Al-Khwarizmi Engineering Journal, Vol. 14, No. 1, March, (2018)
P.P. 77-89

Al-Khwarizmi Engineering Journal

Creating Through Points in Linear Function with Parabolic Blends Path by Optimization Method

Saad Zaghlul Saeed Al-khayyt
Department of Mechatronics / College of Engineering/ Mosul University/ Iraq
Email: alkhyaat@yahoo.com

(Received 8 May 2017; accepted 9 October 2017)
https://doi.org/10.22153/kej.2018.10.005

Abstract

The linear segment with parabolic blend (LSPB) trajectory deviates from the specified waypoints. It is restricted to that the acceleration must be sufficiently high. In this work, it is proposed to engage modified LSPB trajectory with particle swarm optimization (PSO) so as to create through points on the trajectory. The assumption of normal LSPB method that parabolic part is centered in time around waypoints is replaced by proposed coefficients for calculating the time duration of the linear part. These coefficients are functions of velocities between through points. The velocities are obtained by PSO so as to force the LSPB trajectory passing exactly through the specified path points. Also, relations for velocity correction and exact velocity solution are derived. Simulation results show that the engagement of modified LSPB trajectory with PSO to work well on the tested cases. This proposed method is very simple which can be used for on-line path planning, and not necessarily to use high acceleration magnitude.

Keywords: *Adaptive inertia weight*, *Linear segment with parabolic blend*, *Particle swarm optimization*, *Robot manipulator*, *Through point*, *Trajectory generation.*

1. Introduction

Straight line segments is the output from motion planners. This path has velocity discontinuity at waypoints. To achieve an efficient execution on the robot, blends are added to ensure a smooth transition between segments [1]. A common trajectory for industrial robots is the linear segment with parabolic blend (LSPB) [2, 3]. The LSPB needs only the initial and final joint angles, traveling time, and either angular acceleration or angular velocity. Numerous methods and algorithms have been established which generate such trajectories with velocity, acceleration, and jerk limitations [4, 5].

In LSPB, it is required to use high acceleration's magnitude to be quite close to the desired waypoint. Time-optimal solution for time durations of LSPB so as to satisfy the constraints velocity and acceleration is presented in [6]. This requires calculation a factor for velocity reduction of two neighboring linear segments in order to prevent overlapping of blend phases. Yet, this ethomd can lead to very slow trajectories [7]. Rymansaib et al. used a series of time-delayed third-order exponential function to generate an approximation to the trapezoidal velocity profile of LSPB [8]. The motion duration is affected by high blending percentage as well as the corresponding accuracy at waypoints. A new technique "envelope of tangents planning" had been developed so as to generate trajectory that reaches waypoints in specified moments of time [9]. This is achieved by assigning positions and tangential velocities that the joints must have when the end-effector passes through each of those waypoints. Additionally, the online trajectory generation algorithm was combined with the Reflexxes libraries to make the trajectory reaches waypoint with continuous acceleration and jerk [10, 11].

New path planning algorithm was developed for the control of an XY -motion stage using LSPB and minimum time trajectory for an aerosol printing system [12]. The algorithm calculates blend times and constant velocity based on given trajectory conditions. But large error in speed appeared for acute angle trajectory. This problem can be considered as multi-segment trajectories without stopping at waypoints. Weber et al. used the tool CorDe (Corner Drive with Defined Speed) to visualize the characteristic of the distance to the corner depending on the speed deviation for transition between path segments [13]. But the absolute maximum acceleration is difficult to obtain as a realistic value which is the sole characteristic value. This results in a loss on performance for many transition points.

Classic optimization approaches suffer from many drawbacks, such as high time complexity in high dimensions and trapping in local minima, which make them inefficient in practice. Modern or nontraditional optimization methods such as genetic algorithm and swarm intelligence are widely used in path planning problems [14-16]. Particle swarm optimization (PSO) is simple and fast because it has few parameters to be adjusted [17].

In the above mentioned works, the suggested methods have limitations and use the same basic equations of LSPB. The LSPB trajectory still deviates from the specified waypoints. In this work, the LSPB trajectory is modified and engaged with PSO. The novelty in this work is to modify LSPB trajectory using two coefficients for calculating the time duration of the linear part in LSPB trajectory. These coefficients are functions of velocities between through points. The velocities are obtained by PSO to force the LSPB trajectory passing exactly through the specified path points. Also, relations for velocity correction and exact velocity solution are derived.

2. Multisegment Linear Path with Blends

For the case in which there are many waypoints, linear segments with parabolic blends are considered. In LSPB, the segment is divided into three parts: parabolic, linear, and parabolic; respectively (Fig.1).

Considering the path waypoints which are *j*, *k*, and *1*. The time duration for blend region of point *k* is t_k. The time duration for linear part between points *j* and *k* is t_{jk}. The time duration of the segment which connects points *j* and *k* is t_{djk}. According to Fig. 1, the linear velocity between points *j* and *k* is $\dot{\theta}_{jk}$, the acceleration at point *k* is $\ddot{\theta}_k$, and the path point position is θ_k. The blend times t_k is computed from the given: path points θ_k, desired time durations t_{djk}, and the magnitude of acceleration $|\ddot{\theta}_k|$. For interior path points, this follows simply the equations [2]:

$$\dot{\theta}_{jk} = \frac{\theta_k - \theta_j}{t_{djk}} \quad \ldots(1.a)$$

$$\ddot{\theta}_k = SGN(\dot{\theta}_{kl} - \dot{\theta}_{jk}) |\ddot{\theta}_k| \quad \ldots(1.b)$$

$$t_k = \frac{\dot{\theta}_{kl} - \dot{\theta}_{jk}}{\ddot{\theta}_k} \quad \ldots(1.c)$$

$$t_{jk} = t_{djk} - \frac{1}{2}t_j - \frac{1}{2}t_k \quad \ldots(1.d)$$

where *SGN* ( ) returns the sign of the value in the brake.

In the first and last segments there is an entire blend region at one end of the segment. For the first segment [2]:

$$\ddot{\theta}_1 = SGN(\theta_2 - \theta_1) |\ddot{\theta}_1| \quad \ldots (2.a)$$

$$t_1 = t_{d12} - \sqrt{t_{d12}^2 - \frac{2(\theta_2 - \theta_1)}{\ddot{\theta}_1}} \quad \ldots(2.b)$$

$$\dot{\theta}_{12} = \frac{\theta_2 - \theta_1}{t_{d12} - \frac{1}{2}t_1} \quad \ldots(2.c)$$

$$t_{12} = t_{d12} - t_1 - \frac{1}{2}t_2 \quad \ldots(2.d)$$

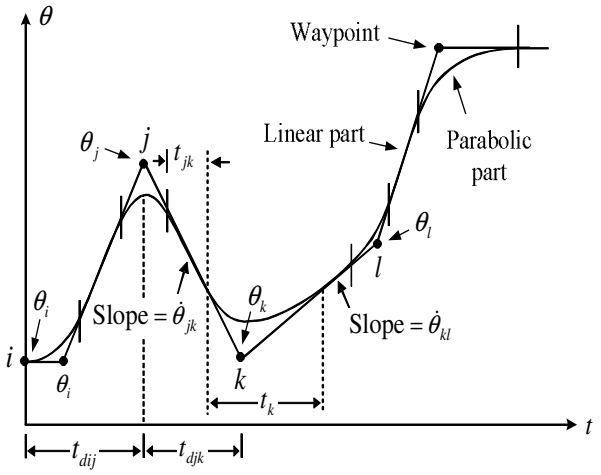

Fig. 1. Multisegment linear path with blends [2]

Likewise, for the last segment (the one connecting points *n*-1 and *n*), which leads to the solution:

$$\ddot{\theta}_n = SGN(\theta_{n-1} - \theta_n) |\ddot{\theta}_n| \quad \ldots(3.a)$$

$$t_n = t_{d(n-1)n} - \sqrt{t_{d(n-1)n}^2 + \frac{2(\theta_n - \theta_{n-1})}{\ddot{\theta}_n}} \quad \ldots(3.b)$$

$$\dot{\theta}_{(n-1)n} = \frac{\theta_n - \theta_{n-1}}{t_{d(n-1)n} - \frac{1}{2}t_n} \quad \ldots(3.c)$$

$$t_{(n-1)n} = t_{d(n-1)n} - t_n - \frac{1}{2}t_{n-1} \quad \ldots(3.d)$$

In these linear-parabolic-blend splines, when acceleration capability is sufficiently high, the paths will come quite close to the desired waypoint. The manipulator must come to a complete stop if it is desired to pass exactly through a waypoint. The term "through point", as it was mentioned in [2], will be used in the next sections to specify a path point through which the manipulator is forced to pass exactly.

3. Problem Definition

In the previous section, the LSPB trajectory is constrained to the following [6]:

1-The velocity at the first and last through point must be zero.

2-The velocity and acceleration of the trajectory must be:

$|\dot{\theta}(t)| \le \dot{\theta}_{\max} \wedge |\ddot{\theta}(t)| \le \ddot{\theta}_{\max}$.

The limitations of this algorithm are:

1-Sufficiently large acceleration is required so as to obtain linear portion in the segment.

2-The manipulator's velocity must be zero so as to pass into waypoints.

3- The system should generate two pseudo points so as to make the manipulator passes exactly through a path point without stopping.

4-The parabolic portion is assumed to be centered equally in time about through point.

This later assumption makes the apex of parabolic part to be shifted away from through point as shown in Fig. 2.

For interior path point (equation (1)), the apex of the parabolic portion is not equally centered in time about waypoint. This can be easily proved using the kinematic equations for LSPB trajectory as presented below.

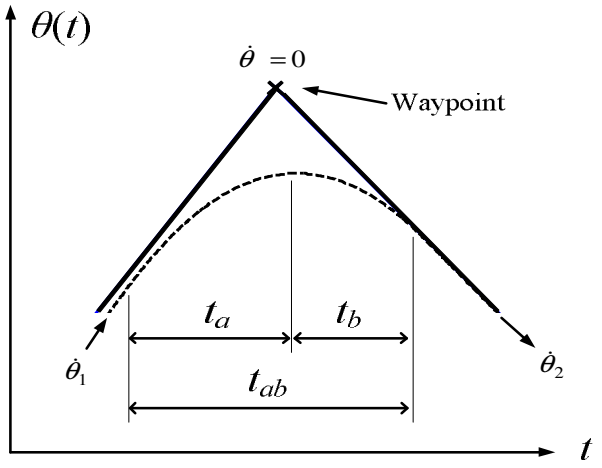

Fig. 2. Trajectory of parabolic part.

Proof 1:

Given $\dot{\theta}_1 > 0,\ \dot{\theta}_2 < 0, \text{and} \ddot{\theta} < 0$, the velocities at the parabolic path for constant acceleration are:

$$-\dot{\theta}_2 = \dot{\theta}_1 - \ddot{\theta}\, t_{ab} \quad \ldots(4.a)$$

$$\dot{\theta} = \dot{\theta}_1 - \ddot{\theta}\, t_a \quad \ldots(4.b)$$

$$-\dot{\theta}_2 = \dot{\theta} - \ddot{\theta}\, t_b \quad \ldots(4.c)$$

where $\dot{\theta}_1, \dot{\theta}, \text{and } \dot{\theta}_2$ are the velocities of previous segment, apex point, and next segment; respectively. But when the trajectory changes its direction, the apex point's velocity becomes zero. The above equations of velocities are solved for unknown time durations as:

$$t_a = \dot{\theta}_1 / \ddot{\theta} \quad \ldots(5.a)$$

$$t_b = \dot{\theta}_2 / \ddot{\theta} \quad \ldots(5.b)$$

$$\therefore t_a \neq t_b \quad \ldots(5.c)$$

Therefore the apex of parabolic path is not exactly placed under a waypoint unless $\dot{\theta}_1 = \dot{\theta}_2$ in magnitude.

4. Proposed Method

The assumption of normal LSPB that parabolic portion is centered in time around waypoints is replaced in this work by proposed coefficients which are functions of velocities between through points. From equations (4 and 5), the time durations on the parabolic blend around through point are obtained as:

$$t_a / t_{ab} = \frac{\dot{\theta}_1 / \ddot{\theta}}{(\dot{\theta}_1 + \dot{\theta}_2) / \ddot{\theta}}$$

$$t_a / t_{ab} = \frac{\dot{\theta}_1}{(\dot{\theta}_1 + \dot{\theta}_2)} \quad \ldots(6.a)$$

$$t_b / t_{ab} = \frac{\dot{\theta}_2}{(\dot{\theta}_1 + \dot{\theta}_2)} \quad \ldots(6.b)$$

From equation (6), now two coefficients (α_j & α_k) are obtained for calculating the time duration of the linear portion of the trajectory. These coefficients are obtained from the LSPB kinematic equations. Two coefficients are used to calculate the time's duration of the parabolic portions in the segment's time duration (t_{djk}).

Given through points (joint's angles), time durations, and assuming accelerations for all through points, the modified LSPB equations are as below:

Modified mid segments

$$t_k = (\dot{\theta}_{kl} - \dot{\theta}_{jk}) / \ddot{\theta}_k \quad \ldots(7.a)$$

$$t_{jk} = t_{djk} - \alpha_j t_j - \alpha_k t_k \quad \ldots(7.b)$$

$$\alpha_j = \left|\dot{\theta}_{jk}\right| / (\left|\dot{\theta}_{ij}\right| + \left|\dot{\theta}_{jk}\right|) \quad \ldots(7.c)$$

$$\alpha_k = \left|\dot{\theta}_{jk}\right| / (\left|\dot{\theta}_{jk}\right| + \left|\dot{\theta}_{kl}\right|) \quad \ldots(7.d)$$

Modified first segment

$$t_1 = \dot{\theta}_{12} / \ddot{\theta}_1 \quad \ldots(8.a)$$

$$t_{12} = t_{d12} - t_1 - \alpha_{k=2} t_2 \quad \ldots(8.b)$$

$$\alpha_2 = \left|\dot{\theta}_{12}\right| / (\left|\dot{\theta}_{12}\right| + \left|\dot{\theta}_{23}\right|) \quad \ldots(8.c)$$

Modified last segment

$$t_n = -\dot{\theta}_{(n-1)n} / \ddot{\theta}_n \quad \ldots(9.a)$$

$$t_{(n-1)n} = t_{d(n-1)n} - \alpha_{j=n-1} t_{n-1} - t_n \quad \ldots(9.b)$$

$$\alpha_{n-1} = \left|\dot{\theta}_{(n-1)n}\right| / (\left|\dot{\theta}_{(n-2)(n-1)}\right| + \left|\dot{\theta}_{(n-1)n}\right|) \quad \ldots(9.c)$$

By using these coefficients, the apex of parabolic portion is positioned exactly under the through point. Although the apex of parabolic part is now positioned exactly under through point, but it stills not passing through it. This is because of replacing a linear part region of the segment by parabolic part. This can be proved as below.

Proof 2:

The position for linear part with constant velocity during a time t is

$$\theta_l = \dot{\theta}_l \cdot t \quad \ldots(10)$$

and that for parabolic part with constant acceleration in the same time duration is:

$$\theta_p = \dot{\theta}_l \cdot t - \frac{1}{2} \ddot{\theta} t^2 \quad \ldots(11)$$

The difference ($\Delta\theta$) between these parts is:

$$\Delta\theta = \theta_p - \theta_l \quad \ldots(12)$$

Substituting equations (10) and (11) into equation (12) gives:

$$\Delta\theta = \dot{\theta}_l \cdot t - \frac{1}{2} \ddot{\theta} t^2 - \dot{\theta}_l \cdot t$$

$$\Delta\theta = -\frac{1}{2} \ddot{\theta} t^2 \quad \ldots(13)$$

which means this displacement of parabolic part is less than that of linear part for same time duration.

This problem can be solved by increasing the velocity between through points in presence of acceleration limits. The problem of finding the velocities' value can be solved by optimization methods such as PSO.

4.1. Velocity Correction

In the above modified LSPB, the initial velocities are obtained from equation (1). But from proof 2, there is an error due to the parabolic part (equation (13)). Therefore, these velocities can be corrected by adding the change of velocity (equation (14.b)) based on the error of the parabolic parts as:

$$(\dot{\theta}_{jk})_{\text{corrected}} = \dot{\theta}_{jk} + \Delta\dot{\theta}_{jk} \quad \ldots(14.a)$$

$$\Delta\dot{\theta}_{jk} = SGN\,(\dot{\theta}_{jk})\, \Delta\theta_{jk} / t_{jk} \quad \ldots(14.b)$$

$$\Delta\theta_{jk} = \frac{1}{2} \left(\left|\ddot{\theta}_j\right| \cdot (\frac{1}{2} t_j)^2 + \left|\ddot{\theta}_k\right| \cdot (\frac{1}{2} t_k)^2\right) \quad \ldots(14.c)$$

In the above equation, half of the blend durations (t_j and t_k) are accepted as an approximated value. This corrected velocity is used as initial velocity to begin the optimization process.

4.2 Exact Solution of Velocity

The LSPB offers exact solution of velocity for path segment when there is acceleration from zero velocity to linear velocity and deceleration to zero velocity. For example, the first and second segments (ij and jk) or the second and third segments (jk and kl) have such situation (Fig.1). The second and third segments are considered to derive a general solution. Let t_s, t_l, and t_e are times' duration for first parabolic part, linear part, and last parabolic part for the path from through point j to through point k. The total displacement is the summation of these three parts as below:

$$\left|\theta_k - \theta_j\right| = \left|\ddot{\theta}_j\right| t_s^2 / 2 + \dot{\theta}_{exact}\, t_l + \left|\ddot{\theta}_k\right| t_e^2 / 2 \quad \ldots(15.a)$$

$$t_l = t_{djk} - t_s - t_e \quad \ldots(15.b)$$

$$t_s = \dot{\theta}_{exact} / \left|\ddot{\theta}_j\right| \quad \ldots(15.c)$$

$$t_e = \dot{\theta}_{exact} / \left|\ddot{\theta}_k\right| \quad \ldots(15.d)$$

Solving the above quadratic equation (equation (15.a)) for the exact velocity to obtain:

$$\dot{\theta}_{exact} = \frac{+t_{djk} - \sqrt{t_{djk}^2 - 2\left|\theta_k - \theta_j\right| \cdot (\left|\ddot{\theta}_j\right| + \left|\ddot{\theta}_k\right|) / (\left|\ddot{\theta}_j\right| \cdot \left|\ddot{\theta}_k\right|)}}{(\left|\ddot{\theta}_j\right| + \left|\ddot{\theta}_k\right|) / (\left|\ddot{\theta}_j\right| \cdot \left|\ddot{\theta}_k\right|)} \cdot SGN(\theta_k - \theta_j) \quad \ldots(16)$$

In the above derivation, absolute values are used to prevent sign confusion. The sign of velocity is introduced after solving the quadratic equation.

4.3. Steps of Proposed Method

The proposed modified LSPB is presented here which overcomes the limitations of the normal LSPB. It provides logic sequence for computer programming to generate through points:

Step 1: Calculate velocities and accelerations according to equation (1).

Step 2: Use equation (16) to solve for exact velocity if there is a change in velocity direction between two following through points.

Step 3: Obtain the time durations for the trajectory using equations (7-9).

Step 4: Apply velocity correction using equation (14) to all calculated velocities at step 1 except that obtained at step 2.

Step 5: Use these velocities as initial solution for the optimization process.

This algorithm can be easily converted into computer program to perform optimization process (Fig. 3). The velocities are obtained from optimization process so as to force the LSPB trajectory passing exactly through specified path points.

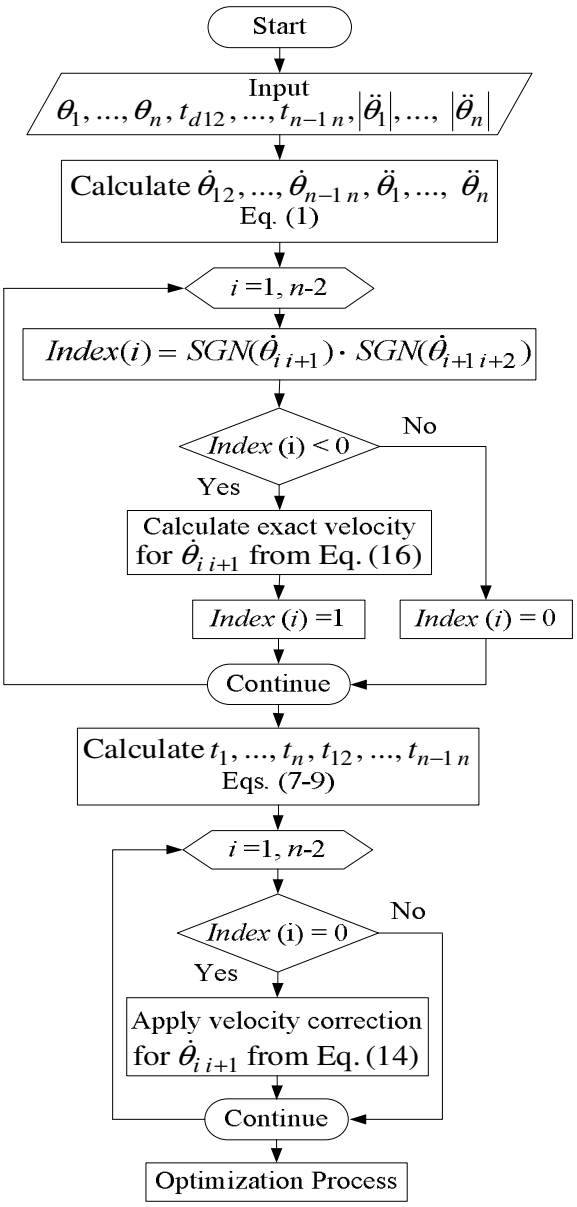

Fig. 3. Algorithm flowchart for proposed modified LSPB trajectory.

5. Particle Swarm Optimization

Recently, modern or nontraditional methods of optimization are widely used for solving different optimization problems. Edward and Kennedy formulated PSO in 1995. The process was inspired by the social behavior of animals, such as bird flocking or fish schooling. Each particle has two characteristics which are: a position and a velocity. It must remember the best position in terms of objective function value. The particles adjust their individual positions and velocities by sharing the received information of the best position [18]:

$$\mathbf{v}_{i,j}^{new} = w\mathbf{v}_{i,j}^{old} + \Gamma_1 \times r_1 \times (\mathbf{p}_{i,j}^{local\ best} - \mathbf{p}_{i,j}^{old}) + \Gamma_2 \times r_2 \times (\mathbf{p}_{i,j}^{global\ best} - \mathbf{p}_{i,j}^{old}) \quad \ldots(17.a)$$

$$\mathbf{p}_{i,j}^{new} = \mathbf{p}_{i,j}^{old} + \mathbf{v}_{i,j}^{new} \quad \ldots(17.b)$$

where **v** is particle's velocity, w is inertia weight, **P** is particle position or variable, r_1 and r_2 are uniform distributed random numbers, $\mathbf{P}^{local\ best}$ is best local postion, $\mathbf{P}^{global\ best}$ is best global position, i is particle index, j is dimension of variable, Γ_1 is individual learning rate, Γ_2 is group learning rate.

Premature and local optimum convergence are the disadvantages of traditional PSO. Modifications to PSO were applied so as not to skip the optimal solution [19]. These modifications are happened either on basic components or on swarms itself. Modifications on basic components of PSO are: inertia weight [20, 21], velocity constriction [22, 23], and velocity clamping [24]. Five basic benchmark optimization functions had been tested by using fifteen different inertia weight variants in [25]. They concluded that chaotic inertia weight improves the accuracy of the solution. Modifications on the swarms itself are: insertions new swarms [26, 27], mutation [28], and swarm initiation [29]. These modifications can increase the search diversity. Applying multiple modifications on the basic components of PSO and swarms was suggested as future work in [30]. An improved chaotic PSO algorithm based on adaptive inertia weight (AIWCPSO) was proposed in [31]. Initially, the positions and velocities of the population are generated by using chaotic mapping. The inertia weight is adjusted according to the values of: iterative number, aggregation degree factor, and improved evolution speed parameter. AIWCPSO algorithm with chaotic swarm initiation and swarm injection were used in [32]. This combines modifications to basic components of PSO and swarms.

In this work, improved chaotic PSO algorithm based on adaptive inertia weight (AIWCPSO) is used. Also velocity constriction factor, λ, is included (equation (18)).

$$\mathbf{p}_{i,j}^{new} = \mathbf{p}_{i,j}^{old} + \lambda\, \mathbf{v}_{i,j}^{new} \quad \ldots(18)$$

Steps of AIWCPSO Algorithm

Step 1: Cubic mapping (equation (19)) is used to generate double or triple swarm size as chaotic initialization. The cubic mapping is described as following:

$$\begin{cases} p_{n+1} = 4\, p_n^3 - 3\, p_n \\ -1 < p_0 < 1 \end{cases} \quad \ldots(19)$$

where p_0 is a random number substituted as initial value. These swarms are tested so as to select those of best fitness as initial solution to particle position. Then this initial solution is mapped to the search space range.

Step 2: Chaotic initialization is also applied to generate N initial velocities by cubic mapping (equation (19)).

Step 3: The inertia weight is updated by the equations below for a single particle:

$$w_i = w_{\max} - (w_{\max} - w_{\min}) \frac{iter}{iter_{\max}}\, e^{-(\alpha\, esf_i - \beta\, adf)} \quad \ldots(20.a)$$

$$esf_i = \frac{abs\,[F(pbest_i^k) - F(pbest_i^{k-1})]}{abs\ F(pbest_i^k) + abs\ F(pbest_i^{k-1})} \quad \ldots(20.b)$$

$$adf = \frac{abs\,[\min\,(F_{best}^k,\ F_{avg}^k)]}{abs[\max\,(F_{best}^k, F_{avg}^k)]} \quad \ldots(20.c)$$

where k is the current iteration value. $w_{\max}$ and $w_{\min}$ are maximum and minimum values of inertia weight; respectively. $iter$ is the current iteration number and $iter_{\max}$ is the maximum number of iteration. The value α and β has the range [0,1]. esf_i is the improved evolution speed parameter of particle i (i =1, 2,..., N), adf is the aggregation degree factor of swarm, F ($pbest_i^k$) is the best fitness value of particle i at the k^{th} iteration, F_{best} is the best fitness obtained from all particles, F_{avg} is the mean fitness of all particles in the swarm at the same iteration.

Step 4: The variance (σ) is calculated for the population's fitness (equation (21)). When variance is less than threshold value and the optimal fitness of current iteration is worse than the desired fitness value, chaotic disturbance is applied.

$$\sigma^2 = \sum_{i=1}^{N} \left(\frac{F(x_i^k) - F_{avg}^k}{\max\,[\max\,[abs[F(x_i^k) - F_{avg}^k]],1]}\right)^2 \quad \ldots(21)$$

where $F(x_i)$ is fitness values of particle i.

Step 5: Chaotic disturbance strategy:

Cubic mapping (equation (19)) is used to generate chaotic vector o_{ij} (i =1, 2,..., N; j =1,2,..., J), where o_{0j} is (-1,1) of random numbers, and the component of this vector is loaded to the range of chaotic disturbance of $[-\gamma_j, \gamma_j]$ (j=1, 2,..., J). Then chaotic disturbance variation is $\Delta p_i = (\gamma_1\, o_{i1}, \gamma_2\, o_{i2}, \ldots, \gamma_J\, o_{iJ})$. The position updated of particle after adding the chaotic disturbance variation is given by: $pb_{ij}\ (k+1) = p_{ij}\ (k) + v_{ij}\ (k) + \Delta p_{ij}$. Finally, comparison is made between the fitness values of $F(pb_i\ (k+1))$ and $F\ (p_i\ (k+1))$. If $F(pb_i\ (k+1))$ is

better than $F(p_i\ (k+1))$, then $p_i\ (k+1)$ is updated by $pb_i\ (k+1)$. Note: J is the variable's dimension of particle i. For more details about AIWCPSO algorithm see [31].

6. Simulation Results

Simulations are presented to validate the proposed method. These simulations are implemented in Matlab7 on Pentium 4 PC processor (Intel (R) Core (TM) i5-2450M CPU @ 2.50 GHz). The parameters of PSO are set as follows: Γ_1=2.05; Γ_2=2.05; λ=0.7298 [22]; N= 40; $iter_{max}$= 30; w_{max}=0.09, w_{min}= 0.05; α=0.99; β=0.01; γ=10^{-4}; threshold value= 10^{-5}; desired fitness value =10^{-12}. The objective function to be minimized has the form:

$$\text{Fitness}=\sum_{i=2}^{np} e_i^2 \qquad \ldots(22)$$

where e is the error at through point, np is total number of through points. The error at through point 1 is always zero because it is starting point. Considering a single joint: Through points of the path in degrees: 10, 35, 25, 10. The time durations of the segments are: 2, 1, 3 seconds; respectively. The acceleration at blend points is 50 degrees/second^2. Although different accelerations at different through points can be used.

At first, the trajectories of normal LSPB (equations (1-3)) and modified LSPB (equations (7-9)) are compared (Fig. 4). The normal LSPB trajectory is shifted while the modified is equally spaced about through points. A comparison of linear part velocities is presented in **Table 1** for 50 degrees/second^2 acceleration between normal LSPB, modified LSPB with velocity correction, and optimum modified LSPB (section 4.3). It is clear that modified LSPB with velocity correction is better choice as initial velocity to begin the optimization process. A range between 0.9 to 1.2 of these later calculated velocities is used as initial value for the optimization. This will reduce the number of iterations to reach optimum solution.

PSO is used to generate optimal linear velocities for the linear parts. In fact, that increasing the velocities of the linear portions will compensate the error resulted from inserting parabolic region in the path between two neighboring through points. Figure 5 shows comparison between the normal and modified LSPB trajectories using PSO method. The normal LSPB trajectory deviates from the through points, while the engagement of modified LSPB with PSO (proposed method in section 4.3) passes into them. In the normal LSPB method, it is restricted to use acceleration's value higher than 40 degrees/second^2 [2]. Figure 6 shows results of proposed method for the two acceleration values using PSO. The results of comparison are presented in Table 2 for different acceleration values. Acceleration values are: 30, 50, 70 degrees/second^2.

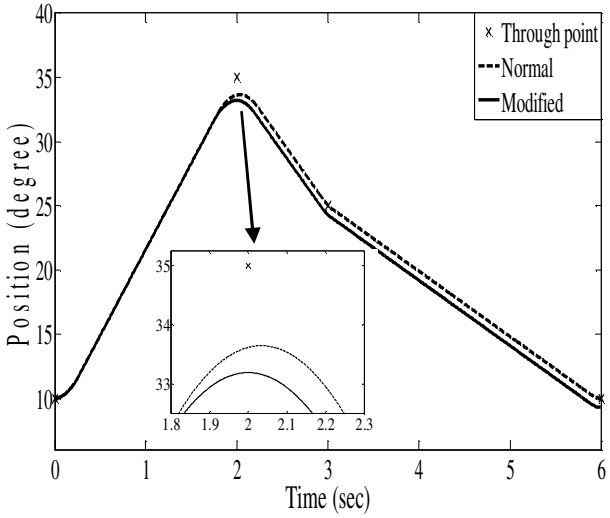

Fig. 4. Trajectories of normal and modified LSPB

Table 1,
Comparision of linear velocities.

Method / Velocity	Normal LSPB (eqs.1-3)	Modified LSPB & velocity correction (eqs. 7-9&14)	Modified LSPB & PSO (eqs. 7-9) & (eqs. 14, 16-24)
$\dot{\theta}_{12}$ (deg/sec)	13.3975	14.3854	14.6446
$\dot{\theta}_{23}$ (deg/sec)	-10.0000	-11.8111	-11.5306
$\dot{\theta}_{34}$ (deg/sec)	-5.0862	-5.1090	-5.0728
t_1 (sec)	0.2679	0.2877	0.2929

The error at through points is reduced for the cases of proposed method. The error at point 2 is zero because of using the exact velocity solution (equation (16)). The error at point 3 is $0.2916\cdot10^{-12}$, $0.3588\cdot10^{-12}$, and $0.0107\cdot10^{-12}$ degree for accelerations 30, 50, and 70 degrees/second^2; respectively. Optimal velocities' value are presented in Table 3.

Table 2,
Comparison of error at through points between normal and modified LSPB using PSO.

Case	Acceleration (deg/sec^2)	e_2 (deg)	e_3 (deg)	e_4 (deg)
Normal LSPB (eqs.1-3) & PSO (eqs. 17-22)	30	0.7919	0.9480	0.6877
	50	0.3785	0.5844	0.5021
	70	0.2539	0.4010	0.3431
Proposed method (eqs. 7-9 & eqs. 14, 16-24)	30	$0.0003\cdot10^{-12}$	$0.2916\cdot10^{-12}$	$0.3038\cdot10^{-12}$
	50	$0.0002\cdot10^{-12}$	$0.3588\cdot10^{-12}$	$0.5524\cdot10^{-12}$
	70	$0.0001\cdot10^{-12}$	$0.0107\cdot10^{-12}$	$0.5240\cdot10^{-12}$

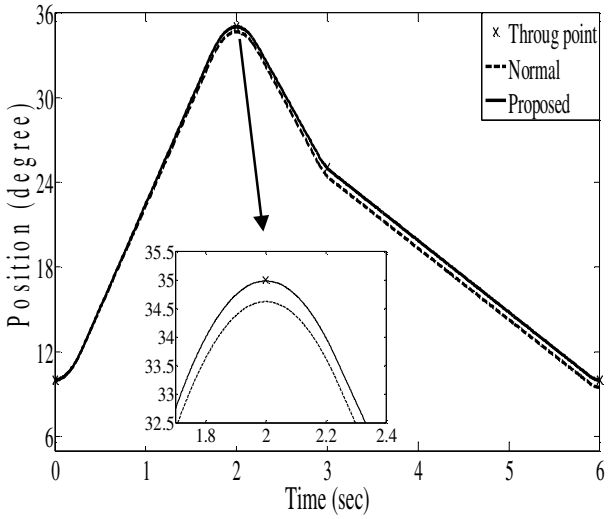

Fig. 5. Comparison between normal and modified LSPB trajectory using PSO.

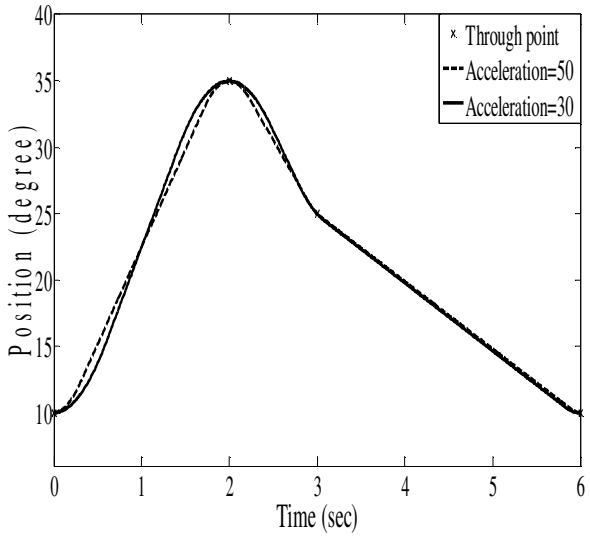

Fig. 6. Trajectories obtained by proposed method.

Table 3,
Optimal values obtained by proposed method.

Variable	Acceleration (deg/sec^2)		
	30	**50**	**70**
$\dot{\theta}_{12}$ (deg/sec)	17.7526	14.6446	13.8751
$\dot{\theta}_{23}$ (deg/sec)	-13.9218	-11.5307	-10.9786
$\dot{\theta}_{34}$ deg/sec)	-5.1142	-5.0728	-5.0525
t_1 (sec)	0.5918	0.2929	0.1982

The run execution time is found to be about 5.4% of the first parabolic portion (0.2929 seconds) for the 50 degrees/second^2 acceleration. This value is enough to generate trajectory of desired path by using the algorithm of the proposed method for on-line path planning.

Point to point position trajectory is simulated for two-link planar robot manipulator (Fig. 7). It has revolute joints. The masses m_1 and m_2 are assumed to be concentrated at the distal end of the links which have the lengths l_1 and l_2; respectively. The robot starts at point (2.95, 0.05) and passes through points (2.45, 0.05) and (2.7, 0.30). Then it stops at point (2.95, 0.05). All coordinates are in meters. The time durations are: 4, 3, 3 seconds. The robot dynamic equation and controller are taken from early published paper [33]. The desired through points in joint space are obtained by solving the inverse kinematics equations [2]. The trajectory for the two joints is generated by using the proposed method (Fig.8).

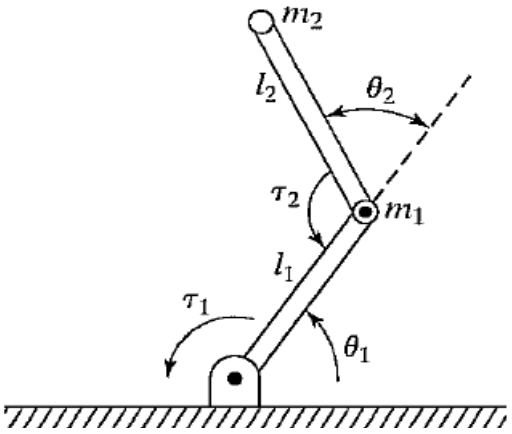

Fig. 7. Planar robot arm with two links [2]

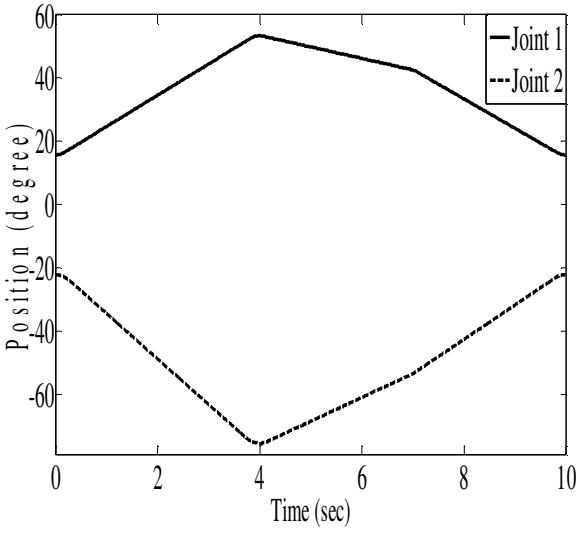

Fig. 8. Two-link joint's desired trajectory obtained by proposed method.

The Cartesian trajectory of the two-link manipulator is presented in Fig. 9. The errors in the Cartesian space are: $0.5722 \cdot 10^{-5}$, $0.4180 \cdot 10^{-5}$, $0.4365 \cdot 10^{-5}$, and $0.2551 \cdot 10^{-5}$ in meters for these through points. The error at the starting point is not zero because of the used controller which is learning on-line. The error at through points is presented in Table 4.

Finally, a comparison of results is presented for path of acute angles from [12]. The pattern contains 14 line segments, and beginning at the origin. The segments form angles starting at 125° and decrease linearly to 5°. The peak velocity error shown in Fig. 10 increases as angle decreases. For example, a complete change in direction or a 0° angle results in a 100% error according to the method of [12]; while the error is no more 21% by using the proposed method in this work. In LSPB, the trajectory deviates slightly from the straight path. This deviation is increased as the angle formed by two segments is decreased (Fig. 11).

Table 4,
Error at through points for point to point trajectory using the proposed method

Link	Error (deg)		
	e_2	e_3	e_4
1	$0.0071 \cdot 10^{-12}$	$0.2061 \cdot 10^{-12}$	$0.1474 \cdot 10^{-12}$
2	$0.0284 \cdot 10^{-12}$	$0.1137 \cdot 10^{-12}$	$0.0675 \cdot 10^{-12}$

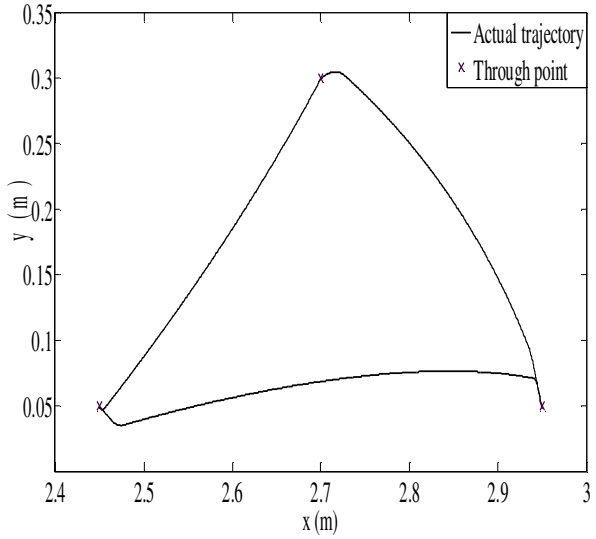

Fig. 9. Two-link Cartesian path obtained by proposed method.

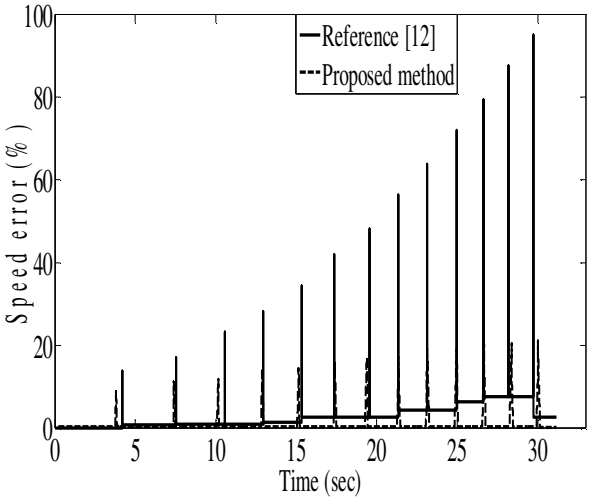

Fig. 10. Comparison of Speed error resulting from decreasing segment angle

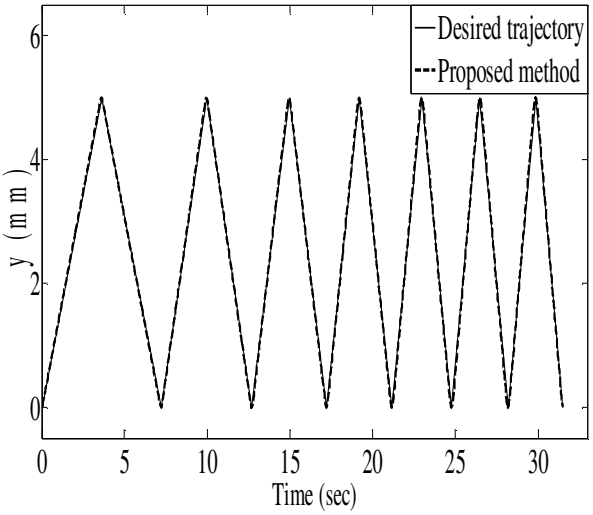

Fig. 11. Trajectory of y-component with decreasing segment angles.

7. Conclusions

The normal LSPB is restricted to that the acceleration must be sufficiently high. In this work, modified LSPB is engaged with PSO for generating smooth valid trajectories that passes through specified path points while satisfying velocity and acceleration constraints of physical mechanical robot manipulator.

Increasing the velocity of linear portions compensates the error due to inserting parabolic part. Velocity correction is used to obtain close values to the optimal solution. This reduces the number of iterations to obtain the optimum solution. Also, exact solution of velocity can be used for LSPB path segment when there is acceleration from zero velocity to linear velocity and deceleration to zero velocity.

Simulation results show that the proposed method to work well on the tested cases. The error at through points is almost zero. Advantages of the modified LSPB algorithm are: through points can be created, easily engaged with optimization method, very simple which can be used for on-line path planning, and not necessarily to use high acceleration's magnitude. The proposed method in this work has no chance to fail to create through points within a reasonable number of iterations.

8. References

[1] Ellekilde, L. P. and Petersen, H. G. (2013). Motion planning efficient trajectories for industrial bin-picking. International Journal of Robotics Research, 32(9-10): 991-1004.

[2] Craige, J. J. (2005). Introduction to robotics: mechanics and control. New Jersey. Person Prentice-Hell, Inc.

[3] Siciliano, B., Sciavicco, L., Villani, L. and Oriolo, G. (2009). Robotics: modelling, planning and control. London. Springer-Verlag: pp.168.

[4] Ata, A. A. and Sa'adah, M. Y. (2006). December 8. Soft motion trajectory for planar redundant manipulator. 9th International Conference on Control, Automation, Robotics and Vision. Singapore.

[5] Xianhua, L., Shili, T., Xiaowei, F. and Hailiang, R. (2009). December 19-20. LSPB Trajectory planning: design for the modular robot arm applications. IEEE International Conference on Information Engineering and Computer Science. Wuhan. DOI: 10.1109 / ICIECS. 2009.5365861:1-4.

[6] Kunz, T. and Stilman, M. (2011). Turning paths into trajectories using parabolic blends. GT-GOLEM-2011-006. Georgia Institute of Technology.

[7] Kunz, T. and Stilman, M. (2012). Time-optimal trajectory generation for path following with bounded acceleration and velocity. Proceedings of Robotics: Science and Systems. Sydney, Austria. DOI: 10.15607 / RSS.2012.VIII.027: 9-13.

[8] Rymansaib, Z., Iravani, P. and Sahinkaya, M.N. (2013). July 9-12. Exponential trajectory generation for point to point motions. IEEE/ASME International Conference on Advanced Intelligent Mechatronics, Wollongong-Australia, pp.906-911.

[9] Rossi, C. and Savino, S. (2013). Robot trajectory planning by assigning positions and tangential velocities. ELSEVER: Robotics and Computer-Integrated Manufacturing. 29: 139-156.

[10] Kröger, T. and Wahl, F. M. (2010). Online trajectory generation: basic concepts for instantaneous reactions to unforeseen events. IEEE Transactions on Robotics, 6(1): 94-111.

[11] Kröger, T. (2012). May 14-18. On-line trajectory generation: Nonconstant motion constraints. International Conference on Robotics and Automation, River Center, Saint Paul, Minnesota, USA. DOI: 10.1109 / ICRA. 2012.6225186: 2048-2054.

[12] Thompson, B. and Yoon, H.-S. (2014). Efficient Path Planning Algorithm for Additive Manufacturing Systems. IEEE Transactions on components, packaging and

manufacturing technology, 4(9): 1555-1563, DOI: 10.1109/TCPMT.2014.2338791

[13] Weber, W., König, A., Nodem, D. X., Darmstadt, H. (2016). June 21 – 22, 2016. User-defined transition between path segments in terms of tolerances in speed and position deviation. Proceedings of ISR 2016, 47st International Symposium on Robotics, Munich-Germany, pp.187-193.

[14] Masehian, E. and Sedighizadeh, D. (2010). Multi-objective PSO- and NPSO-based algorithms for robot path planning. Advances in Electrical and Computer Engineering, 10(4): 69-76.

[15] Bailón1, W. P., Cardiel1, E. B., Campos, I. J. and Paz1, A. R. (2010). September 8-10. Mechanical energy optimization in trajectory planning for six dof robot manipulators based on eighth-degree polynomial functions and a genetic algorithm. 7th International Conference on Electrical Engineering, Computing Science and Automatic Control. México: 446 – 451.

[16] Gong, D., Lu, L. and Li, M. (2009). May 18-21. Robot path planning in uncertain environments based on particle swarm optimization. IEEE Congress on Evolutionary Computation. Trondheim. DOI:10.1109/CEC.2009.4983204: 2127-2134.

[17] Pedersen, M. E. H. and Chipperfield, A. J. (2010). Simplifying particle swarm optimization. Applied Soft Computing, 10(2): 618-628.

[18] Rao, S. S. (2009). Engineering optimization theory and practice. New Jersey. John Wiley & Sons, Inc.: 4th Edition, pp. 708.

[19] Chavan, S.D. and Adgokar, N.P. (2015). An overview on particle swarm optimization: basic concepts and modified variants. International Journal of Science and Research, 4(5): 255-260.

[20] Alfi, A. (2012). Particle swarm optimization algorithm with dynamic inertia weight for on-line parameter identification applied to Lorenz chaotic system. International Journal of Innovative Computing, Information and Control, 8(2):1191-1203.

[21] Djoewahir, A., Kanya, T. and Shenglin, M. (2012). A modified particle swarm optimization with nonlinear decreasing inertia weight based PID controller for ultrasonic motor. International Journal of Innovation and Technology, 3(3): 198-201.

[22] Tian, D. (2013). A Review of convergence analysis of particle swarm optimization, International Journal of Grid and Distributed Computing, 6(6): 117-128.

[23] Malwiya, R. and Rai, V. (2015). Optimal speed controlling of induction motor using new PSO. International Journal of Advanced Technology & Engineering Research, 5(2): 39-43.

[24] Ibrahim, N.M.A., Atti, H.E.M., Talaat, H.E.A. and Alaboudy, A.H. K. (2015). Modified particle swarm optimization based proportional-derivative power system stabilizer. International Journal of Intelligent Systems and Applications. DOI: 10.5815 / ijisa. 2015. 03.08: 62-76.

[25] Bansal, J.C., Singh, P. K., Saraswat, M., Verma, A., Jadon, S. S. and Abraham, A. (2011). October 19-21. Inertia weight strategies in particle swarm optimization. IEEE 3rd World Congress on Nature and Biologically Inspired Computing. Salamanca. DOI: 10.1109 / NaBIC. 2011. 6089659: 633-640.

[26] Yang, C. H., Tsai, S. W., Chuang, L. Y. and Yang, C. H. (2011). A modified particle swarm optimization for global optimization. International Journal of Advancements in Computing Technology, 3(7): 169-189.

[27] Elsayed, S. M., Sarker, R. A. and Montes, E. M. (2013). June 20-23. Particle swarm optimizer for constrained optimization. IEEE Congress on Evolutionary Computation. Cancun-Mexico. DOI: 10.1109/CEC.2013.6557896: 2703-2711.

[28] Ratanavilisagul, C. and Kruatrachue, B. (2014). A modified particle swarm optimization with mutation and reposition. International Journal of Innovative Computing, Information and Control, 10(6): 2127-2142.

[29] Tian, D. (2015). Particle swarm optimization with chaotic maps and Gaussian mutation for function optimization. International Journal of Grid and Distributed Computing, 8(4): 123-134.

[30] Jamous, R.A., Tharwat, A.A., EL-Seidy, E. and Bayoum, B.I. (2015). Modifications of particle swarm optimization techniques and its application on stock market: A survey. International Journal of Advanced Computer Science and Applications, 6(3): 99-108.

[31] Li, J., Cheng, Y., and Chen, K. (2014). May 31-June 2. Chaotic Particle swarm optimization algorithm based on adaptive inertia weight. 26th Chinese Control and Decision Conference. Changsha. DOI: 10.1109 /CCDC. 2014. 6852369: 1310-1315.

[32] Al-khayyt, S. Z. S., M. A. Abdilatef, Z. M. Yosif (2016). Visual tracking enhancement of object on circular path based on tuned kalman filter by particle swarm optimization. International Journal of Computer Applications. 146(4): 43-50.

[33] Al-khayyt, S. Z. S. (2013). Tuning PID controller by neural network for robot manipulator trajectory tracking. Al-Khwarizmi Engineering Journal, 8(2): 19-28.

خَلْق نقاطِ بينية في الدالة الخطيةِ المندمجة مع مسار القطع المكافئِ بطريقةِ تحقيقِ الأمثلية

سعد زغلول سعيد الخياط
قسم الميكاترونكس / كلية الهندسة / جامعة الموصل
البريد الألكتروني: alkhyaat@yahoo.com

الخلاصة

أن مسار القطعة الخطية المندمجة مع مسار القطع المكافئ ينحرف عن المسار المخطط و هو مقيد بقيمة تعجيل يجب ان تكون كبيرة بما فيه الكفاية. من ناحية أخرى، التعامل مع مسار القطعة الخطية المندمجة مع مسار القطع المكافئ في المقالات الموجودة حاليا حول الموضوع غير قابل للتطبيق مباشرة على المسارات الحركية. في هذا العمل، اقترح تعديل على مسار القطعة الخطية المندمجة مع مسار القطع المكافئ و تم تعشيقه مع أمثلة جسيم السرب لتوليد نقاط عبر المسار. أن الطرق السابقة التي تعتمد على فرضية التوزيع المتساوي لزمن مسار القطع المكافئ حول نقطة المسار، استبدلت بمعاملات مقترحة لحساب الفترة الزمنية لمسار القطعة الخطية. هذه المعاملات هي دوال من السرع بين نقاط المسار. أن مسار القطعة الخطية المندمجة مع مسار القطع المكافئ المطور يستخدم السرع بين نقاط المسار التي تستحصل بطريقة امثلية جسيم السرب لإجبار مدبر الروبوت على المرور خلال نقاط المسار المحددة أخذا بنظر الاعتبار تقيدي السرعة و التعجيل للروبوت المستعمل. كذلك تم اشتقاق علاقات لتصحيح السرعة و معادلات لحساب السرعة الفعلية. نتائج المحاكاة العددية أظهرت ان تعشيق المسار الخطي المندمج مع مسار القطع المكافئ المطور مع أمثلة جسيم الحشد يعمل بشكل جيد في الاختبارات. وان هذه الطريقة المقترحة بسيطة جدا و يمكن ان تستعمل لتخطيط المسار مباشرة و غير ضروري فيها استخدام قيمة تعجيل كبيرة.