

Feasible Joint Angle Continuous Function of Robotics Arm in Obstacles Environment Using Particle Swarm Optimization

Affiani Machmudah and Setyamartana Parman

Abstract. This paper addresses a point-to-point robotic arm path planning in complex obstacle environments. To guarantee a smoothness of a motion during a manipulation, a continuous function of a sixth degree polynomial is utilized as a joint angle path. The feasible sixth degree joint angle path will be searched utilizing a Particle Swarm Optimization (PSO). There is no information regarding the region of this feasible joint angle so that the PSO should search it first. At the first computation where the population is generated randomly, all particles commonly collide with obstacles. The searching computation will be continued till at certain iteration for which the feasible particle is met. Then, the PSO should evolve this particle to find the best one with the highest fitness value. It is very hard computation since it involves a requirement to escape from zero fitness. The most difficult computation in this case is in finding at least one particle that lies in the feasible zone. In this paper, the PSO has shown its good performance in finding the feasible motion of the sixth degree polynomial joint angle path by utilizing just the information of a forward kinematics. To simulate the proposed path planning, 3-Degree of Freedom (DOF) planar robot will be utilized.

1 Introduction

The PSO is one of the natural computation techniques firstly proposed by Kennedy et al in 1995. Although it is new compared with other natural computation methods such as a Genetic Algorithm (GA) and an Artificial Neural Network (ANN), existing researches have indicated that the PSO is very challenging to be utilized to solve very complex optimization problem [2]. The PSO is inspired by the natural animal social behavior such as bird flocking and fish schooling. Different with the GA inspired from the natural selection where the

Affiani Machmudah · Setyamartana Parman
Universiti Teknologi PETRONAS, Mechanical Engineering Departement,
Bandar Seri Iskandar, Tronoh, 31750 Perak, Malaysia

fittest individual is the most survive individual in the population, the individual in the PSO, called particle, has very good communication and mutual aid. The global best solution in the current generation is always informed to all particles in the population. Each particle then tries to update its velocity and position to enhance the current best global solution. This behavior will be continued in order to discover the most optimal solution. The problem lays on the strategy to update the existing velocity so that the enhancement of the current global best solution will be obtained. Kennedy et al [1] proposed the update velocity formulation utilizing Reynold model of bird flocking. The PSO algorithm is relatively simple compared with other evolutionary computations, such as the GA, where it does not need a complicated procedure such as a crossover and a mutation.

Implementing the PSO to solve the complex optimization problem then becomes very active research to be conducted. This paper will investigate the performance of the PSO to solve the arm robot path planning. The optimization problem of the arm robot path planning is very complex regarding the existence of the obstacles.

The recent issue in this research area is finding the best strategy to move the robot from the given initial configuration to the final configuration safely and optimally. One of the path planning strategies is finding the end-effector path first. It needs to solve the inverse kinematics to find the associate feasible joint angle. Boriga et al [6] utilized the polynomials of degrees 9, 7 and 5 as the end-effector path. Then, its inverse kinematics problem was solved and on this basis the runs of displacements, velocities, accelerations and angular jerks of each kinematic chain link were established.

Using the continuous function with intermediate points has also been proposed. In this strategy, it should be noted that many intermediates points will contribute many unknown variables that will induce the computationally expensive problem. Gaspareto et al [7] presented the trajectory planning of robot manipulators with the objective function containing a term proportional to the integral of the squared jerk along the trajectory. Fifth-order B-splines are then used to compose the overall trajectory. Saravanan et al [10] utilized a cubic B-spline for point-to-point robot motion planning under the velocity, the acceleration, and the jerk constraints.

Utilizing the GA in the arm robot path planning has been considered also. Pires et al [9] presented the point-to-point manipulator trajectories using a multi-objective genetic algorithm for two DOF and three DOF manipulators when one circular obstacle was present. They discovered the collision-free joint angle trajectories using the discrete analysis with nine configuration points.

This paper considers the sixth degree polynomial continuous function of the joint angle path to achieve the smoothness and the continuity of the arm robot motion. The joint angle path in the form of the continuous function will give very smooth motion in the manipulation; however, it is not easy to be discovered regarding the presence of the obstacles. The process conveys the complexity where it should be analyzed in two different coordinates. The joint angle should be generated in the joint coordinate while the avoiding collision should be done in the Cartesian coordinate. This problem needs the well-performed searching technique to find the intersection space of the joint angle path and obstacle-free area.

The path planning deals with the searching algorithm of a sequence of robot configurations. They must be collision-free and have connectivity each other. In the presence of the obstacles, the performance analysis of the robot will be difficult to be calculated. The free workspace of the arm robot in the obstacle environment can be very complicated than the one with no obstacle in the environment. The analysis of the connectivity, which is very important to obtain the smooth motion, will also be very complicated due to obstacle barrier. The solution of point-to-point path planning commonly is proposed by finding free configuration point by point from initial configuration to final configuration. In this case, besides the free configuration analysis, the requirement of the connectivity between these discrete solutions should be strongly considered to avoid discontinuity that will make the motion becomes unreachable.

The sixth degree polynomial is utilized in this paper to model the joint angle path. The joint angle changes continuously as sixth degree polynomial function from the initial angle to the final angle. The feasible joint angle trajectories are regarded to the collision-free configuration and the connectivity during the manipulation from the initial configuration to the final configuration. To find this feasible range, this paper utilized the random search technique, namely the PSO. Firstly, the computation begins by searching the particles randomly where they will give fitness equal to zero when there is the collision. After the non-zero fitness has been discovered, the evolution process is started to find the best particle with the highest fitness value.

2 Problem Statements

Fig. 1 gives the illustration of the path planning. The grayed space concerns with free configuration while the white space corresponds to obstructed configuration.

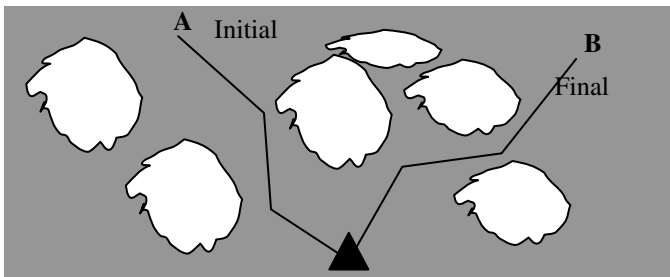


Fig. 1 Path planning

The point-to-point path planning is the problem to search the feasible joint angle trajectories in such a way that the end-effector move from point A to point B. The motion should lay in the free configuration and avoid obstructed configuration. In this paper, the sixth degree polynomial will be utilized as joint angle path while the optimization goal is to minimize the joint angle traveling distance.

3 Link Model

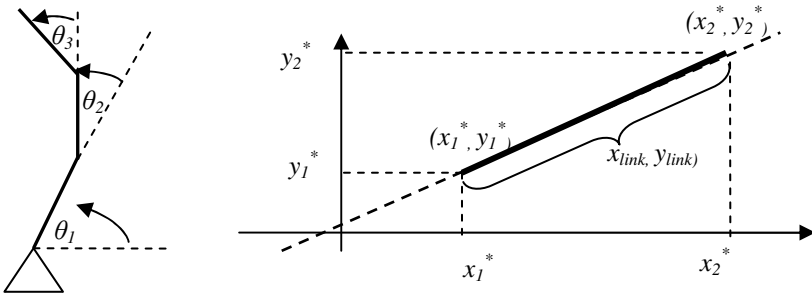


Fig. 2 Modeling of the link

The relation of the joint position and the end-effector position can be determined by calculating the following forward kinematics

$$x_1^* = l_1 \cos \theta_1, \quad y_1^* = l_1 \sin \theta_1 \tag{1}$$

$$x_2^* = x_1^* + l_2 \cos(\theta_1 + \theta_2), \quad y_2^* = y_1^* + l_2 \sin(\theta_1 + \theta_2) \tag{2}$$

$$x_3^* = x_2^* + l_3 \cos(\theta_1 + \theta_2 + \theta_3), \quad y_3^* = y_2^* + l_3 \sin(\theta_1 + \theta_2 + \theta_3) \tag{3}$$

where θ_i , (x_1^*, y_1^*) , (x_2^*, y_2^*) , and (x_3^*, y_3^*) , are the joint angle of i^{th} link, the first joint position, the second joint position, and end-effector position, respectively.

Fig. 2 gives the illustration of the link modeled in the Cartesian coordinate. The model of the link for each position can be obtained using intersection of the straight-line equation with its x area as follows

$$\frac{y_{link(i)} - y^{*(i-1)}}{x_{link(i)} - x^{*(i-1)}} = \frac{y_i^* - y^{*(i-1)}}{x_i^* - x^{*(i-1)}} \quad \text{and} \quad x^{*(i-1)} \leq x_{link(i)} \leq x_i^* \tag{4}$$

where x_{link} and y_{link} describe link position in x-axis and y-axis, respectively, while (x_i^*, y_i^*) is a joint position.

The links position will change in line with the changing of the joint angle. Thus, the collision detection procedure will involve great numbers of a collision checking activity since the collision-free should be guaranteed achievable for all configurations. Fig. 3. gives the illustration of the collision detection in the computation process.

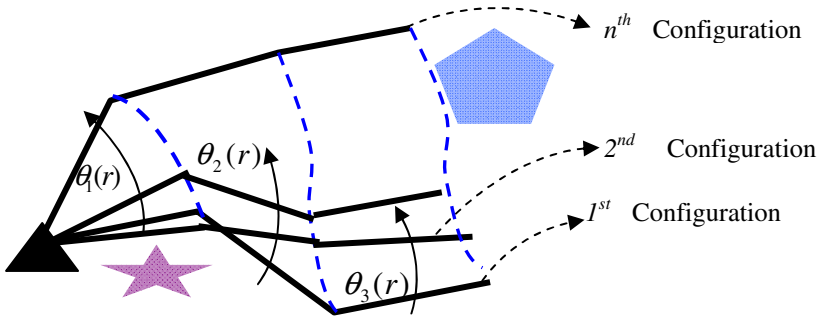


Fig. 3 Collision checking procedure for each link configuration

4 Joint Angle Traveling Distance

The joint angle travelling distance is very important parameter in the arm robot path planning. Fig. 4 gives the representation of this joint angle traveling distance.

Similar to the translation motion, the joint angle traveling distance represents the distance of the rotational motion of the links. It is the length of the joint angle curve which can be formulated as

$$f = \int_0^1 \sqrt{1 + \left(\frac{d\theta(r)}{dr}\right)^2} dr \tag{5}$$

where f is the joint angle traveling distance, $\theta(r)$ is the joint angle path, and r is the linear time-scale, respectively.

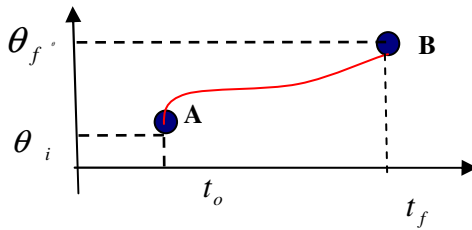


Fig. 4 Joint angle traveling distance

The optimization objective in this paper is to minimize the joint angle traveling distance, so that Eq. 5 needs to be optimized into the minimal one.

5 Avoiding Collision Arm Robot Path Planning

To analyze the arm robot motion planning can be done in two different coordinates. The first coordinate is the Cartesian coordinate or often called an operational coordinate. The second coordinate which can be utilized is the joint coordinate. The joint coordinate is the coordinate where the x-axis and y-axis represent the time and joint angle terms, respectively.

For sixth degree polynomial joint angle path, the joint trajectories are generated in the joint space while the avoiding collision is done in the Cartesian space. Thus, the analysis will be done in the both spaces, the operational space and the joint space, simultaneously. In this avoiding collision case, the feasible joint angle path is not easy to be searched according to a difficulty to find an intersection between obstacle-free area and links configurations. Fig. 5 describes this process.

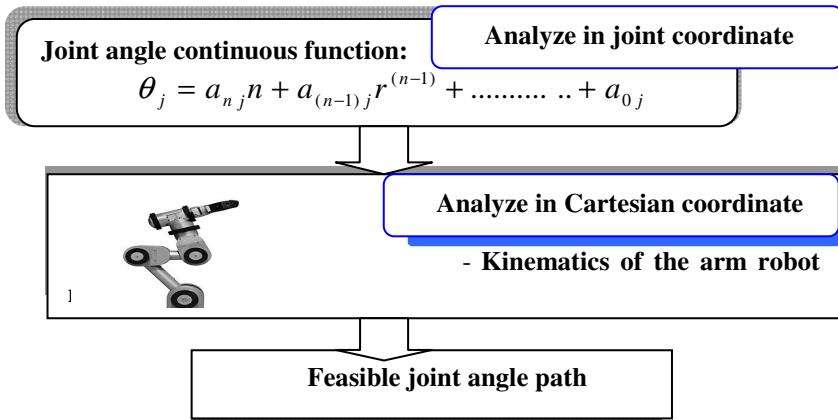


Fig. 5 Arm robot path planning process

Between these two coordinates is related with the arm robot kinematics. It can be in the form of either inverse kinematics or forward kinematics. When the analysis is done in the operational coordinate commonly by generating end-effector path, the collision-free path can be obtained easily; however it takes computationally expensive due to the inverse kinematics. The end-effector path should be translated into the joint angle coordinate since the control of the link is done by motor rotation. Utilizing the PSO, this paper will govern the forward kinematics for checking collision to generate the collision-free joint angle path.

6 Performance of Robotics Arm in the Presence of Obstacles

The feasibility analysis of the motion of the robotics arm has been investigated since many years ago. For point-to-point motion, it is very important to guarantee that from initial configuration to final configuration can be done safely without any discontinuity. Some tools to analyze the performance of the robotics arm has been introduced, such as the concepts of the workspace, the aspect, and the connectivity. Detail of them can be found in [27]. This section will give an introduction of the important concepts of the workspace, the aspect, and the connectivity.

In the presence of obstacles, the free workspace are very essential to analyze the robot performance during the manipulation. Without proximity of the obstacles, the workspace of 3-DOF planar arm robot will be equal to $l_1 + l_2 + l_3$, with l is the length of the link.

In the presence of the obstacle, the wokspace of the robot can be very complicated. Workspace is utilized to analyze the performance of accessibility of the arm robot manipulator; however, in the presence of the obstacles, the concept of the acessibility will not be sufficient [27]. Other two important tools are namely the aspect and the connectivity.

Refer to Fig. 6a, the trajectories from point A and point B are not feasible with that path configuration. By workspace analysis, points A and B are acessible, or lay in the free workspace, so that between these points there are feasible trajectories but not in that path configuration. Due to obstacle barrier, the path shown in Fig 6a, is unreachable. Aspect analysis needs to be done in point C and point D. Three link robot is a redundant robot where it has many possible solutions for one end-effector point. The aspect analysis deals with the posture of the link robot between trajectories that should be reachable. There is correct configuration of point C and point D so that between these points can be reached. Fig. 6b shows one of the feasible path from point A to point B without including any discontinuities.

The connectivity should be strongly considered in the presence of obstacles besides the free workspace concept. The configurations can be consisted of some discrete trajectories. Among these discrete trajectories, it should be guaranteed that there are n-connectivity. The detail of the connectivity tool can be found in [27]. This paper utilizes sixth degree polynomial that gives collision-free configuration as the joint angle profile. By keeping the joint angle traveling distance into the minimum one, the feasible continuous joint angle path will have the smooth motion.

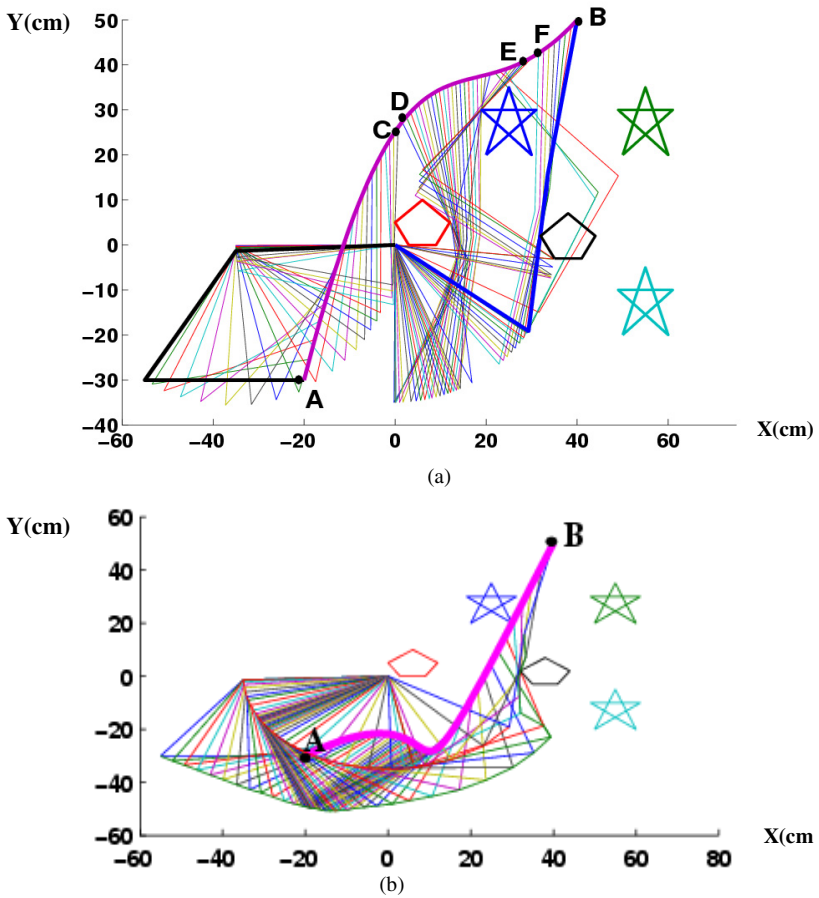


Fig. 6 (a) unfeasible motion (b) feasible motion

7 Sixth Degree Polynomial Joint Angle Path

The joint angle as function of time can be expressed as composition function of joint angle profile and linear time-scale as follows

$$\theta(t) = \theta(r) \circ r(t) \tag{6}$$

where $\theta(t)$, $\theta(r)$, $r(t)$ are the joint angle function of time, the joint angle profile, and linear time-scale, respectively.

With 6th degree polynomial function as joint angle path, the path profile can be expressed in the following

$$\theta_{nk}(r) = a_{6k}r^6 + a_{5k}r^5 + a_{4k}r^4 + a_{3k}r^3 + a_{2k}r^2 + a_{1k}r + a_{0k} \tag{7}$$

where r is linear time-scale, t/T , with t is time and T is total transfer time. a_{nk} , θ_{nk} , and k are the n^{th} polynomial coefficient of k^{th} link, the n^{th} joint angle of k^{th} link, and the number of the link, respectively.

The path planning problem then can be reduced into the problem to find the feasible joint angle path from parameter 0 to 1.

Utilizing the chain rule, the velocity and acceleration can be derived as follows

$$\dot{\theta}(t) = \dot{\theta}(r) \frac{1}{T} \tag{8}$$

$$\ddot{\theta}(t) = \ddot{\theta}(r) \frac{1}{T^2} \tag{9}$$

In this path planning, boundary conditions utilized are known initial joint angle, known final joint angle, zero value of the initial velocity, the final velocity, the initial acceleration, and final acceleration.

By taking all boundary conditions into Eqs. (7, 8, 9), the following equations will be obtained

$$a_{0k} = \theta_{ik} \quad ; \quad a_{1k} = a_{2k} = 0; \quad a_{5k} = -3a_{6k} - 6\theta_{ik} + 6\theta_{fk} \tag{10}$$

$$a_{4k} = 0.5(-9a_{6k} - 5a_{5k}) \quad ; \quad a_{3k} = \theta_{fk} - \theta_{ik} - a_{6k} - a_{5k} - a_{4k}$$

where θ_{ik} and θ_{fk} are the initial joint angle of k^{th} link and the final joint angle of k^{th} link, respectively .

Therefore, the joint angle equation in Eq. (7) will be reduced in the following expression

$$\theta_{nk} = a_{6k}r^6 + a_{5k}r^5 + a_{4k}r^4 + a_{3k}r^3 + a_{0k} \tag{11}$$

Then, the unknown variables of the path planning with sixth degree polynomial are as follows

a_{61}	a_{62}	a_{63}
----------	----------	----------

where a_{nk} is the n^{th} polynomial coefficient of k^{th} link. Each link will contribute one unknown variable.

7.1 Intersection of 6th Coefficient with Obstacle-Free Area

From the previous section, it can be seen that analysis of the feasibility of the arm robot motion is extremely important. Some concepts to investigate the arm robot performance have been proposed by researchers. They are generally utilized as the tools to analyze the arm robot performance. However, in the presence of the obstacles, the analytical techniques to solve them generally will be very difficult and tedious. Existing methods to calculate them have many limitations [27]. In the contrary, the numerical analysis using the random search techniques will be relatively simple and powerful.

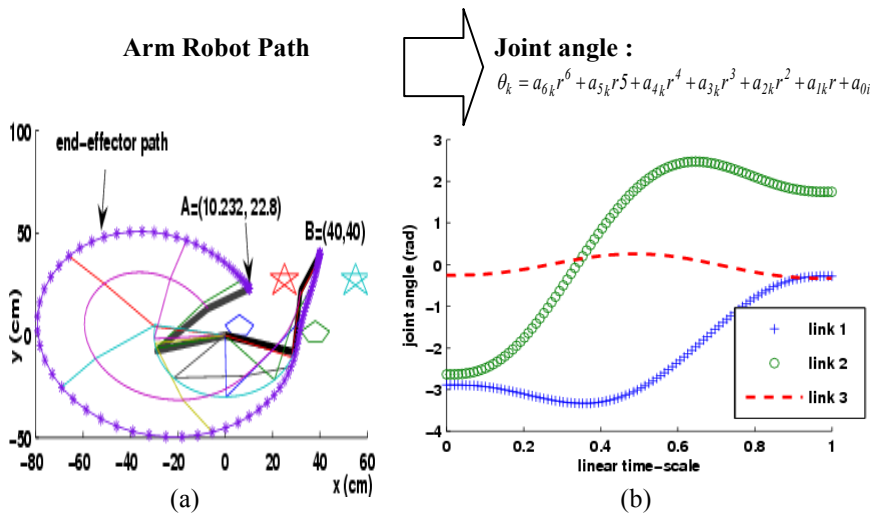


Fig. 7 (a) link robot path (b) joint angle

The continuous function of the joint angle trajectories will start from the initial joint angle to the final joint angle. The trajectories continuously move between these initial and final joint angle values. Fig.7b gives the illustration of the continuous joint angle trajectories of path configuration of Fig. 7a. In this paper, the joint angle trajectories are in the form of the sixth degree polynomial function.

The problem becomes how to find the intersection area of the sixth degree polynomial joint angle trajectories with the free configuration. Return back to Fig. 5, the connection between the Cartesian coordinate and joint angle coordinate was in the form of the unknown variable, the 6th coefficient of polynomial function, a_{6k} . The procedure to find the collision-free configuration is then done by checking the collision of link configurations using the forward kinematics.

8 NLP of Arm Robot Path Planning

In the optimization technique, the Non Linear Programming (NLP) formulation has been utilized as a test function to investigate the performance of the proposed method.

The general formulation of the NLP is as follows

$$\begin{aligned}
 \text{Optimize} & : f(\bar{x}), \quad \bar{x} = (x_1, x_2, \dots, x_n) & (12) \\
 \text{Subjected to:} & \quad g_j(\bar{x}) \leq 0 \\
 & \quad h_j(\bar{x}) \leq 0
 \end{aligned}$$

where \bar{x} , $f(\bar{x})$, $g_j(\bar{x})$, and $h_j(\bar{x})$ are the optimization variables, the objective function, and the constraint functions, respectively.

The NLP formulation for the arm robot path planning can be derived using kinematics of the robot as well as the obstacle analysis, which have been introduced in the previous section. It is formulated in Table 1. It considers the avoiding collision rules as a constraint so that this NLP will be suitable for any kind of the obstacle environment. The obstacle can be modelled in separate equation depends on the environment of the robot.

Table 1 NLP of arm robot path planning

Min	$\int_0^l \sqrt{1 + \left(\frac{d\theta(r)}{dr}\right)^2} dr$
Subject to :	<ul style="list-style-type: none"> • Collision detection rules : if \exists link position (i) \cap obstacle area, then the collision happen, otherwise is free from collision
Where :	$\theta_j = a_{n_j}n + a_{(n-1)_j}r^{(b-1)} + \dots + a_{0_j}$ <p style="margin-left: 40px;"> $r = \frac{t}{T}$, $0 \leq r \leq 1$, Link position (i) = $(x_{link(i)}, y_{link(i)})$ </p> $\frac{y_{link(i)} - y_{(i-1)}^*}{x_{link(i)} - x_{(i-1)}^*} = \frac{y_i^* - y_{(i-1)}^*}{x_i^* - x_{(i-1)}^*} \text{ and } x_{(i-1)}^* \leq x_{link(i)} \leq x_i^*$ <p style="margin-left: 40px;"> (x_i^*, y_i^*) is joint positions of each link </p>

9 Pseudo Code Computation of Proposed Path Planning

Mathematically, the previous NLP illustrates the complexity of the path planning of the robotic arm in the obstacle environment. It needs the computational strategy to solve it. The pseudo code of the avoiding collision path planning utilized in this paper is shown in Fig 8.

- ✓ For initial configuration to final configuration, generate : a_{61}, a_{62}, a_{63}
- ✓ Calculate the joint angle equations for each link :

$$\theta_1 = a_{61}r^6 + a_{51}r^5 + a_{41}r^4 + a_{31}r^3 + a_{01}$$

$$\theta_2 = a_{62}r^6 + a_{52}r^5 + a_{42}r^4 + a_{32}r^3 + a_{02}$$

$$\theta_3 = a_{63}r^6 + a_{53}r^5 + a_{43}r^4 + a_{33}r^3 + a_{03}$$

where : $a_{0k} = \theta_{ik}$

$$a_{5j} = -3a_{6j} - 6\theta_{ij} + 6\theta_{jj}; a_{4j} = 0.5(-9a_{6j} - 5a_{5j})$$

$$a_{3j} = \theta_{jj} - \theta_{ij} - a_{6j} - a_{5j} - a_{4j}$$

→ Check collision :

- ✓ For $0 \leq r \leq 1$ divide r into n point
- ✓ For all point :
 By forward kinematics check whether any link intersect with obstacle → Eq. (4)

Collide :

Collision-free

- ✓ Find the feasible area of σ^{th} coefficient, a_{6k} , for each link
- ✓ Find the optimal σ^{th} coefficient, a_{6k} , for each link

Fig. 8 Pseudo-code

10 Particle Swarm Optimization

To execute the pseudo code of the path planning presented in the previous section, the PSO is utilized as searching method. The PSO is inspired by the social behavior of animals, such as bird flocking or fish schooling. It was originally proposed by Kenedy et al [1] in 1995. They utilize the Reynold model of bird flocking as natural computing to evolve the existing solution generation to generation until the best particle is discovered. Starting with the random population, optimization variables, which are called particles, have certain position and velocity. According to the local and global best solutions in the population, each particle flies with new velocity to obtain the enhancement of the global best solution . The PSO adds this velocity to the particle position. If the best local solution has the fitness value less than the fitness of the current global solution, then the best local solution replaces the best global solution.

In the first time, Kennedy et al [1] proposed the update velocity formulation as follows

$$v_{t+1} = v_t + \varphi_1 \beta_1 (p_i - x_i) + \varphi_2 \beta_2 (p_g - x_i) \quad (13)$$

$$x_{t+1} = x_i + v_{t+1} \quad (14)$$

where v_t , v_{t+1} , φ_1 , and φ_2 are the velocity, the update velocity, influence of individual knowledge, influence of group knowledge, respectively. β_1 and β_2 are uniformly distributed random numbers, p_i and p_g are the individual's previous best position and the group's previous best position, while x_i is the current position in the dimension considered.

In 1998 Shi et al [3] introduced an inertia weight factor, ω . Eq. (13) can be expressed in the following

$$v_{t+1} = \omega v_t + \varphi_1 \beta_1 (p_i - x_i) + \varphi_2 \beta_2 (p_g - x_i) \quad (15)$$

Kennedy et al [3] proposed to improve the velocity by utilizing constriction factor, χ , as follows

$$v_{t+1} = \chi \{ \omega v_t + \varphi_1 \beta_1 (p_i - x_i) + \varphi_2 \beta_2 (p_g - x_i) \} \quad (16)$$

$$\chi = \frac{2}{2 - \varphi - \sqrt{\varphi^2 - 4\varphi}} \quad , \quad \varphi = \varphi_1 + \varphi_2 \quad , \quad \varphi > 4$$

The PSO starts to develop into very challenging computational method. Works to improve the performance of the velocity formulation as well as to apply the PSO to solve more complex optimization problem become very active research [2]. Fig. 9 illustrates the procedure of the PSO.

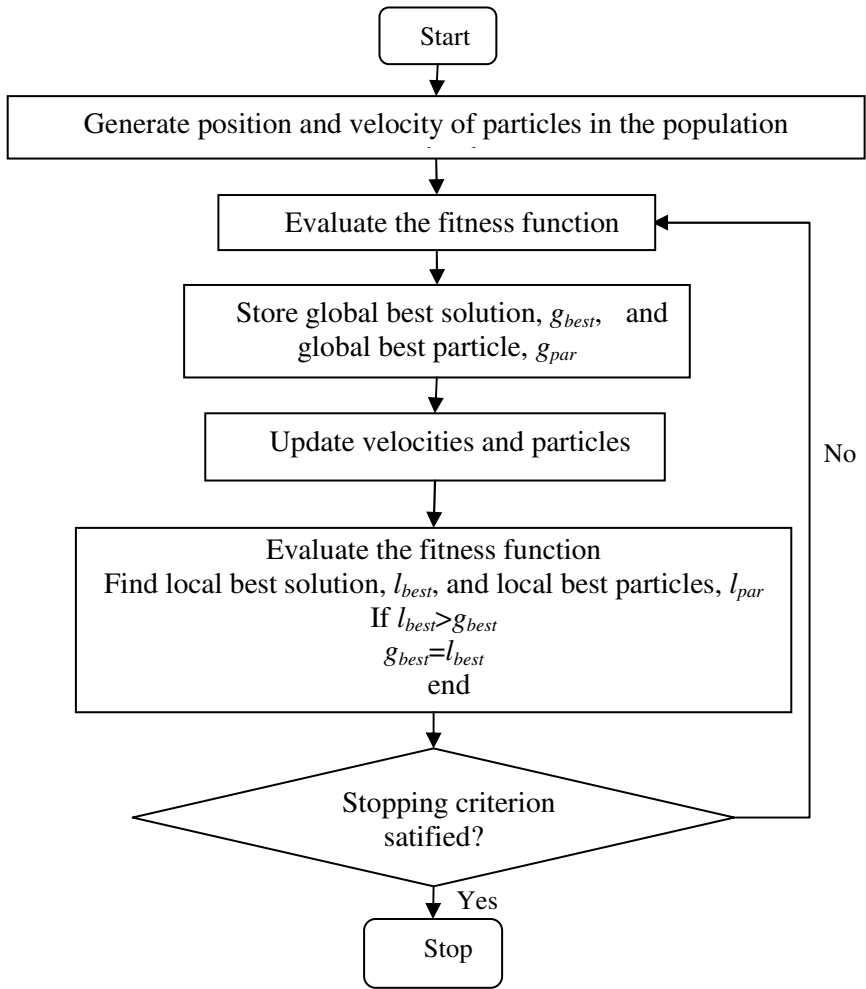


Fig. 9 PSO procedure

10.1 Fitness Function

The objective optimization in this path planning is to minimize the joint angle traveling distance. There is one strict constraint in this case, the avoiding collision. To guarantee the collision-free, the collision detection should be executed in the computation. Due to there is no information of the position of the a_{6k} feasible range, the death penalty has been chosen in the PSO. The death penalty will turn the PSO into totally randomized searching method. It will automatically kill every particle that collides with the obstacles by giving the zero fitness. The PSO needs to escape from zero fitness condition to discover the range of feasible a_{6k} .

The death penalty can be expressed as rules in the following

1. If any constraint does not satisfy by the joint angle path, then the path is failure
2. Otherwise, the path is success

To follow these rules, the membership fitness function has been composed as follows

$$F_1 \begin{cases} = 0 & \exists (x_{link(i)}, y_{link(i)}) \cap \text{obstacle area} \\ = \frac{1}{f_1 + f_2 + f_3} & \text{Otherwise} \end{cases} \tag{17}$$

where f_i is the i^{th} joint angle traveling distance.

For the joint angle as function of time-scale, r , the joint angle traveling distance is the integration formula in Eq. (5). This paper utilizes the Simpson’s rule to compute the integration.

10.2 Computation of Avoiding Collision Sixth Degree Polynomial Path Planning by PSO

How to search the area of feasible a_{6k} was the fundamental problem in this path planning. Instead of using analytical technique, this paper proposes to use the numerical method governing the random search technique where in this paper; the PSO is chosen as shown in Fig. 10.

The PSO is the natural computing method based on the social behavior of the animals. The information of the best solution is very important to update the velocities of the particles as shown in Eqs. (13, 14).

At the first time of computation, there is no information about the range of this feasible a_{6k} . Due to lacking of this range information, the death penalty is chosen where the fitness value is set to be zero when there is collision between the link

and the obstacle. The problem then becomes how to escape from this zero fitness condition. The aim at this stage is to search the zone of the feasible sixth polynomial coefficients. At the first iteration till the certain iteration, the system will contain fully zero fitness where all particles collide with obstacle. As soon as the non-zero fitness has been discovered, the problem becomes to evolve this variable, till the best particle is met.

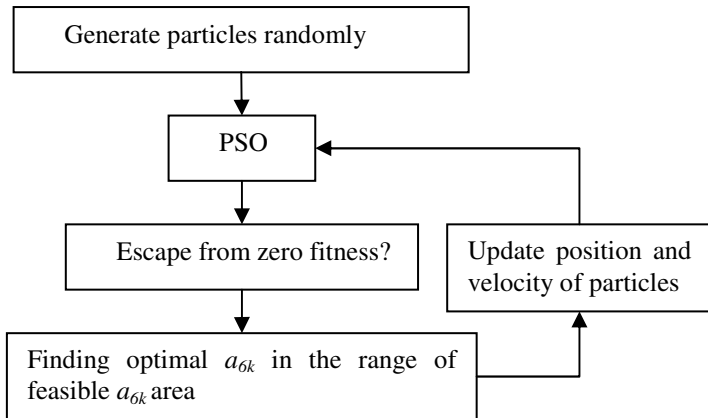


Fig. 10 Feasible sixth degree polynomial joint angle path computation by PSO

11 Simulation Results and Discussion

A simulation in MATLAB had been done, by coding in m file. The simulations use 20 individuals in the population. The constraint values of the arm robot manipulator are listed in Table 1. The lengths of the links are 30 cm, 30 cm, and 20 cm for the first link, the second link, and the third link, respectively. The masses of the first link, the second link, and the third link links are 3 kg, 3 kg, and 2 kg, respectively.

11.1 Case 1

This paper will search the range of this feasible a_{6k} area utilizing the PSO. Fig. 11 illustrates the case 1, with obstacle environment. The initial point is (50, -10) while the final point is (50, 40). At the first time of the computation, there is no information regarding this range. The first computation is done by generating the particles randomly where the particles commonly collide with the obstacles. This

paper continues the searching process and investigates whether the PSO can escape from this collision condition and find the feasible a_{6k} or not.

For case 1, seventh running have been done to investigate the escaping and evolving process till 100 generations. The results of this activity are presented in Table 2. The PSO utilized is the inertia weight method for different values of c_1 and c_2 , and also the constriction method.

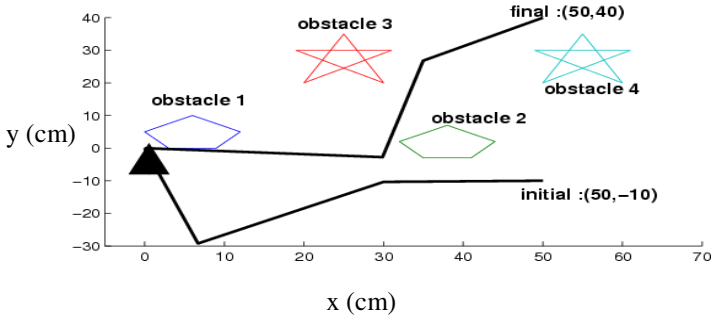


Fig. 11 Case 1 and obstacle environment

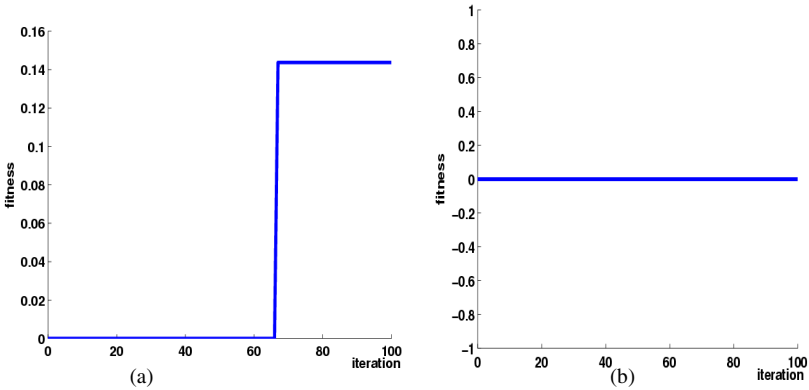


Fig. 12 Escaping process during 100 iterations (a) success (b) fail

Fig. 12a illustrates the success of the escaping process while Fig. 12b shows the failure example of the escaping process during 100 iterations.

Table 2 presents the detail information of the escaping process and the evolving result at 100 iterations. The PSO utilizes the weight method for different value of c_1 and c_2 and the constriction method. It shows that it is very difficult to escape from zero fitness. During seventh running times, there are four failures. This paper utilizes just 100 iterations since it prefers to investigate the PSO performance in escaping zero fitness at short computational time. Although, the failure probably happens during 100 generations; however, the very interesting result is the PSO

can find the feasible sixth degree polynomial function of joint angle. From Table 2, it can be seen that the first time of the non-zero fitness met is unpredictable. The most important thing is the PSO succeeded in finding this feasible joint angle path so that the evolution to get the optimal particles can be done easily now.

Table 2 Escaping and evolving process with PSO for seventh running time for 100 iterations

PSO	Run 1 (gen, iter)	Run 2 (gen, iter)	Run 3 (gen, iter)	Run 4 (gen, iter)	Run5 (gen, iter)	Run 6 (gen,iter)	Run 7 (gen,iter)
A	<i>Escaping</i> (48, 0.1717) a ₆₁ = 59.418 a ₆₂ =-63.667 a ₆₃ =0	fail	<i>Escaping</i> (24,0.1695) a ₆₁ =57.396 a ₆₂ =-65.55 a ₆₃ =10.886	fail	fail	fail	<i>Escaping</i> (19, 0.1393) a ₆₁ =82.7691 a ₆₂ =-71.535 a ₆₃ =-34.144
	<i>Evolving</i> (100, 0.1724) a ₆₁ = 58.91 a ₆₂ =-63.237 a ₆₃ =-0.4425		<i>Evolving</i> (100,0.1774) a ₆₁ = 52.461 a ₆₂ =-60.1392 a ₆₃ =10.8932				<i>Evolving</i> (100, 0.1732) a ₆₁ = 55.1786 a ₆₂ =-64.2508 a ₆₃ =7.1811
B	<i>Escaping</i> (29, 0.1323) a ₆₁ =86.4098 a ₆₂ =-71.8554 a ₆₃ =-44.7995	fail	<i>Escaping</i> (87, 0.1589) a ₆₁ =67.8054 a ₆₂ =-69.8544 a ₆₃ =-12.1606	<i>Escaping</i> (8, 0.1357) a ₆₁ =85.0301 a ₆₂ =-73.5545 a ₆₃ =-37.2313	fail	fail	fail
	<i>Evolving</i> (100, 0.1389) a ₆₁ =78.5906 a ₆₂ =-67.7068 a ₆₃ =-43.614		<i>Evolving</i> (100, 0.1592) a ₆₁ =66.985 a ₆₂ =-70.2461 a ₆₃ =-12.1546	<i>Evolving</i> (100,0.1458) a ₆₁ =80.0597 a ₆₂ =-73.9362 a ₆₃ =-19.8461			
C	fail	<i>Escaping</i> (32, 0.1397) a ₆₁ =80.0420 a ₆₂ =-66.9973 a ₆₃ =-41.2126	fail	<i>Escaping</i> (39, 0.1255) a ₆₁ =86.7794 a ₆₂ =-65.8419 a ₆₃ =-65.4384	<i>Escaping</i> (94, 0.1565) a ₆₁ =70.377 a ₆₂ =-73.4002 a ₆₃ =-5.3923	fail	fail
		<i>Evolving</i> (100, 0.1419) a ₆₁ =77.3644 a ₆₂ =-65.8755 a ₆₃ =-40.9938		<i>Evolving</i> (100, 0.1589) a ₆₁ =67.6419 a ₆₂ =-62.9829 a ₆₃ =-23.1282	<i>Evolving</i> (100,0.158) a ₆₁ =70.644 a ₆₂ =69.649 a ₆₃ =9.8101		

- A: inertia weight method : $c_1=c_2=1.5$, $\omega=(\text{maxit-iter})/\text{maxit}$
 - B: inertia weight method : $c_1=c_2=0.5$, $\omega=(\text{maxit-iter})/\text{maxit}$
 - C: constriction method : $c_f=3$, $c_2=2$
- Cost = fitness value, iter = iteration

11.2 Other Simulation Cases

Previous section investigated that the PSO succeeded to find the range of feasible a_{6k} of case 1. Table 3 presents other path planning cases for different initial and final configurations utilizing the inertia weight with $c_1 = c_2 = 1.5$.

Table 3 Other simulation cases results

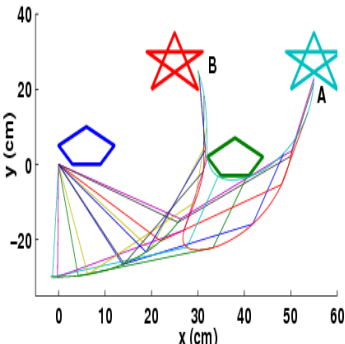
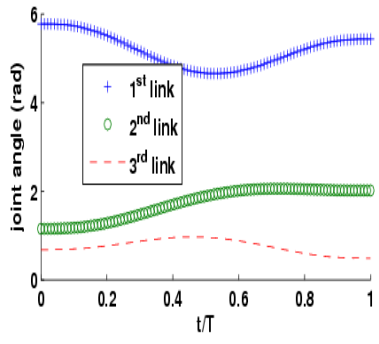
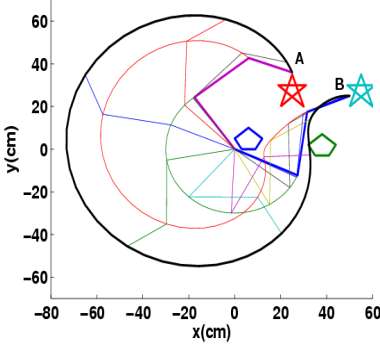
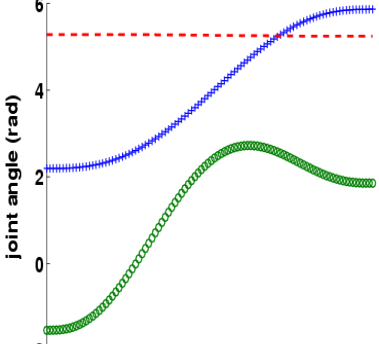
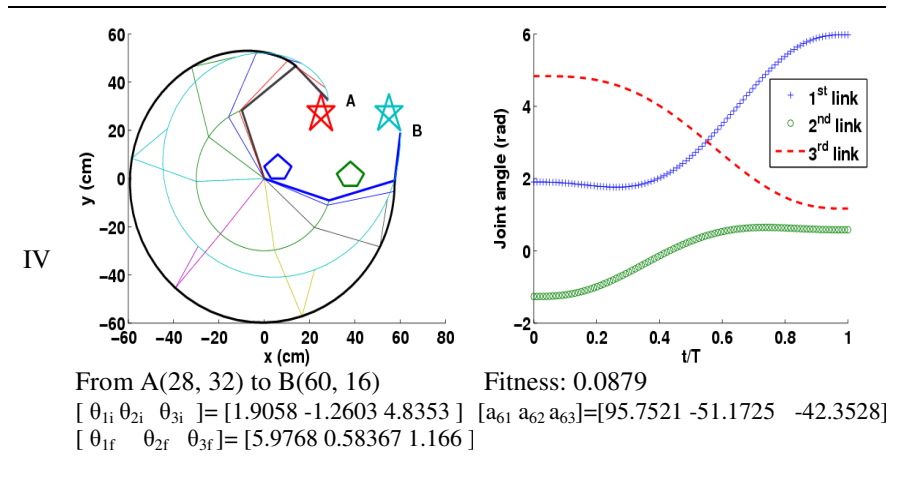
case	configurations	Optimal(100 generations)
II	 <p>From A(55, 23) to B(30, 25) $[\theta_{1i} \ \theta_{2i} \ \theta_{3i}] = [5.7784 \ 1.1523 \ 0.68017]$ $[a_{61} \ a_{62} \ a_{63}] = [66.7388 \ -28.1028 \ -9.8855]$ $[\theta_{1f} \ \theta_{2f} \ \theta_{3f}] = [5.4343 \ 2.0154 \ 0.48554]$</p>	 <p>Fitness: 0.1946 $[a_{61} \ a_{62} \ a_{63}] = [66.7388 \ -28.1028 \ -9.8855]$</p>
III	 <p>From A(25 36) to B(50 25) $[\theta_{1i} \ \theta_{2i} \ \theta_{3i}] = [2.1948 \ -1.5348 \ 5.2794]$ $[\theta_{1f} \ \theta_{2f} \ \theta_{3f}] = [5.8605 \ 1.8551 \ 5.2402]$</p>	 <p>Fitness: 0.0973 $[a_{61} \ a_{62} \ a_{63}] = [3.4136 \ -139.7617 \ -0.2667]$</p>

Table 3 (continued)



11.3 Range of the Feasible Sixth Polynomial Coefficient, a_{6k}

This section will investigate the behavior of the feasible a_{6k} coefficient. Case II will be examined.

It has been known that it is very difficult to find the feasible point-to-point configuration in the obstacle environment. It needs to guarantee the connectivity among trajectories for all configurations. The path planning result of sixth degree polynomial joint angle path was very interesting where the PSO can find the feasible a_{6k} . It will be very challenging if there is the pattern of the range of this feasible a_{6k} . This section investigates whether the feasible a_{6k} can be expressed simply as $a_{6k(min)} \leq a_{6k} \leq a_{6k(max)}$ or not.

Table 4 consists of the examples of the feasible $[a_{61} \ a_{62} \ a_{63}]$. Their values are obtained from the PSO.

Table 4 Feasible compositions of a_{6k}

composition	$[a_{61} \ a_{62} \ a_{63}]$	Fitness
(1)	[113.0533 -42.3741 -10.0300]	0.1476
(2)	[147.7472 -70.80701 48.89361]	0.10656
(3)	[124.1641 -53.39219 9.386029]	0.13575
(4)	[119.2847 -80.32816 41.86955]	0.11624
(5)	[181.0683 -34.8128 -147.858]	0.080496

Based on the composition (1) and the composition (2) in Table 4, we can see that there are possible ranges for a_{6k} : $113.0553 \leq a_{61} \leq 147.7472$, $-70.80701 \leq a_{62} \leq -42.3741$, $-10.0300 \leq a_{63} \leq 48.89361$. Do these ranges become the feasible a_{6k} ? To investigate this issue, the test cases based on the predicted zones must be constructed. Some test cases on $[a_{61} a_{62} a_{63}]$ have been done as presented in Table 5.

Table 5 Test cases to investigate the range of feasible a_{6k}

Prediction $(a_{6k})_{min} \leq a_{6k} \leq (a_{6k})_{max}$ from	Test case $[a_{61} a_{62} a_{63}]$	Fitness	Remarks
(1) & (2) Prediction : $113.0553 \leq a_{61} \leq 147.7472$ $-70.80701 \leq a_{62} \leq -42.3741$ $10.0300 \leq a_{63} \leq 48.89361$	[120 -50 -8]	0.14	Cannot predict the range of feasible a_{6k} from (1) & (2)
	[120 -50 20]	0	
	[120 -40 20]	0	
	[120 -65 0]	0.1335	
	[140 -60 40]	0	
(1) & (3) $113.0553 \leq a_{61} \leq 124.1641$ $-53.39219 \leq a_{62} \leq -42.3741$ $9.386029 \leq a_{63} \leq 48.89361$	[120 -50 9.386029]	0.1397	✓ The range seems can be predicted, but it will be unique for one composition. ✓ For composition [120 -50 a_{63}], the maximum a_{63} is 16 ✓ For composition [124.1641 -53.39219 a_{63}], the maximum a_{63} is 21 ✓ For composition [113.0533 -42.3741 a_{63}], the maximum a_{63} is 4 ✓ Generally, the feasible a_{6k} seems to have random pattern.
	[120 -50 12]	0.1391	
	[120 -50 16]	0.1380	
	[120 -50 17]	0	
	[124.1641 -53.39 17]	0.1338	
	[124.1641 -53.39 20]	0.1329	
	[124.1641 -53.39 21]	0.1325	
	[124.1641 -53.39 22]	0	
	[113.05 -42.3741 4]	0.1486	
	[113.05 -42.3741 5]	0	

The results of the fitness value for each composition of $[a_{61} \ a_{62} \ a_{63}]$ shows that we cannot predict the feasible range of a_{6k} from the known feasible compositions of a_{6k} . Next test cases have been done with the searching ranges between the composition (1) and the composition (3). It can be seen in Table 5 that when the searching areas have been reduced, the range of feasible a_{6k} will be resulted. Each a_{6k} composition will have the certain unique or specific range. The feasible zone seemed to have the random pattern and cannot be expressed in the form of $a_{6k(min)} \leq a_{6k} \leq a_{6k(max)}$. This result shows that it is very difficult to predict the range of feasible a_{6k} for one case point-to-point path planning using the known feasible composition of a_{6k} only. Next section will investigate the PSO behavior in evolution process when the feasible a_{6k} is given.

11.4 Evolution Process

From previous section, it can be seen that the range of feasible a_{6k} were random. Instead of investigating the range of feasible a_{6k} , this paper will investigate the performance of the PSO to evolve the known feasible a_{6k} resulted from the escaping process. The simple numerical experiment is done by using the feasible a_{6k} listed in Table 4 as part of the initial population in the PSO.

Table 6 Evolution by PSO when the known feasible a_{6k} is given

One of initial particle in population	Evolution after 100 generations
[181.0683 -34.8128 -147.858] fitness = 0.080496	[62.9 -25.115 -15.554] fitness = 0.19765
[119.2847 -80.32816 41.86955] fitness = 0.11624	[63.934 -26.45 -13.261] fitness = 0.19707
[66.7388 -28.1028 -9.8855] fitness = 0.1946	[61.51 -22.95 -18.847] fitness = 0.19819

Table 6 presented the results of the evolution by PSO. From Table 6, it can be seen that although the range of feasible a_{6k} is difficult to be predicted as presented in the previous section; however, by the PSO procedure, each case can evolve into same area of a_{6k} composition after 100 generations. The final a_{6k} for each initial particle lies in the same zone. For this experiment, the value of a_{6k} will be around 60, -25, and -16, for a_{61} , a_{62} , and a_{63} , respectively.

These results are very interesting because the PSO shows as the powerful random searching technique in finding the composition of $[a_{61} \ a_{62} \ a_{63}]$ that is very difficult to be calculated by the conventional method. For future, the deep investigation of the pattern of this range will be very challenging to be conducted. It is also a challenge to investigate the PSO to solve other complex optimization

problem involving uncompleted information by simply executing the dead penalty method. It will need a good searching ability, thus the strategy to turn the PSO into fully randomized searching method is also very expected for escaping from zero fitness.

Finding the feasible configuration of the point-to-point path planning of robotics arm in the obstacle environment has been becoming the issue in robotics field since many years ago. This paper has also showed that the range of feasible $a_{\delta k}$ is difficult to be predicted from the known feasible compositions since it had the random pattern. However, by evolution process of the PSO, this feasible composition can be evolved into the best composition easily. The PSO can find the feasible sixth degree polynomial of joint angle path of the robotics arm simply by executing the algorithm following the natural behavior of animal. The unknown variables are the particles with certain position and velocity. These particles will always move by updating the velocity and the position until the optimal position and velocity are discovered. At the beginning of the computation, the PSO searched the feasible $a_{\delta k}$ utilizing the death penalty. This is due to no information of the feasible $a_{\delta k}$ yet. Thus, at the first iteration, where the particles are generated randomly, all particles in the population collide with the obstacles. The PSO continues the searching process until the feasible zone is met. Results show that PSO succeeded in escaping and searching this feasible zone. When the information of the feasible $a_{\delta k}$ has been known already, the evolution will be started by evaluating the fitness function and updating the velocity and position of the particles.

12 Conclusions

The arm robot path planning utilizing 6th degree polynomial function in the complex geometrical environments has been presented. The PSO has succeeded in solving this complex optimization problem where it can escape from zero fitness and then evolve the non-zero fitness particle into the best one. Utilizing the PSO as random search computational method, it is possible to find feasible continuous joint angle path in the presence of obstacle. This work seems to be very complicated using the conventional method; however, the PSO can find it easily. It will give significant breakthrough potentially in the motion planning of the robotics arm.

References

- [1] Kennedy, J., Eberhart, R.: Particle Swarm Optimization. In: Proceedings of IEEE International Conference on Neural Networks, Piscataway, NJ, pp. 1942–1948 (1998)
- [2] Banks, A., Vincent, J., Anyakoha, C.: A review of particle swarm optimization. Part I: background and development. *Natural Computing Journal* 6, 467–468 (2007)
- [3] Shi, Y., Eberhart, R.C.: A modified particle swarm optimizer. In: Proceedings of the IEEE International Conference on Evolutionary Computation, pp. 69–73

- [4] Chettibi, T., Lehtihet, H.E., Haddad, M., Hanchi, S.: Minimum Cost Trajectory Planning for Industrial Robots. *European Journal of Mechanics* 23, 703–715 (2004)
- [5] Chettibi, T.: Synthesis of Dynamic Motions for Robotic Manipulators with Geometric Path Constraints. *Mechatronics* 16, 547–563 (2006)
- [6] Boriga, M., Grabos, A.: Planning of Manipulator motion Trajectory with higher-degree Polynomials use. *Mechanism and Machine Theory* 44, 1400–1419 (2009)
- [7] Gasparetto, A., Zanotto, V.: A new method for smooth trajectory planning of robot manipulators. *Mechanism and Machine Theory* 42, 455–471 (2007)
- [8] Gasparetto, A., Zanotto, V.: Optimal trajectory planning for industrial robots. *Advances in Engineering Software* 41, 548–556 (2010)
- [9] Pires, E.J.S., Oliveira, P.B.M., Machado, J.A.T.: Manipulator Trajectory Planning using a MOEA. *Journal of Applied Soft Computing* 7, 659–677 (2007)
- [10] Saravanan, R., Ramabalan, S., Balmurugen, C.: Evolutionary multi-criteria trajectory modeling of industrial robots in the presence of obstacles. *Engineering Applications of Artificial Intelligence Journal* 22, 329–342 (2009)
- [11] Pires, E.J.S., Machado, J.A.T., de Moura Oliveira, P.B.: Robot Trajectory Planning using Multi-Objective Genetic Algorithm Optimization. In: Deb, K., et al. (eds.) *GECCO 2004. LNCS, vol. 3102*, pp. 615–626. Springer, Heidelberg (2004)
- [12] Garg, D.P., Kumar, M.: Optimization Techniques Applied to Multiple Manipulators for Path Planning and Torque Minimization. *J. Eng. App. of Artificial Intell.* 15, 241–251 (2002)
- [13] Yang, X., Wang, H., Zhang, C., Chen, K.: A method for mapping the boundaries of collision-free reachable workspaces. *Mechanism and Machine Theory Journal* 45, 1024–1033 (2010)
- [14] Conkur, E.S.: Path planning using potential fields for highly redundant manipulators. *Robotics and Autonomous Systems* 52, 209–228 (2005)
- [15] Cheng, H., Cheng, H.D.: Feasible map algorithm for path planning. *Robotics and Autonomous Systems* 17, 257–268 (1996)
- [16] Bazaz, S.A., Tondu, B.: Minimum Time On-line Joint Trajectory Generator Based on Low Order Spline Method for Industrial Manipulators. *Robotics and Autonomous Systems Journal* 29, 209–228 (1999)
- [17] Chong, J.W.S., Ong, S.K., Nee, A.Y.C., Youmi, K.Y.: Robot Programming using Augmented Reality: An Interactive Method for Planning Collision-free Paths. *Robotics and Computer-Integrated Manufacturing Journal* 25, 689–701 (2000)
- [18] Rodriguez, A.G.G.: Collision-free Motion Planning and Scheduling. *Robotics and Computer-Integrated Manufacturing Journal* (2011) (article in press, corrected proof)
- [19] Yahya, S., Moghavvemi, M., Mohamed, H.A.F.: Geometrical approach of planar hyper-redundant manipulators: Inverse kinematics, path planning and workspace. *Simulation Modelling Practice and Theory* 19, 406–422 (2011)
- [20] Pereira, G.S., Kumar, V., Campos, M.F.M.: Closed Loop Motion Planning of Cooperating Mobile Robots using Graph Connectivity. *Robotics and Autonomous Systems* 56, 373–384 (2008)
- [21] Clark, C.M.: Probabilistic Road Map Sampling Strategies for Multi-robot Motion Planning. *Robotics and Autonomous Systems* 53, 244–264 (2005)
- [22] Marcos, M.G., Machado, J.A.T., Perdicou'lis, T.P.A.: A Fractional Approach for the Motion Planning of Redundant and Hyper-Redundant Manipulators. *Signal Processing Journal* 91, 974–984 (2011)

- [23] Khoukhi, A., Baron, L., Balazinski, M., Demirli, K.: A Hierarchical Neuro-Fuzzy System to Near Optimal-time Trajectory Planning of Redundant Manipulators. *Engineering Applications of Artificial Intelligence* 21, 562–570 (2008)
- [24] Hammour, Z.S.A., Mirza, N.M., Mirza, S.M., Arif, M.: Cartesian Path Generation of Robot Manipulators using Continuous Genetic Algorithms. *Robotics and Autonomous Systems* 41, 179–223 (2002)
- [25] Pozna, C., Troester, F., Precup, R.E., Tar, J.K., Preitl, S.: On the design of an obstacle avoiding trajectory: Method and simulation. *Mathematics and Computers in Simulation* 79, 2211–2226 (2009)
- [26] Angeles, J.: *Fundamental of Robotic: Mechanical Systems: Theory, Methods, and Algorithms*. Springer, New York (2002)
- [27] Dombre, E., Khalil, W.: *Robot Manipulators: Modelling, Performance, Analysis, and Control*. ISTE Ltd, London (2002)