Pareto design of fuzzy tracking control based on particle swarm optimization algorithm for a walking robot in the lateral plane on slope

M.J. Mahmoodabadi1*, M. Taher-Khorsandi2

1 Department of Mechanical Engineering, Sirjan University of Technology, Sirjan, Iran
2 Department of Mechanical Engineering, University of Texas at San Antonio, San Antonio, USA

ABSTRACT: Many researchers have controlled and analyzed biped robots that walk in the sagittal plane. These robots require the capability of walking merely laterally when they are faced with the obstacles such as a wall. In this field of study, both nonlinearity of the dynamic equations and also having a tracking system cause an effective control has to be utilized to address these problems. Therefore, this paper presents a nonlinear fuzzy tracking control for the walking robots that step in the lateral plane on a slope. When fuzzy control is utilized to track the desired trajectories of the joints, there has to be a trade-off between tracking errors and control efforts. Consequently, a particle swarm optimization algorithm is used to obtain the Pareto front of these non-commensurable objective functions to determine the fuzzy control parameters. In this paper, normalized summation of angle errors and normalized summation of control efforts are considered as the objective functions. These objective functions have to be minimized simultaneously. A vector which contains the control parameters is considered as the vector of selective parameters with positive constant values. The obtained Pareto front by the proposed multi-objective algorithm is compared with three prominent algorithms, modified NSGAII, Sigma method and MATLAB Toolbox MOGA. The result dramatizes the superiority of innovative particle swarm optimization over the algorithms.

1- Introduction
The challenging walking robot field has attracted the interest of many researchers for several decades. The dynamic of this sort of robot is extremely nonlinear and difficult to control since researchers confront with a heavy nonlinearity in the dynamic equations which must track the desired trajectory. Lately, fuzzy control has been utilized by researchers as an effective control to satisfy criteria for nonlinearity of dynamic equation and tracking system. Liu et al. [1] illustrated that the fuzzy control could be utilized to control the walking robots. Li et al. [2] considered a robot walking in the sagittal plane in its desired ZMP and utilized a fuzzy motion control based on reinforcement learning and Lagrange polynomial interpolation for gait synthesis of walking robots. The most appropriate stability criterion is Zero Moment Point (ZMP). This criterion does take dynamic forces, as well as static forces, into consideration. Goswami et al. [3] computed ZMP by an approximation-based method and they optimized walking parameters such as step-length, bending-height, etc. Some studies introduced the concept of combined trajectory paths. Nasrardin mousavi et al. [4] illustrated that hip height plays an important role in both the stability and optimum actuator torques of the joints. All the above-mentioned works have studied walking robots that walk in the sagittal plane, but they did not study walking in the lateral plane. Ito et al. [5] proposed a static balance control based on the feedback of ZMP positions and extended these methods to the walking in-place lateral stepping motion; however, they did not study the walking of the robot in that plane. Taher Khorsandi et al. [6] introduced the motion of the walking robot in the lateral plane and linearized the dynamical equations of the robot in the trajectory and utilized linear quadratic tracking control to control them. Their work would be developed by the present study. They linearized the dynamical equation to control the robots; however, in this study a nonlinear fuzzy tracking control is used to control them and linearization would be relaxed. They considered a flat surface while in this paper, the robot walks on slope. Furthermore, an innovative optimization method is utilized to optimize the control parameters. The parameters of fuzzy tracking control have to be determined and they are usually identified by trial-and-error process. One proper way to choose these factors is using an evolutionary algorithm such as particle swarm optimization, genetic algorithm and etc. Shook et al. [7] used genetic algorithm to optimize fuzzy logic controllers which were designed to manage two 20 kN magnetorheological dampers for mitigation of seismic loads. Shayeghi et al. [8] proposed a multi-stage fuzzy controller for solution of the load frequency control which operated under deregulation and also the membership functions were designed automatically by particle swarm optimization (PSO). Bingul [9] controlled a 2-DOF planer robot by fuzzy logic controller, and particle swarm optimization was utilized to tune fuzzy parameters. A PID controller was also tuned by particle swarm optimization to contrast with the fuzzy controller. Therefore, an innovative particle swarm optimization presented in this paper is utilized to eliminate the boring and repetitive trial-and-error process and to find the fuzzy parameters. PSO, as one of the most recent modern heuristic algorithms, was primarily presented by Kennedy and Eberhart [10].

Keywords:
Walking Robot
Fuzzy Tracking Control
Particle Swarm Optimization
Multi-objective Optimization

Available Online: 19 October 2018

Review History:
Received: 15 April 2018
Revised: 4 October 2018
Accepted: 14 October 2018

Corresponding author, E-mail: Mahmoodabadi@sirjantech.ac.ir
It was developed through simulation of simplified social systems, and has been found to be robust in solving nonlinear optimization problems [11]. The PSO technique can generate a high quality solution with short calculation time and a more stable convergence characteristic compared to other evolutionary methods [12,13]. In this paper, for increasing the convergence of the population and to avoid local minima, PSO is merged with convergence and divergence operators. In the recent years, several approaches have been proposed to extend the PSO algorithm to deal with multi-objective optimization problems. For instance, dynamic neighborhood PSO [14], dominated tree [15], Sigma method [16], and vector evaluated PSO [17] have been proposed to solve the multi-objective optimization problems.

This research paper develops considerably the authors’ previous study [6] in the aspect of proposing the fuzzy tracking control optimized by multi-objective particle swarm optimization as an effectual controller for a walking robot that walks merely on the lateral plane of a slope. As a noteworthy development, optimal fuzzy tracking control based upon particle swarm optimization is proposed here to control effectively the walking of the biped robot in the lateral plane of a slope.

2- The model of the walking robot
A three link planar model in the lateral plane is used to model the robot [6]. The first link is anchored to the ground surface while the third link moves freely along lateral plane and the second link represents the head, arms and trunk. Each link is defined by four characteristics, that are mass, length, inertia and the center of gravity. The variable joints for this robot are regarded as \( \theta_1 \), \( \theta_2 \), and \( \theta_3 \). The numerical values of the related parameters are exactly considered same as those in [18]. The dynamical equations of the robot’s motion can be derived by using the Newton-Euler or the Lagrange-Euler formalisms as those in [6].

3- Fuzzy tracking control of walking robot
The proposed fuzzy tracking control is based on the closed-loop fuzzy system. The stages of the control method was designed and constructed step by step as follows. To control the system, the state variable vector is chosen as \( \{ x_1, x_2, x_3, x_4, x_5, x_6 \} = [ \theta_1, \theta_2, \theta_3, \theta_2, \theta_3, \theta_1 ] \). The errors could be defined as follows.

\[
E_p = \theta_{\text{ref}} - \theta_p \\
E_q = \theta_{\text{ref}} - \theta_q
\]  

(1)

The new error indexes parameters would be introduced as:

\[
\text{Index}_{ep} = \frac{E_p}{\left| \theta_{\text{ref}} \right| \left| \theta_p \right|}, \quad (p = 1, 2, 3)
\]

\[
\text{Index}_{eq} = \frac{E_q}{\left| \theta_{\text{ref}} \right| \left| \theta_q \right|}, \quad (q = 1, 2, 3)
\]  

(2)

Then, a membership function is constructed by Fig. 1 illustrated by Table 1. In the Fig. 1, the inference result \( f_i \) of the consequent variable \( f_1 \) should be calculated by the product-sum gravity method [19] via the following formulation.

\[
f_i(\text{Index}_x) = \sum_{i=1}^{5} \mu_i(\text{Index}_x) \]

(3)

where \( y' \) is the center of an output membership function and \( \mu_i \) is an input membership function. Finally, the control efforts are obtained by the following equation.

\[
u_i = w_i f_i + w_2 f_2
\]

\[
u_i = w_i f_i + w_3 f_3 + w_4 f_4
\]

(4)

In this equation, \( w_i, w_2, w_3, w_4 \), and \( w_5 \) are weight constants and these parameters are usually identified by trial-and-error process. One proper way to choose these factors is using evolutionary algorithms. Therefore, an innovative particle swarm optimization presented in this paper is utilized to eliminate the boring and repetitive trial-and-error process and to find fuzzy parameters \( w_i (i = 1, 2, 3, 4, 5, 6) \).

4- Single objective optimization algorithm
4-1- Particle swarm optimization
Particle swarm optimization is a population-based evolutionary algorithm and is similar to other population based evolutionary algorithms. PSO is motivated by the simulation of social behavior instead of survival of the fittest [10]. Although originally adopted for balancing weights in neural networks [20], PSO soon became a very popular global optimizer, mainly in problems in which the decision variables are real numbers [21].

In PSO, each candidate solution is associated with a velocity. The candidate solutions are called particles and the position of each particle is changed according to its own experience and that of its neighbors (velocity). It is expected that the particles will move towards better solution areas. Mathematically, the particles are manipulated according to the following equations,

\[
\dot{x}_i(t+1) = \dot{x}_i(t) + \nu_i(t+1),
\]

\[
\dot{v}_i(t+1) = \nu_i(t+1) + C_I \sum_{j \neq i} (x_{\text{best}} - x_j(t)) + C_R \sum_{j \neq i} (x_{\text{random}} - x_j(t)),
\]

(5)

(6)

where \( x_i(t) \) and \( v_i(t) \) denote the position and velocity of each particle at the time \( t \).
particle $i$, at time step $t$. $r, r_i \in [0,1]$ are random values. $C_i$ is the cognitive learning factor and represents the attraction that a particle has toward its own success. $C_p$ is the social learning factor and represents the attraction that a particle has toward the success of the entire swarm. $W$ is the inertia weight which is employed to control the impact of the previous history of velocities on the current velocity of a given particle. The personal best position of the particle $i$ is $\text{x}_\text{ipbest}$. and $\text{x}_\text{gbest}$ is the position of the best particle of the entire swarm. Inertia weight is used to balance the global and local search ability.

The characteristics of the inertia weight are similar to those of the temperature parameter in the simulated annealing algorithm [12]. A large inertia weight facilitates a global search while a small inertia weight facilitates a local search. By changing the inertia weight dynamically, the search ability is dynamically adjusted. Experimental results indicated that the linearly decreasing inertia weight over the iterations improve the performance of PSO [21]. With a large value of $C_1$ and a small value of $C_3$, particles are allowed to move around their personal best position ($\text{x}_\text{ipbest}$). With a small value of $C_1$ and a large value of $C_3$, particles converge to the best particle of the entire swarm ($\text{x}_\text{gbest}$). From the results, it was observed that best solutions were determined when $C_1$ is linearly decreased and $C_3$ is linearly increased over the iterations [22]. Hence, in this paper, the following linear formulation for inertia weight and learning factors are used.

$$W = W_i - (W_i - W_f) \times \left(\frac{t}{\text{max iteration}}\right),$$  \hspace{1cm} (7)

$$C_i = C_{i0} - (C_{i0} - C_{i_f}) \times \left(\frac{t}{\text{max iteration}}\right),$$  \hspace{1cm} (8)

$$C_p = C_{p0} - (C_{p0} - C_{p_f}) \times \left(\frac{t}{\text{max iteration}}\right).$$  \hspace{1cm} (9)

where $W_i$ and $W_f$ are the initial and final values of the inertia weight, respectively. $C_{i0}$ and $C_{i_f}$ are the initial values of the learning factors $C_1$ and $C_3$, respectively, $C_{p0}$ and $C_{p_f}$ are the final values of the learning factors $C_1$ and $C_3$, respectively. $t$ is the current iteration number and max iteration is the maximum number of allowable iterations.

### 4-2- Convergence operator

A novel convergence formula that contains four parent particles has been proposed in [23, 24]. Let $\rho \in [0,1]$ be a random number. If $\rho \leq P_{\text{Convergence}}$ ($P_{\text{Convergence}}$ is convergence probability), then, one of the following operators shall be performed to generate the new particle position $\bar{x}_i(t+1)$ from the old particle position $\bar{x}_i(t)$:

If fitness $\bar{x}_i(t)$ is smaller than fitness $\bar{x}_j(t)$ and fitness $\bar{x}_k(t)$ then:

$$\bar{x}_i(t+1) = \bar{x}_{\text{gbest}} + \sigma_1\left(\frac{\bar{x}_{\text{gbest}}}{\bar{x}_i(t)}\right)(2\bar{x}_k(t) - \bar{x}_j(t) - \bar{x}_i(t))$$  \hspace{1cm} (10)

If fitness $\bar{x}_j(t)$ smaller than fitness $\bar{x}_i(t)$ and fitness $\bar{x}_k(t)$ then:

$$\bar{x}_j(t+1) = \bar{x}_{\text{gbest}} + \sigma_2\left(\frac{\bar{x}_{\text{gbest}}}{\bar{x}_j(t)}\right)(2\bar{x}_k(t) - \bar{x}_j(t) - \bar{x}_i(t))$$  \hspace{1cm} (11)

If fitness $\bar{x}_k(t)$ is smaller than fitness $\bar{x}_j(t)$ and fitness $\bar{x}_i(t)$ then:

$$\bar{x}_i(t+1) = \bar{x}_{\text{gbest}} + \sigma_3\left(\frac{\bar{x}_{\text{gbest}}}{\bar{x}_k(t)}\right)(2\bar{x}_j(t) - \bar{x}_j(t) - \bar{x}_i(t))$$  \hspace{1cm} (12)

where particles $\bar{x}_i(t)$ and $\bar{x}_j(t)$ are selected from swarm by uniformly selection method. $\sigma_1$, $\sigma_2$, and $\sigma_3$ are random numbers selected from $[0,1]$ and $\bar{x}_{\text{gbest}}$ is the position of the best particle of the entire swarm. After calculating the convergence phase, the superior member between $\bar{x}_i(t)$ and $\bar{x}_i(t+1)$ should be selected. If $\rho > P_{\text{Convergence}}$, then no convergence operation is performed for $\bar{x}_i(t)$.

### 4-3- Divergence Operator

This operator, that was proposed in [23, 24], provides a possible leap on some chosen particles. Let $\theta \in [0,1]$ be a random number. If $\theta \leq P_{\text{Divergence}}$ ($P_{\text{Divergence}}$ is divergence probability) and particle $\bar{x}_i(t)$ was not enhanced by convergence operator, then the following divergence operator is performed to generate a new particle.

$$\bar{x}_i(t+1) = \text{Normrand}\left(\bar{x}_i(t), S_o\right).$$  \hspace{1cm} (13)

Normrand($\bar{x}_i(t), S_o$) generates random numbers from the normal distribution with mean parameter $\bar{x}_i(t)$ and standard deviation parameter $S_o$. ($S_o$ is a positive constant). If $\theta > P_{\text{Divergence}}$ or particle $\bar{x}_i(t)$ was enhanced by convergence operator, no divergence operation is performed.

### 4-4- Hybrid of PSO, Convergence, and Divergence Operators

It is now possible to present a novel PSO which is improved by utilizing the convergence and divergence operators to update the particle positions. Initially, particles forming the population are randomly generated. Then, convergence and divergence probabilities, the inertia weight and the learning factors are selected. In each iteration, after calculation of the fitness values of all particles, $\bar{x}_{\text{ipbest}}$ and $\bar{x}_{\text{gbest}}$ are determined. Then, for each particle a random number $\rho \in [0,1]$ would be allocated. If a particle has $\rho < P_{\text{Convergence}}$, a new particle will be produced by the convergence operator. For each particle that is not chosen for convergence operation another random number $\theta \in [0,1]$ would be allocated. If $\theta < P_{\text{Divergence}}$, then the divergence operator generates a new particle. Otherwise, the new particles that are not selected for convergence or divergence operation will be enhanced by PSO. This cycle should be repeated until the user-defined stopping criterion is satisfied [23, 24].

### Table 2. Optimization benchmark test functions

<table>
<thead>
<tr>
<th>Name</th>
<th>Formula</th>
<th>Comment</th>
<th>Search range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Noise</td>
<td>$f(x) = \sum_{i=1}^{n} x_i^2 + \text{random}[0,1]$</td>
<td>Unimodal</td>
<td>[-1.28,1.28]</td>
</tr>
<tr>
<td>Quadric</td>
<td>$f(x) = \sum_{i=1}^{n} \sum_{j=1}^{n} x_i x_j^2$</td>
<td>Unimodal</td>
<td>[-100,100]</td>
</tr>
<tr>
<td>Schwefel</td>
<td>$f(x) = \sum_{i=1}^{n} [\sin(\sqrt{</td>
<td>x_i</td>
<td>})^2]$</td>
</tr>
<tr>
<td>Griewank</td>
<td>$f(x) = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \sum_{i=1}^{n} \cos(\frac{x_i}{\sqrt{n}})+1$</td>
<td>Multimodal</td>
<td>[-600,600]</td>
</tr>
<tr>
<td>Ackley</td>
<td>$f(x) = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$</td>
<td>Multimodal</td>
<td>[-32,32]</td>
</tr>
</tbody>
</table>
4- 5- Evaluation of single objective optimization algorithm

To evaluate the accuracy of the proposed method, three renowned PSO algorithms are used for comparison, (HPSO-TVAC [22], DMS-PSO [25], and APSO [26]). Moreover, five nonlinear benchmark functions introduced in Table 2 are performed. These test functions should be minimized. In all the tests, the inertia weight $W$ is linearly decreased from $W_o = 0.9$ to $W_f = 0.4$. $C_1$ is linearly decreased from $C_1_o = 2.5$ to $C_1_f = 0.5$ while $C_2$ is linearly increased from $C_2_o = 0.5$ to $C_2_f = 2.5$ over time. The related variables used in convergence and divergence operators are: $R_{convergence} = 0.02$ and $P_{divergence} = 0.02$. Furthermore, the term $\bar{y}(t)$ is limited to the range $[-\bar{y}_{min} + \bar{y}_{max}]$ which $\bar{y}_{max} = \frac{\bar{y}_{max}}{\bar{y}_{min}}$. While the velocity violates this range, it will be multiplied by a random number between [0,1].

The mean and standard deviation fitness of the best particle for thirty runs are summarized in Table 3. The population size, maximum iteration number of periods and dimension are set at 20, 10000 and 30, respectively. It can be observed from this table that the incorporation of the convergence and the divergence operators can greatly change the performance of PSO.

From the results given in Table 3, it can be seen that the proposed method outperforms other three algorithms for different complex functions. Obviously, the technique of combination of PSO, convergence and divergence operators improves PSO and make the swarm easily escape from the local minima and converge to the global minimum, robustly.

5- Multi-objective optimization algorithm

Moore and Chapman proposed the first extension of the PSO strategy for solving multi-objective problems in an unpublished manuscript in 1999 [27]. After this early attempt, a great interest to extend PSO arose among researchers, but interestingly, the next proposal was not published until 2002. Nevertheless, there are currently different proposals of multi-objective PSOs reported in the specialized literature [28].

5-1- Definitions of multi-objective optimization problem

A multi-objective optimization problem is of the following form,

Minimize $f(x) = [f_1(x), f_2(x), \ldots, f_m(x)]$ \hspace{1cm} (14)

where $x = \{x_1, x_2, \ldots, x_n\}$ is the vector of decision variables, $f_i : R^n \rightarrow R$, $i = 1, \ldots, K$ are the objective functions. To describe the concept of optimality, some definitions should be introduced.

Definition 1. Given two vectors $x, y \in R^k$, we say that $x \preceq y$ if $x_i \leq y_i$ for $i = 1, \ldots, n$ and that $x$ dominates $y$ (denoted by $x < y$) if $x \preceq y$ and $x \neq y$ [29].

Definition 2. A vector of decision variables $\vec{x} \in \chi \subset R^n$ is non-dominated with respect to $\chi$, if there does not exist another $\vec{x}' \in \chi$ such that $f(x') < f(x)$ [29].

Definition 3. A vector of decision variables $\vec{x}' \in F \subset R^n$ ($F$ is the feasible region) is Pareto-optimal if it is non-dominated with respect to $F$ [29].

Definition 4. The Pareto optimal set $P^*$ is defined by [29]:

$p^* = \{ \vec{x} \in F \mid \vec{x} \text{ is Pareto optimal} \}$

(15)

Definition 5. The Pareto front $pF^*$ is defined by [29]:

$pF^* = \{ f(x) \in R^k \mid x \in p^* \}$

(16)

Thus, determination of the Pareto optimal set from the set $F$ of all the decision variable vectors is desired [29].

5-2- Multi-objective CDPSO

When solving single-objective optimization problems by PSO approach, $x_{best}$ is used as a leader to update particles position. However, in the case of multi-objective optimization problems, each particle might have a set of different leaders (each non-dominated solution could be selected as a leader) which only one of them can be selected in order to update its position. In this paper, we describe a leader selection technique that is based on the density measures. For this purpose, a neighborhood radius $R_{neighborhood}$ is defined for leaders. Two leaders are neighbors if their Euclidean distance (measured in the objective domain) is less than $R_{neighborhood}^{-1}$. Using this definition, the number of neighbors of each leader is calculated in the objective function domain. The particle which has fewer neighbors is preferred as leader. However, after several iterations, the leader position and its density will change. Thus, leader selection operation should be repeated and a new leader must be identified. Therefore, the maximum iteration is divided into several equal periods, and each period has the same iteration $T$. The relationship of maximum iteration, number of periods and $T$ satisfies the following equation.

$maximum\, iteration = number\, of\, periods \times T$ \hspace{1cm} (17)

In each period, the leader selection operation could be done and the non-dominated solution which has fewer neighbors is preferred as leader. Also, in the start of each iteration of a period, if a particle dominates the leader, then this particle will be considered as the new leader. This algorithm is named periodic multi-objective optimization. The proposed multi-objective method allows us to independently select inertia weight and learning factors in each period. The following equations are suggested to select the inertia weight and learning factors in each period.

$W = W_f + (W_o - W_f) \left( \frac{t}{T} - \text{fix}\left(\frac{t-1}{T}\right) \right)$ \hspace{1cm} (18)

$C_1 = C_1_f + (C_1_o - C_1_f) \left( \frac{t}{T} - \text{fix}\left(\frac{t-1}{T}\right) \right)$ \hspace{1cm} (19)

$C_2 = C_2_f + (C_2_o - C_2_f) \left( \frac{t}{T} - \text{fix}\left(\frac{t-1}{T}\right) \right)$ \hspace{1cm} (20)
The parameters of proposed multi-objective algorithm are chosen as it follows. In each period, the inertia weight $W$ is linearly decreased from $W_1 = 0.9$ to $W_2 = 0.4$. $C_1$ is linearly decreased from $C_{i1} = 2.5$ to $C_{i2} = 0.5$, and $C_2$ is linearly increased from $C_{21} = 0.5$ to $C_{22} = 2.5$ over time. The related variables used in the convergence and divergence operators are: $P_{\text{Convergence}} = 0.1$, $P_{\text{Divergence}} = 0.1$, and $S_0 = \frac{2\pi}{\nu_{max} - \nu_{min}}$. The term $\nu_0(t)$ is limited to the range $[-\nu_{max}, +\nu_{max}]$ in which $\nu_{min} = \frac{3\pi - \pi_{min}}{2}$ and $\nu_{max} = \frac{3\pi - \pi_{max}}{2}$. While the velocity violates this range, it will be multiplied by a random number between $[0,1]$. Furthermore, the positive constant for $\xi_{\text{elimination}}$ is given by $\xi = 300$ and the neighborhood radius for leader selection is as $R_{\text{neighborhood}} = 0.04$. The number of iterations in the period $T$ is 7; the swarm size and the maximum iteration are 50 and 100, respectively. The Pareto front of this multi-objective problem is shown in Fig. 2. Also, the feasibility and the efficiency of the proposed multi-objective algorithm is assessed in comparison with Sigma method [16], modified NSGAII [32] and MATLAB (R2010a) Toolbox MOGA. Although the performances of these algorithms are competitively good over this problem, the most interesting result is that the proposed algorithm has more uniformity and diversity.

In Fig. 2, points A and C stand for the best normalized summation of angles errors and normalized summation of control efforts, respectively. It is clear from this figure that all the optimum design points in the Pareto front are non-dominated and could be chosen by a designer as optimum
fuzzy tracking controllers. It is also clear that choosing a better value for any objective function in the Pareto front would cause a worse value for another objective. The corresponding values of those objective functions show an undesirable situation in comparison with the Pareto front. This implies that if any other set of decision variables is chosen, the corresponding values of the pair of those objective functions will locate a point inferior to that Pareto front. Such inferior area in the space of the two objectives is located in fact on the top right side of Fig. 2. Clearly, there are some important optimal design facts between these two objective functions which have been discovered by the Pareto optimum design approach. Such important design facts could not have been found without the use of multi-objective Pareto optimization process. From Fig. 2, point B demonstrates such important optimal design fact. Point B could be the trade-off optimum choice when considering minimum values of both the normalized summation of angles
errors and normalized summation of control efforts. Design variables and objective functions corresponding to the optimum design points A, B, and C are illustrated in Table 4. The real tracking trajectory of the optimum design points A, B, and C are shown in Figs. 3 to 5. The control effort of the optimum design points A, B, and C are also shown in Figs. 6 to 8.

7- Conclusion
This paper presented fuzzy tracking control for a walking robot that stepped purely in the lateral plane of a slope. Fuzzy control was utilized as an effective control to satisfy criteria for nonlinearity of dynamic equation and tracking system. An innovative multi-objective PSO algorithm was used to obtain the Pareto front of the non-commensurable objective functions in the design of fuzzy tracking controller. In the multi-objective optimization method, firstly, PSO was combined with convergence and divergence operators to modify converging process and also to skip every possible local optimum. Then, the periodic multi-objective optimization helped particles explore more areas in the solution space. Two conflicting objective functions, namely the normalized summation of angles errors and normalized summation of control efforts, were utilized for optimal control design. The Pareto front of innovative PSO was compared with the Pareto front of three renowned algorithms, namely modified NSGAII, Sigma method and MATLAB Toolbox MOGA. The Pareto front of innovative PSO was much more scattered than the other three and the points spread along adjacent axes. Therefore, the designer has the ample opportunity to select the finest point. Three points of Pareto front obtained from the innovative PSO were selected and the related design variables were used to illustrate the time responses of the system states. The result illustrated that the first selected point had minimum normalized summation of angles errors and maximum normalized summation of control efforts. However, the third point was opposite to the first one. It had maximum normalized summation of angles errors and minimum normalized summation of control efforts. The second well-chosen point could be the trade-off optimum choice.

Reference
Swarm Intelligence, John Wiley & Sons, Inc., 2006.


