, Oct. 1999.

[23] Y. S. Zhou and L.Y. Lai, "Optimal design for fuzzy controllers by genetic algorithms," *IEEE Trans. Ind. Appl.*, vol. 36, no. 1, pp. 93–97, Jan./Feb. 2000.

[24] K. Belarbi and F. Titel, "Genetic algorithm for the design of a class of fuzzy controllers: An alternative," *IEEE Trans. Fuzzy Syst.*, vol. 8, no. 3, pp. 398–405, Aug. 2000.

[25] W. K. Liu, "Applications of hybrid fuzzy-GA controller and FPGA chip to TRMS," M. S. thesis, Dept. Commun. Guid. Eng., Nat. Taiwan Ocean Univ., Keelung, Taiwan, 2005.

[26] M. Mitchell, *An Introduction to Genetic Algorithms*. Cambridge, MA: MIT Press, 1998.

[27] A. A. Adewuya, "New methods in genetic search with real-valued chromosomes," M. S. thesis, Dept. Mech. Eng., Massachusetts Inst. Technol., Cambridge, 1996.

[28] T. Coleman, M. A. Branch, and A. Grace, *Optimization Toolbox: For Use With MATLAB*. Natick, MA: The MathWorks, Inc., 1999.