World Journal Control Science and Engineering, 2013, Vol. 1, No. 1, 15-24
Available online at http://pubs.sciepub.com/wjcse/1/1/3

DOI:10.12691/wjcse-1-1-3

Controllers Tuning through Multi-objective Non-Dominated Sorting Genetic Algorithms

CHOURAQUI Samira*, BENZATER Habiba

Department of Computer Science, University of Sciences and Technology USTO'MB, El Mnouar, Algeria
*Corresponding author: s_chouraqui@yahoo.fr

Received October 10, 2013; Revised November 02, 2013; Accepted November 05, 2013

Abstract In control system design there are often a number of design objectives to be considered. The objectives are sometimes connecting and no design exists which can be considered best with respect to all objectives. Hence, there is an inevitable tradeoff between design objectives, for example, between and output performance objective and stability robustness. These considerations have led to the study of multi objective optimization methods for control systems. In this paper a multi-objective Non-Dominated sorting genetic Algorithms NSGA-II is used to tuning of Proportional Derivative (PD) controller of a six freedom arm manipulator PUMA560. The NSGAII algorithm searches for the controller PD gains so that the six values of Integral Absolute Error (IAE) in joint space are minimized. Simulation numerical results of multivariable PD control and convergence of the NSGA-II are presented and discussed.

Keywords: *arm manipulator, PD control, Simulink, NSGA-II*

Cite This Article: CHOURAQUI Samira, and BENZATER Habiba, "Controllers Tuning through Multi-objective Non-Dominated Sorting Genetic Algorithms." *World Journal Control Science and Engineering* 1, no. 1 (2013): 15-24. doi: 10.12691/wjcse-1-1-3.

1. Introduction

In control system design there are often a number of design objectives to be considered. The objectives are sometimes conflicting and no design exists which can be considered best with respect to all objectives. Hence, there is an inevitable tradeoff between design objectives, for example, between output performance objective and stability robustness. These considerations have led to the study of multi objective optimization methods for control systems.

Although there has been considerably more research into multi objective decision making in the field of systems engineering than into multi objective optimization design of multivariable control, more and more research in multi objective control has been carried out. The main research areas are multi objective robust control [1,2], multi objective critical control [3], multi objective eigenstructure assignment [4], multi objective PID control [5,6,7], multi objective identification [8], multi objective fault detection [9], and multi objective linear quadratic Gaussian control [10,11,12,13,14].

In this paper a we use the improved non-dominated sorting genetic algorithm (NSGA-II), which is one of the most powerful non aggregative multi-objective techniques, and able to locate the Pareto front in complex reach space. The design of PD controller is considered here and it formulated as multi-objective optimization problem where the integral absolute error is optimized simultaneously. The remainder of the paper is organized as follows:

In section 2, a general description of the multi-objective optimization algorithm is presented. Dynamic formulation of robot manipulator is presented in section 3. In section 4, NSGA-II algorithm tuning PD gains is presented. The simulation results of the model and the NSGAII algorithm are shown in section 5. A conclusion follows in section 6.

2. Multi-objective Optimization Algorithm

The optimization problem is minimized or maximized an objective function. In single-objective optimization, it is possible to determine between any given pair of solutions if one is better than the other, as result the best single solution is the goal. However, multi-objective optimization (MOO) problems in which the designer seeks to optimize simultaneously several objectives, there is usually no single optimal solution [15].

The method most commonly adopted to compare solutions in multi-objective optimization is called Pareto dominance relation which, instead of a single optimal solution, leads to a set of solutions. These solutions are called Pareto optimal solutions or non-dominated solutions. The Pareto set consists of solutions that are not dominated by any other solutions. A solution x is said to dominate y if x is better or equal to y in all attributes, and strictly better in at least one attribute [5].

Figure 1 shows a classification of the methods used to solve an MOO problem in literature given in [16].

Multi-Objective Evolutionary Optimization (MOEA) is an approach useful that offers an alternate means of solving multi-objective optimization problems compared

to classical approaches. Since evolutionary algorithms use a population based approach, they allow an efficient way to find an approximation of the whole Pareto front in a single simulation run [15].

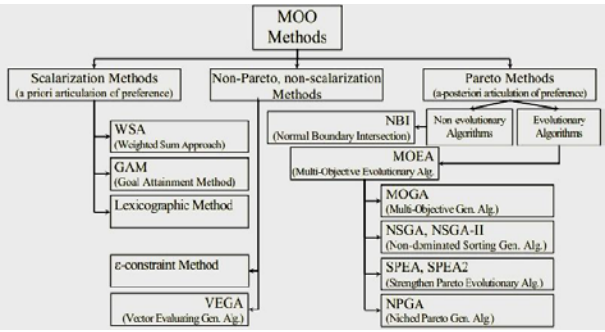

Figure 1. General Classification of MOO solving methods [16]

Today, there are many MOEAs distinguished mainly by the algorithms for the population ranking in the fitness assignment. The most important are: MOGA, NSGA-II (Non-dominated Sorting Genetic Algorithm II).The NSGA-II method is a heavily revised version of the Non-dominated Sorting Genetic Algorithm (NSGA), which was introduced in the mid 1990 [17].

3. Dynamic Model of the PUMA 560 Arm Manipulator

3.1. Presentation of the Puma 560 Arm Manipulator

The PUMA robot is connected to a 1980 series controller that has a programming language called VAL II [18]. The Unimation PUMA 560 is a PC controlled, robotic arm used frequently in industrial applications, it is a serial manipulator that has six revolute joints (or six axes as shown in Figure 2), and each joint is controlled by a DC servo motor. The joint is defined by its angle, and also named axis. The main component of PUMA 560 robot electrical system is the controller [19,20].

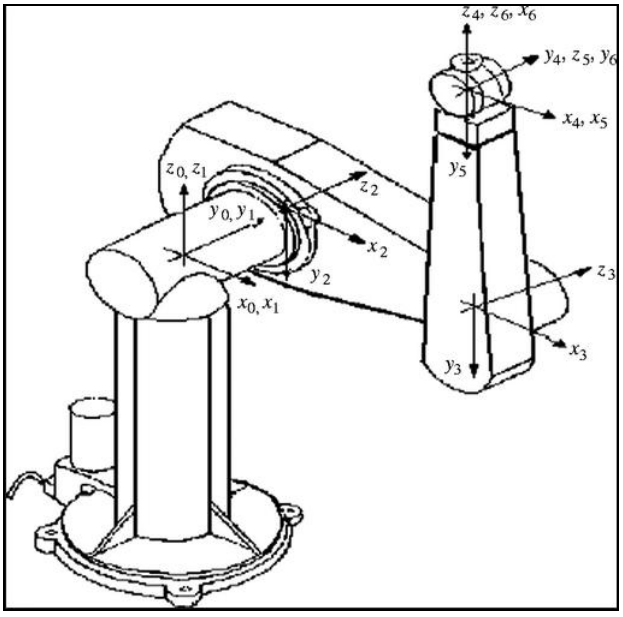

Figure 2. The PUMA 560 in the Zero Position with Attached coordinates Frames Shown [20]

3.2. Dynamic Formulation of the PUMA560 Arm Manipulator

The dynamic equations of the puma 560 arm manipulator given by [20], have been developed in terms of the position and time derivatives of the puma arm joint angles with the formula bellow:

$$\tau = A(q)\ddot{q} + B(q)[\dot{q}\dot{q}] + C(q)\left[\dot{q}^2\right] + g(q) \qquad (1)$$

Where:
q: is the joint position (joint angle).
A (q): is the n * n kinetic energy matrix, which is symmetric.
B (q): is the n * n (n-1)/2 matrix of Coriolis torques.
C (q): is the n *n matrix of centrifugal torques.
g (q):is the n-vector of gravity torques;
$\ddot{q}$: is the n-vector of accelerations.
τ : is the generalized joint force vector.

The symbols $[\dot{q}\dot{q}]$ and $\left[\dot{q}^2\right]$ are respectively the notation for the n*(n-1)/2 vector of velocity products and the n*1vector of squared velocities, given by:

$$\left[\dot{q}^2\right]:\left[\dot{q}_1^2, \dot{q}_2^2 \cdots \dot{q}_n^2\right]. \qquad (2)$$

$$[\dot{q}\dot{q}]:[\dot{q}_1\dot{q}_2, \dot{q}_1\dot{q}_3 \cdots \dot{q}_1\dot{q}_n, \dot{q}_2\dot{q}_3, \dot{q}_2\dot{q}_4, \cdots \dot{q}_2\dot{q}_{n-2}, \dot{q}_{n-1}\dot{q}_n]^T$$

The matrices A (q), B (q), C (q) and g (q) are given in following [19] with the Abbreviation used:

$$\begin{aligned} &S2 = \sin(q2); C5 = \cos(q5); \\ &C23 = \cos(q2 + q3); \\ &S223 = \sin(q2 + q2 + q3); \\ &CC2 = \cos(q2) * \cos(q2); \\ &CS4 = \cos(q4) * \sin(q4). \end{aligned} \qquad (3)$$

The expressions giving the elements of the kinetic energy matrix A, having the units of kg-m^2

$$\begin{aligned} &A_{11} \approx 2.57 + 1.38 * CC2 + 0.30 * SS23 + 7.44 * 0.1 * C2 * S23; \\ &A_{12} \approx 6.90 * 0.1 * S2 - 1.34 * 0.1 * C23 + 2.38 * 0.01 * C2; \\ &A_{13} \approx -1.34 * 0.1 * C23 - 3.97 * 0.001 * S23; \\ &A_{14} \approx A_{15} \approx A_{16} \approx 0; \\ &A_{22} \gg 6.79 + 7.44 * 0.1 * S3; \\ &A_{23} >> 0.333 + 3.72 * 0.1 * S3 - 1.10 * 0.01 * C3; \\ &A_{24} \approx A_{25} \approx A_{26} \approx 0; \\ &A_{33} \approx 1.16; \\ &A_{34} \approx 1.25 * 0.001 * S4 * S5; \\ &A_{35} \approx 1.25 * 0.001 * C4 * C5; A_{36} \approx 0; \\ &A_{44} \approx 0.20; A_{45} \approx A_{46} \approx 0; \\ &A_{55} \approx 0.18; A_{56} \approx 0; A_{66} \approx 0.19. \end{aligned} \qquad (4)$$

The expressions giving the elements of the Coriolis matrix B, having the units of kg-m^2

$$\begin{aligned} B_{112} \approx &-2.76 * SC2 + 7.44 * 0.1 * C223 + 0.60 * SC23 \\ &- 2.13 * 0.01 * (1 - 2 * SS23) \\ B_{113} \approx &\ 7.44 * 0.1 * C2 * + 0.60 * SC23 + 2.20 * 0.01 * C2 * S23 \\ &- 2.13 * 0.01 * (1 - 2 * SS23) \end{aligned}$$

$$B_{114} \approx -2.50*0.001*SC23*S4*S5+8.60*0.0001*C4*S5 - 2.48*0.001*C2*C23*S4*S5$$

$$B_{115} \approx -2.50*0.001*(SS23*S5-SC23*C4*C5) - 2.48*0.001*C2*(S23*S5-C23*C4*C5) + 8.60*0.0001*S4*C5;$$

$$B_{116} = 0;$$

$$B_{123} \approx 2.67*0.1*S23-7.58*0.001*C23$$

$$B_{124} = B_{125} = B_{126} \approx 0; B_{134} = B_{124};$$
$$B_{135} = B_{125}; B_{136} = B_{126}; B_{145} \approx 0;$$
$$B_{146} \approx 0; B_{156} \approx 0; B_{212} = 0; B_{213} = 0$$
$$B_{214} \approx 1.64*0.001*S23-2.50*0.001*C23*C4*S5 + 2.48*0.001*S2*C4*S5 + 0.30*0.001*S23*(1-(2*SS4))$$
$$B_{215} \approx -2.50*0.001*C23*S4*C5 + 2.48*0.001*S2*S4*C5 - 6.42*0.0001*C23*S4$$
$$B_{216} = -B_{126}; B_{223} \approx 2.20*0.01*S3+7.44*0.1*C3;$$
$$B_{224} \approx -2.48*0.001*C3*S4*S5$$
$$B_{225} \approx -2.50*0.001*S5 + 2.48*0.001*(C3*C4*C5-S3*S5)$$
$$B_{226} = 0; B_{234} = B_{224}; B_{235} = B_{225}; B_{236} = 0;$$
$$B_{245} \approx 0; B_{246} \approx 0; B_{256} \approx 0; B_{312} = 0; B_{313} = 0$$
$$B_{314} \approx -2.50*0.001*C23*C4*S5 + 1.64*0.001*S23 + 0.30*0.001*S23*(1-2*SS4)$$
$$B_{315} \approx -2.50*0.001*C23*S4*C5 - 6.42*0.0001*C23*S4$$
$$B_{316} \approx -B_{136}; B_{323} = 0; B_{324} \approx 0;$$
$$B_{325} \approx -2.50*0.001*S5; B_{326} = 0; B_{334} = B_{324};$$
$$B_{335} = B_{325}; B_{336} = 0; \quad (5)$$
$$B_{345} \approx -2.50*0.001*S5*C5; B_{346} = B_{246};$$
$$B_{356} = B_{256}; B_{412} = -B_{214}; B_{413} = -B_{314}; B_{414} = 0;$$
$$B_{415} = -6.42*0.0001*S23*C4$$
$$B_{416} = -B_{146}; B_{423} = -B_{324}; B_{424} = 0;$$
$$B_{425} \approx 6.42*0.0001*S4;$$
$$B_{426} = -B_{246}; B_{434} = 0; B_{435} = B_{425}; B_{436} = -B_{346};$$
$$B_{445} \approx 0; B_{446} = B_{456} = 0; B_{512} = -B_{215}; B_{513} = -B_{315};$$
$$B_{514} = -B_{415}; B_{515} = 0; B_{516} = -B_{156}; B_{523} = -B_{325};$$
$$B_{524} = -B_{425}; B_{525} = 0; B_{526} = B_{256}; B_{534} = B_{524};$$
$$B_{535} = 0; B_{536} = 0; B_{545} = 0; B_{546} = -B_{456};$$
$$B_{556} = 0; B_{612} = B_{126}, B_{613} = B_{136}, B_{614} = B_{146};$$
$$B_{615} = B_{156}; B_{616} = 0; B_{623} = 0; B_{624} = B_{246};$$
$$B_{625} = B_{256}; B_{626} = 0; B_{634} = B_{624}; B_{635} = B_{625};$$
$$B_{636} = 0; B_{645} = B_{456}; B_{646} = 0; B_{656} = 0$$

The expressions giving the elements of the Coriolis matrix C, having the units of kg-m^2

$$C_{11} = 0; C_{12} \approx 6.90*0.1*C2+1.34*0.1*S23 - 2.38*0.01*S2;$$
$$C_{13} = 0.5*B_{123}; C_{14} \approx 0; C_{15} \approx 0; C_{16} = 0;$$
$$C_{21} = 0.5*B_{112}; C_{22} = 0; C_{23} = 0.5*B_{223};$$
$$C_{24} \approx 0; C_{15} \approx 0; C_{26} = 0; C_{31} = -0.5*B_{113};$$
$$C_{32} = -C_{23}; C_{33} = 0$$
$$C_{34} = 1.25*0.001*C4*S5; C_{35} \approx C_{34}; C_{36} = 0^o \quad (6)$$
$$C_{41} = -0.5*B_{114}; C_{42} = 0.5*B_{224}; C_{43} = 0.5*B_{423};$$
$$C_{44} = 0^o; C_{45} = 0^o, C_{46} = 0; C_{51} = 0.5*B_{115};$$
$$C_{52} = -0.5*B_{225}; C_{53} = 0.5*B_{523}; C_{54} = -0.5*B_{445};$$
$$C_{55} = 0^o; C_{56} = 0^o; C_{61}=C_{62}=C_{63}=C_{64}=C_{65}=C_{66} = 0;$$

Gravity terms, having the units of Newton-meters

$$g_1 = 0; g_2 \approx -37.2*C2-8.4*S23+1.02*S2;$$
$$g_3 \approx -8.4*S23+0.25*C23; \quad (7)$$
$$g_4 \approx 2.8*0.01*S23*S4*S5;$$
$$g_5 \approx -2.8*0.01*(C23*S5+S23*C4*C5); g_6 = 0.$$

The direct dynamic model ($\ddot{q}$) given in [8] used to simulate the behavior of our arm manipulator and its control loop, is given :

$$\ddot{q} = -A^{-1}(q)*\left[B(q)[\dot{q}\dot{q}]+C(q)\left[\dot{q}^2\right]+g(q)-\tau\right] \quad (8)$$

So to simulate the behavior of our arm as indicate the Figure 3 in [8], we need several blocks like the trajectory generator block and the control block.

Figure 3. The simulation loop

3.3. Trajectory Generation

The motion can be described both in joint space and Cartesian space as described in [21].in our work, we consider methods of path generation in which the path shapes are described in terms of functions of joint angles.

Many ways to generate a trajectory in the joint space in [21] and [22], among them the fifth order polynomial given by the following equation:

$$q(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5 \quad 0 \le t \le t_f \quad (9)$$

And since in [8]: $q(t) = q^i + r(t)*D$

With:
- $D = q^f - q^i$
- qi : is the initial joint position
- q^f : is the final joint position

Then:

$$r(t) = 10(t/t_f)^3 - 15(t/t_f)^4 + 6(t/t_f)^5 \quad (10)$$

3.4. The Computed Torque Controller

Although control systems based on approximate linear models are popular in current industrial robots, it is

important to consider the complete nonlinear dynamics of the manipulator when synthesizing control algorithms [19]. Computed torque controller is based on feedback linearization and computes the required arm torques using the nonlinear feedback control law [23], this nonlinear technique of controlling a manipulator promise better performance than do simpler linear schemes**.**

We wish to develop this control in the configuration space, under the assumption of that the joint position and velocities are measurable and that the measurements are not affected with noise.

We have chosen the computed control law given in [21] in case that the motion is completely specified (i.e. we wish to cause the manipulator joints to follow prescribed position trajectories), one possibility is to use the dynamic equation of the arm as follow:

$$\tau = A(q)\tau_0 + B(q)[\dot{q}\dot{q}] + C(q)\left[\dot{q}^2\right] + g(q) \qquad (11)$$

Where

$$\tau_0 = \ddot{q}_d + k_v * (\dot{q}_d - \dot{q}) + k_p * (q_d - q) \qquad (12)$$

$\ddot{q}_d$, $\dot{q}_d$, and q_d : Are respectively the desired joint accelerations, velocities and positions in the joint space, calculates by the trajectory generator.

τ : Is the generalized joint force vector

τ_0 : Is the auxiliary control

$\dot{q}, q$: Are respectively the joint velocities and positions determined by using feedback from the joint sensors to compute the torques required.

k_p, k_v : Are controller gains

3.5. The Simulink Diagrams

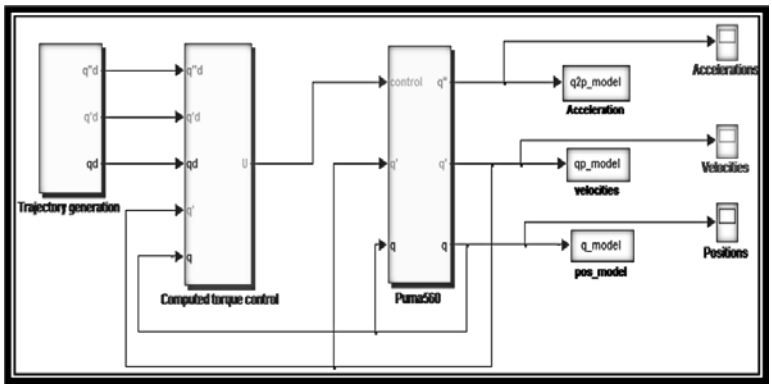

Figure 4. Simulink diagram of the simulation loop of the Puma 560 arm manipulator

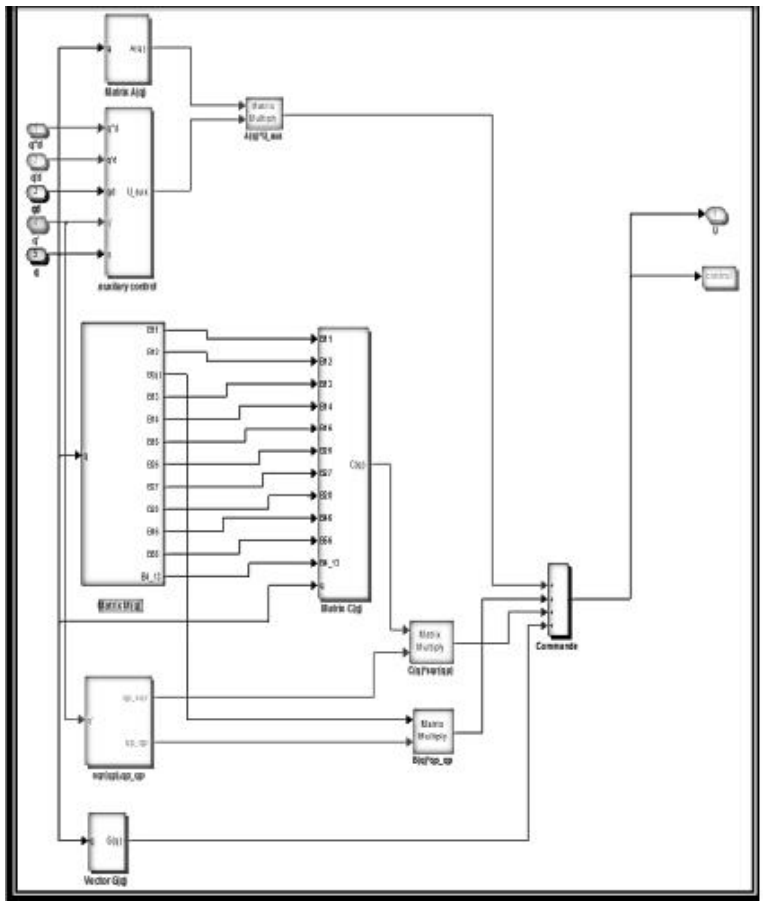

Figure 5. Simulink diagram of the computed torque control

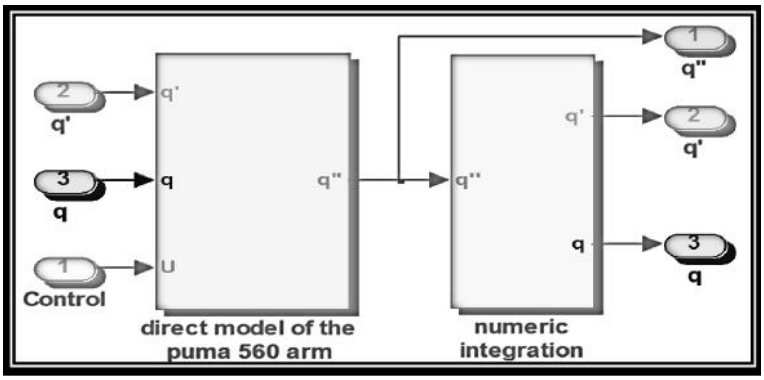

Figure 6. Block diagram for $\ddot{q}, \dot{q}$ and q

4. NSGAII Controller Analysis, Modeling and Implementation on PUMA 560 Arm Manipulator

4.1. PD Controllers

The block diagram shown in the Figure 7 illustrates a closed-loop system with a PD controller in the direct path, which is the usual connection. The system's output should follow as closely as possible the reference signal (set point). The PD controller is characterized by two gains $k_p k_d$, as shown in Figure 8 where $k_p k_d$ controller gains.

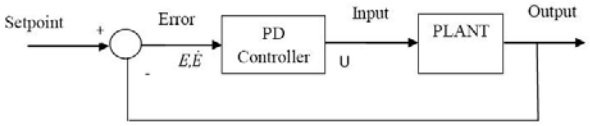

Figure 7. PD control of a plant

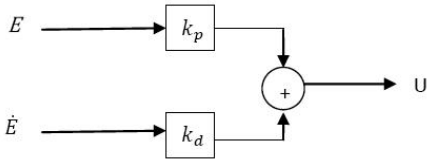

Figure 8. PD controller internal structure

4.2. NSGA-II Algorithm

Non-dominated Sorting Genetic Algorithm II (NSGA-II) is the heavily revised version of the Non-dominated Sorting Genetic Algorithm (NSGA), which was introduced in the mid 1990 [17]. The NSGA algorithm is a popular non-domination based genetic algorithm for multi-objective optimization. It is a very efficient algorithm but has been generally criticized for its computational complexity, needs elitism operator and a value of the sharing parameter 'σ share' should be chosen a priori .The NSGA-II is a modified version that adopts a more efficient ranking procedure than its predecessor [14]. Also, it estimates the density of solutions surrounding a particular solution in the population by computing the average distance of two points on either side of this point along each of the objectives of the problem. This value is the so-called crowding distance. During selection, the NSGA-II uses a crowded -comparison operator which takes into consideration both the non domination rank of an individual in the population and its crowding distance (i.e., non-dominated solutions are preferred over dominated solutions, but between two solutions with the same non domination rank, the one that resides in the less crowded region is preferred) [17].

4.2.1. General Description of NSGA-II

1. The population is initialized as usual.
2. The m fitness values of the individuals in the population are calculated.
3. The rank of each individual (solution) in the population is calculated based in Pareto dominance relation originally proposed by Francis Ysidro Edgeworth in 1881,but generalized by the French Italian economist Vilfredo Pareto in 1896 [15]. The set of Pareto optimal solutions is defined in the following definitions:

Definition: The Pareto optimal set $P*$ is defined as:

$$P* = \left\{ x \in X \,\middle|\, \nexists\, y \in X : f(y) \le f(x) \right\}.$$

4. The first front being completely non-dominant set in the current population and the second front being dominated by the individuals in the first front only and the front goes so on. Each individual in each front are assigned rank (fitness) based on front in which they belong to. Individuals in first front are given a value of 1 and individuals in second are assigned fitness value as 2 and so on.
5. Calculate the crowding distance, a new parameter, calculating as follow for each individual [14]:

For each front F_i , n is the number of individuals.

- Initialize the distance to be zero for all the individuals i.e. $F_i(d_j) = 0$, where j corresponds to the j^{th} individual in front Fi.
- For each objective function m

  *Sort the individuals in front F_i based on objective m i.e. $I = sort(F_i, m)$.

  * Assign infinite distance to boundary values for each individual in F_i i.e.

  $$I(d_1) = \infty \quad and \quad I(d_n) = \infty.$$

  * For k=2 to (n-1)

  $$I(d_k) = I(d_k) + \frac{I(k+1)\cdot m - I(k-1)\cdot m}{f_m^{\max} - f_m^{\min}} \qquad (14)$$

Where $I(k),m$ is the value of the m^{th} objective function of the k^{th} individual in I.

6. The parents for the crossover are selected: Parents are selected from the population by using binary tournament selection based on the rank and crowding distance. An individual is selected if the rank is lesser than the others. In case of solutions that have the same rank, the individual with the greatest crowding distance is chosen.
7. Generation of the off springs: The selected individuals generate off springs from crossover and mutation operators.
8. The old population and current offspring are sorted again based in non-domination, and only the best N individuals are selected, where N is the population size. The selection is based in rank and the crowding distance on the last front.
9. Back to 6 a convergence criterion is met. More details on the algorithm are found in [14].

4.3. NSGAII Tuning of PD Controller of Puma560

The NSGAII algorithm is based on the genetic algorithm, so to represent a solution we must define it as a chromosome (i.e.: a string). Puma560 contains independent controller for each joint. For six joint controllers, there are 12 values for PD parameters.

By taking a 12 variable string as $\left[k_{p1}k_{p2}\ldots k_{p6}k_{v1}k_{v2}\ldots k_{v6}\right]$ for NSGAII, an optimal value can be searched. The method of tuning PD parameters using NSGAII is based in minimizing the Integral of Absolute value of Error IAE and considerate as vector of fitness of joints [24]. If $q_d(k)$ is desired trajectory vector and $q_d(k)$ is output trajectory vector then the error vector $e(k)$ is

$$e(k)=q_d(t)-q(t) \tag{15}$$

$$IAE=\sum_{k=1}^{k=i}\left|e(k)\right| \tag{16}$$

Where $e(k)$ is the system error at l^{th} sampling instant.

The implementation of the NSGAII is presented as follow:

1. Population is initialized
   This function initializes the population with N individuals. Each individual is presented as $\left[k_{p1}k_{p2}\ldots k_{p6}k_{v1}k_{v2}\ldots k_{v6}\right]$, and each $k_{pi}k_{vj}$ taken its value randomly from $1\ldots100\left(i=1\ldots6,\ j=1\ldots6\right)$.
2. Each fitness
   $f_j\ j=1\ldots6\left(f_j=IAE_j\right)$ is calculated for each individual in the population, and save its value, by concatenation of each individual string and its fitness string.
3. Sort the current population based on the non-domination sort by calculating the rank value and the crowding distance value of each individual.
4. Individuals (parents) are selected by tournament selection.
5. The Simulated Binary Crossover (SBC) is applied to generate off springs find in [14]. In addition to the crossover and mutation probabilities, the SBC needs to be implemented two new parameters: the distribution index for crossover η_c and the mutation distribution index η_m.
6. Step 2 and 3 are applied to the whole population
7. Next population is formed by selection of the best n individuals having minimum rank, and the greater crowding distance for the individual of last rank.
8. back to 4 until a predefined number of generations.

NSGAII acts as a controller which modifies the set of parameters of the j^{th} population which consist of P individual parameters of control system. This modification is based in minimizing simultaneously six position errors**.**

In the evaluation step of NSGAII, a simulation is performed for each controller. Parameters used for simulation of NSGAII given in [25] are: Crossover probability < 0.9, Mutation probability > 0.9 [14], $\eta_c=20$, $\eta_m=20$, tour-size=2 (size of the tournament selection), pool-size=ceil (size (population, 1)/2): pool-size is the number of individuals selected to form a mating pool after performing tournament selection.

The population size and the generation number are defined by the user.

5. Results and Discussion

5.1. Result of Simulink

The simulation was implemented in MATLAB/ SIMULINK environment. It is noted that, to simulate the puma560 model in the Simulink environment, we used the PD values given in Table.1, found empirically in [26].

Table 1. PD gains

Joint	1	2	3	4	5	6
k_p	700.0	1100	400.0	40.0	30.0	40.0
k_p	20.0	20.0	20.0	5.0	5.0	5.0

To generate trajectory in joint space directly, we used as initial and final positions noted respectively q^i,q^f found in [17]:

$$q^i=\left(-20^\circ,60^\circ,-120^\circ,0^\circ,-30^\circ,0^\circ\right) \tag{17}$$

$$q^f=\left(20^\circ,-60^\circ,-60^\circ,0^\circ,30^\circ,0^\circ\right) \tag{18}$$

The system is simulated in 1 second and the sampling time is 0.01s.

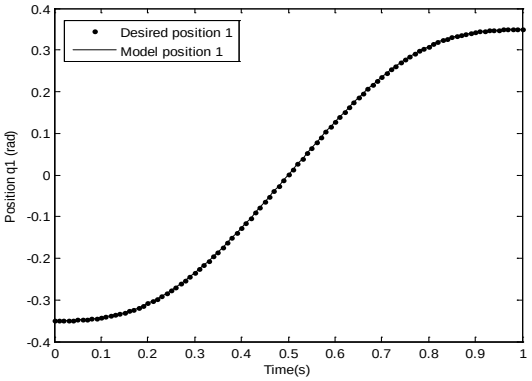

Figure 9. Desired and model joint angles q1

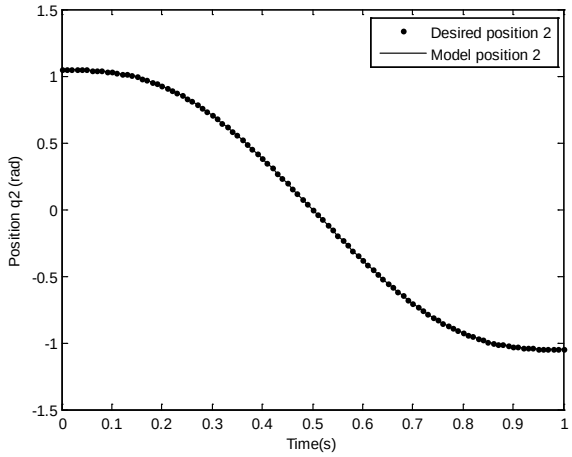

Figure 10. Desired and model joint angles q2

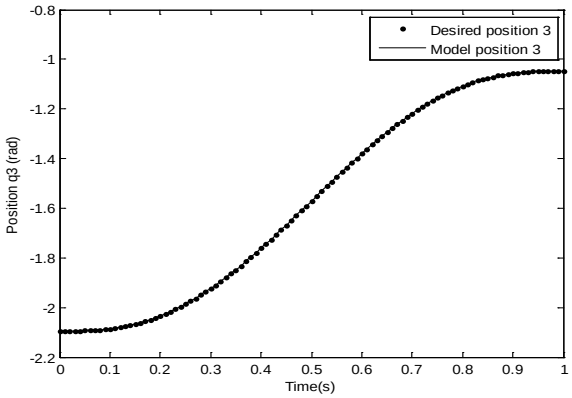

Figure 11. Desired and model joint angles q3

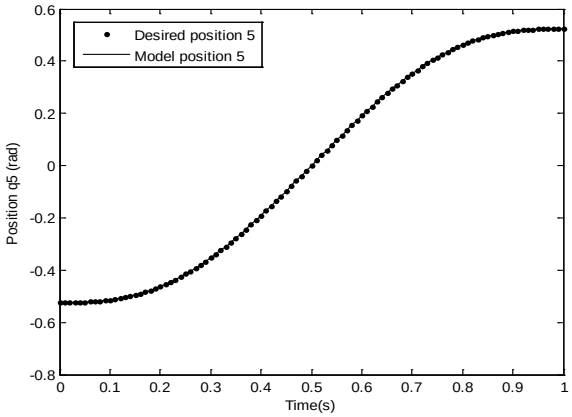

Figure 12. Desired and model joint angles q5

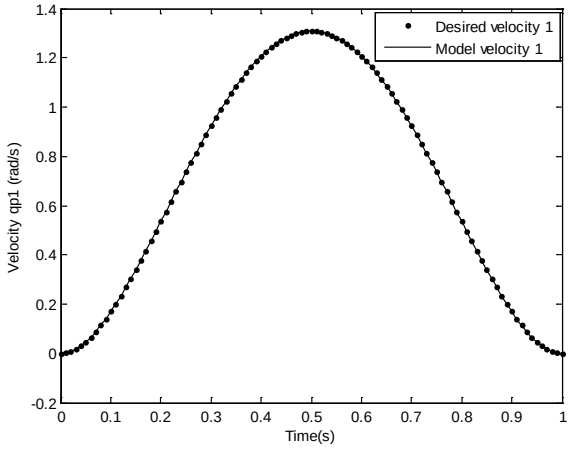

Figure 13. Desired and model joint velocity qp1

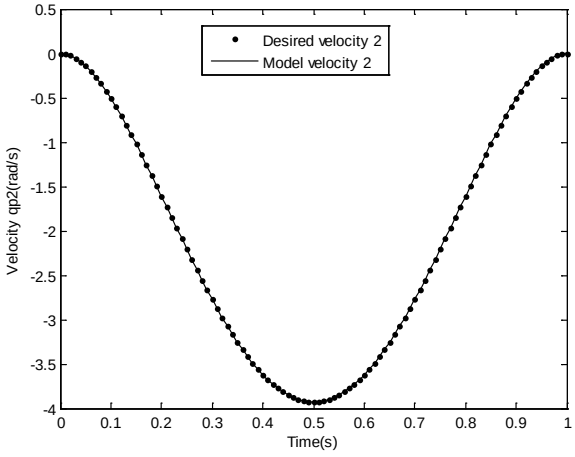

Figure 14. Desired and model joint velocity qp2

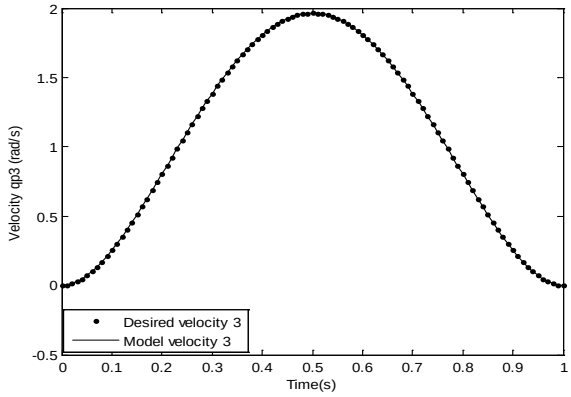

Figure 15. Desired and model joint velocity qp3

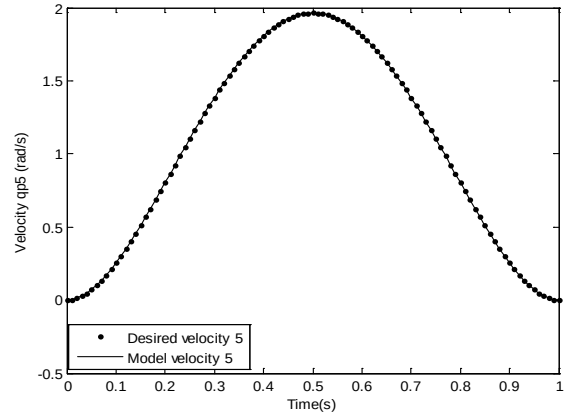

Figure 16. Desired and model joint velocity qp5

5.2. Result of NSGAII Algorithm Using for PD Tuning

In the experiments, the MATLAB R2012a environment has been used for implementing and running the NSGAII algorithm. Experiments with PUMA560 system have been performed for the evaluation of the tuning procedure.

The results obtained are the joint angles (joint positions $q_j\, j = 1,2,3,5$) and the joint velocities ($q\dot{p}_J$, which are respectively, compared with the desired joint angles values and the desired joint velocities values. We don't cite the 4th and the 6th joint because it goes from 0° to 0°.

As a test, the NSGAII algorithm has been configured as follows: Population size: 2, Generations: 2. Results are shown in Figure 17 to Figure 24 with an elapsed time 1.4534 s.

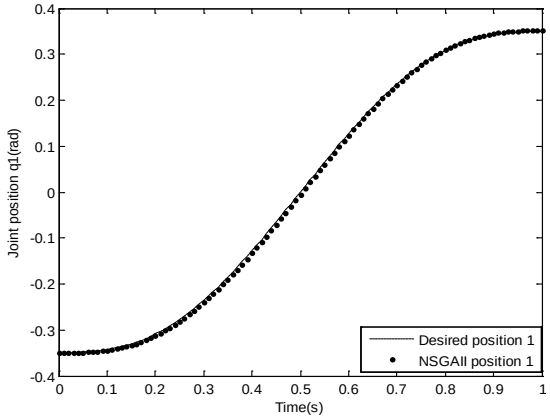

Figure 17. Desired and NSGAII joint angles q1

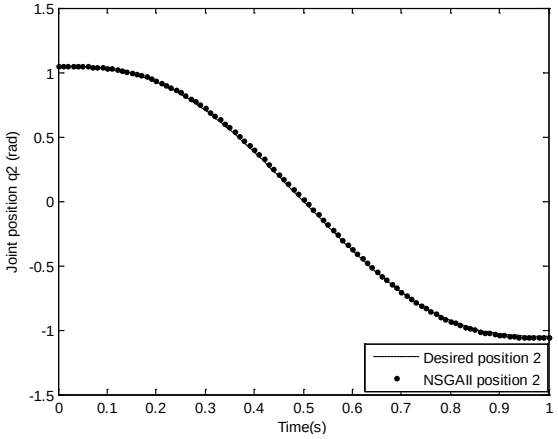

Figure 18. Desired and NSGAII joint angles q2

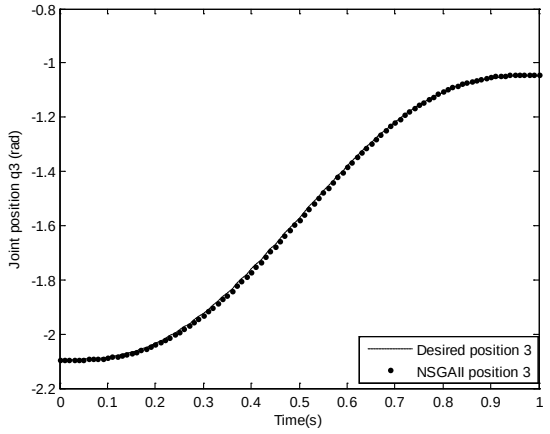

Figure 19. Desired and NSGAII joint angles q3

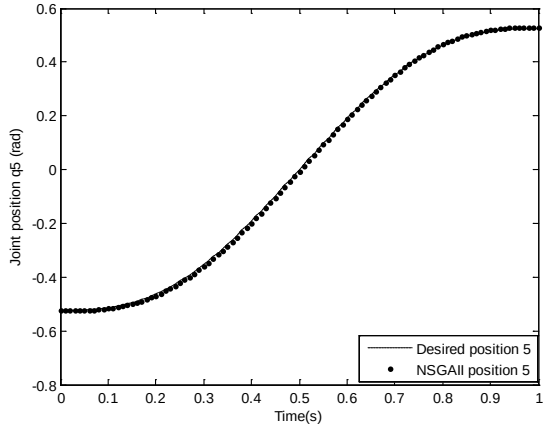

Figure 20. Desired and NSGAII joint angles q5

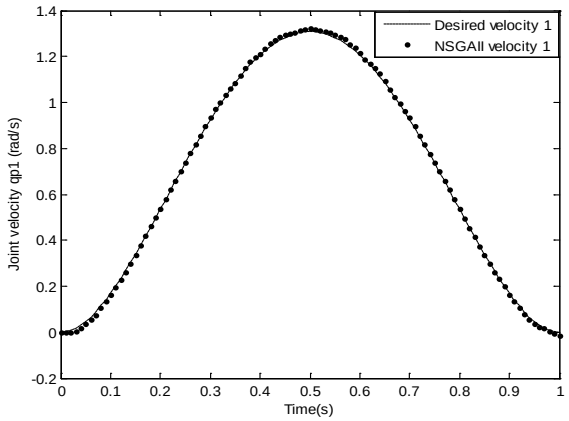

Figure 21. Desired and NSGAII joint velocities qp1

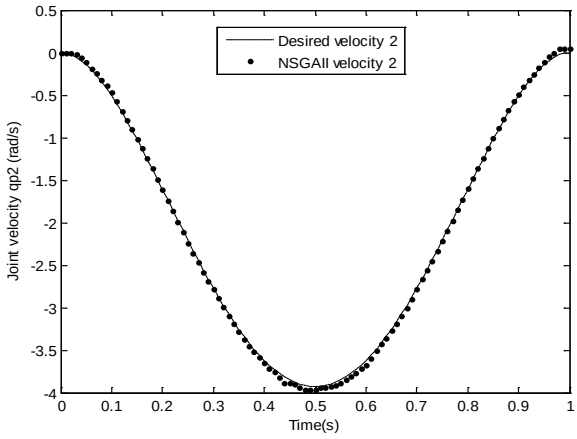

Figure 22. Desired and NSGAII joint velocities qp2

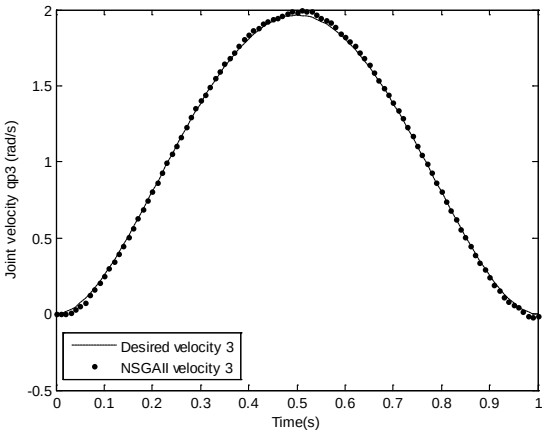

Figure 23. Desired and NSGAII joint velocities qp3

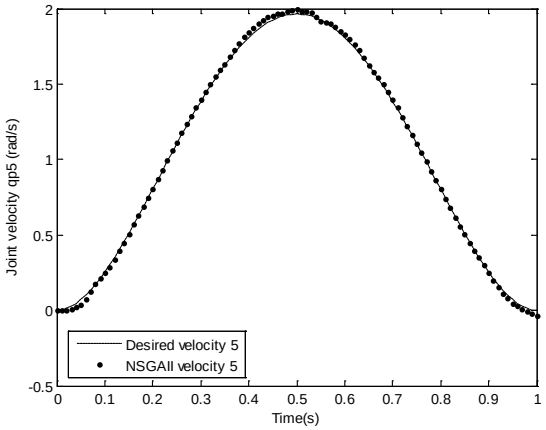

Figure 24. Desired and NSGAII joint velocities qp5

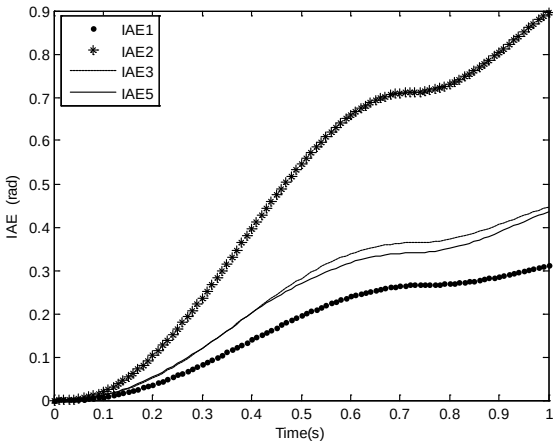

Figure 25. the IAE (fitness values) of the joint positions

It can be seen in Figure 17 to Figure 24 that the NSGAII joint angles curves converge to the desired joint angles curves. Also, it can be observed that the NSGAII joint velocities curves converge to the desired joint velocities curves with very small deviation in some areas of the velocities curves.

The Figure 25 shows that in 0.71s and 0.72 second all fitness values i.e. the IAE (0.71) are nearly equal to IAE (0.72) as shown in table.2, that is to say that the errors in that seconds are near zero in 0.72s, so we can say that the PD gains generated in this time are good for the six joint controllers.

Table 2. IAE error of the 1st,2nd ,3rd and the 5th joint of thePUMA560

Time (s)	IAE1	IAE2	IAE3	IAE5
0.71	0,2652	0,7113	0,3633	0.3395
0.72	0,2662	0,7119	0.3639	0.3395

From the table.2, we can say that the k_p, k_d gains generated at the 0.72s are the best, as shown in table.3:

Table 3. The best PD gains obtained by NSGAII

k_{p1}	k_{p2}	k_{p3}	k_{p4}	k_{p5}	k_{p6}	k_{d1}	k_{d2}	k_{d3}	k_{d4}	k_{d5}	k_{d6}
22	27	54	45	89	6	52	39	65	43	40	51

Results of model simulations where the gains are tuned empirically Figure 9 to Figure 16 are very good, but because some joints P gain are high (700, 1100) compared to those obtained numerically by NSGAII that are between 0 and 100, we can say that tuning through NSGAII led to satisfactory responses for the system tested because in [15], in industry we cannot use regulators with so huge gains.

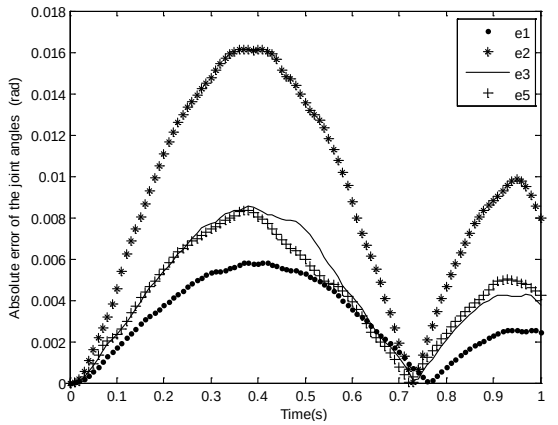

Figure 26. The absolute error of the joint positions

The Figure 26 shows the absolute error of the joint angles obtained by minimizing the IAE errors.

So the NSGAII tuned PD controller is showing good performance in 1.4534 s

6. Conclusion

This work is focused on the NSGAII (Non-dominated Sorting Genetic Algorithm II) algorithm and investigation of its applicability to the automatic tuning of PD controllers of the PUMA560 system. And since the system has six PD controllers, the method searches for a combination of gains so that the errors between actual and desired responses are minimized. The method was carried out in term of IAE in joint space. It can be concluded from the results that NSGAII is showing good results. As sometimes is the case with this method, there was no need for further manual adjustments to the PD gains when automatic tuning was employed.

References

[1] Whidborne, J.F., G. Murad, D.-W. Gu and I. Postlethwaite *Robust control of an unknown plant*, the IFAC 93 benchmark. Int. J. Control, vol. 61, no. 3, pp. 589-640, 1995a.

[2] Whidborne, J. F., I. Postlethwaite and D.-W. Gu *A mixed optimization approach to multi-objective computer-aided control system design*, In Proc. IEEE Symp. on Comp. Aided Contr. Syst. Design (CACSD'96). Dearborn, Michigan: pp. 309-314, 1996.

[3] Liu, G. P. and R. J. Patton, *Parametric state feedback controller design of multivariable systems*, International Journal of Control, vol. 61, no. 6, pp. 1457-1464, 1995.

[4] Liu, G. P. and R. J. Patton, *Robust control design using eigenstructure assignment and multiobjective optimization*, Int. Journal of Systems Science, vol. 27, no. 9, pp. 871-879, 1996.

[5] Whidborne, J.F., I. Postlethwaite and D.-W. Gu, *Multiobjective control system designs a mixed optimization approach*, Recent Trends in Optimization Theory and Applications, R. P. Agarwal (ed.), vol. 5 of World Scientificc Series in Applicable Analysis Singapore: World Scientific pp. 467-482, 1995b.

[6] Liu, G. P. and S. Daley, *Stable dynamical controller designs for robust polynomial pole assignment*,. IEE Proceedings, Part D, vol. 145, no. 3, pp. 259-264, 1998.

[7] Liu, G. P. and S. Daley, *Optimal-tuning PID control for industrial systems*, Control Engineering Practice, vol. 9, 2001 (accepted).

[8] Liu, G. P. and V. Kadirkamanathan, *Multiobjective criteria for nonlinear model selection and identification with neural networks*, IEE Proceedings, Part D, vol. 146, no. 5, pp. 373-382, 1999.

[9] Chen, J., R. J. Patton and G. P. Liu, *Optimal residual design for fault diagnosis using multi-objective optimisation and genetic algorithms*, International Journal of Systems Science, vol. 27, no. 6, pp. 567-576, 1996.

[10] Tabak, T., A. A. Schy, D. P. Giesy, and K. G. Johnson, *Application of multiple objective optimization in aircraft control systems design*,. Automatica, vol. 15, pp. 595-600. 1979.

[11] Skelton, R. E. and M. DeLorenzo, *Spacestructure control design by variance assignment*, J, Guid., vol. 8, no. 4, pp. 454-462. 1985.

[12] Koussoulas, N. T. and C. T. Leondes, *The multiple linear quadratic Gaussian problems*. International Journal of Control, vol. 43, no. 2, pp. 337-349. 1986.

[13] Toivonen, H. T. and P. M. Makila, *Computer-aided design procedure for multi-objective LQG control problems*, International Journal of Control, vol. 49, no. 2, pp. 655-666. 1989.

[14] Khargonekar, P. P. and M. A. Rotea, *Multiple objective optimal control of linear systems: the quadratic norm case*,IEEE Trans. Automat. Contr., vol. 36, no.1, pp. 14-24, 1991.

[15] Jaimes A.L., Zapotecas Martinez S., C.A. C. Coello, *Chapter 1 An Introduction To Multi-Objective Optimization Techniques*, In: Book Title Editor: Editor Name, pp. 1-26, 2009 Nova Science Publishers, Inc.

[16] Gambier A, *MPC and PID Control Based on Multi-objective Optimization*, American Control Conference Westin Seattle Hotel, Seattle, Washington, USA, June 11-13, 2008.

[17] Zitzler E and Thiele L, *Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach*. IEEE Transactions on Evolutionary Computation, 1999, 3(4):257-271.

[18] J. Rutherford, *Using the PUMA560 Robot*, PUMA Unofficial User's Guide, 2012.

[19] J. J. Craig, *Introduction to Robotics Mechanics and Control*, Third Edition" 2005 Pearson Education, Inc.

[20] B Armstrong, O Khatib, Joel Burdick, *The Explicit Dynamic Model and Inertial Parameters of the PUMA 560 Arm*, IEEE, 1986.

[21] W. khalil, E. Dombre, *Modélisation et commande des robots*, Hermes Edition, Paris, 1988.

[22] J. J. Craig, *Introduction to Robotics Mechanics and Control, second Edition*, 1989, 1986 by Addison-Wesley Publishing Company, Inc.

[23] F. Piltan , M.H. Yarmahmoudi, M. Shamsodini, E. Mazlomian & A. Hosainpour, *PUMA-560 Robot Manipulator Position Computed Torque Control Methods Using MATLAB/SIMULINK and Their Integration into Graduate Nonlinear Control and MATLAB Courses*, International Journal of Robotics and Automation, (IJRA), Volume (3): Issue (3) : 2012.

[24] D. Czarkowski, *Identification And Optimization PID Parameters Using Matlab*, report of the project for bachelor of engineering in electronics, Cork Institute of Technology Gdynia Maritime University, Cork 2002.

[25] Sumathi S, Surekha P, *Computational intelligence paradigms: theory & applications using MATLAB*, © 2010 by Taylor and Francis Group, 2010.

[26] N. Moreira , P. Alvito and P. Lima, *First steps towards an open control architecture for a PUMA 560*, Proc. 2nd Portuguese Conf. On Automatic Control, 1996.