Australian Journal of Basic and Applied Sciences, 7(10): 426-445, 2013
ISSN 1991-8178

Optimization of Task Scheduling for Single-Robot Manipulator using Pendulum-Like with Attraction-Repulsion Mechanism Algorithm and Genetic Algorithm

Issa Ahmed Abed, S.P. Koh, Khairul Salleh Mohamed Sahari, S.K. Tiong and Nadia M.L. Tan

College of Engineering, Universiti Tenaga Nasional, Selangor, Malaysia.

Abstract: In order to solve the problem of task scheduling for robot manipulators in terms of minimal error and less cycle time, the combination between Pendulum-like with Attraction-repulsion mechanism algorithm (PA) and Genetic algorithm (GA), known as PA-GA has been proposed. The proposed PA algorithm has been used to generate the multiple robot configurations at each task point with high precision. The PA-GA algorithm is used to solve the problem of task scheduling for robot manipulator, in terms of minimum error and less cycle time. In the task scheduling problem with static obstacles, a single-robot manipulator moves from point to point and guarantees zero collision with the static obstacles at the task points. The movement and the configurations of the manipulators at the task points were illustrated using a simulator developed via Visual Basic. net. The method is verified via simulation for the four-link planar robots in a working environment with different sizes of static obstacles cluttered at different locations.

Key words: Electromagnetism-like, planar robot, collision avoidance, optimal-time

INTRODUCTION

A robotic system is considered redundant when it possesses more degrees of freedom than those required to execute a given task (Marcos *et al*., 2012). Redundant robot manipulators significantly contributed to manufacturing automation. It can perform extra capabilities compare to non-redundant robot. Furthermore, these manipulators should be capable of performing complex tasks in the minimum amount of time for the purpose of increasing productivity (Zacharia and Aspragathos, 2005). Thus, one of the objectives of the robotics research is to develop suitable algorithms in order to enable manipulators to quickly perform designated or desired tasks. The followings are the recent implementation for solving the IK and the task scheduling problem.

Jasim (2011) implemented four Adaptive Neuro-Fuzzy Inference System (ANFIS) systems for solving the IK problem for 4-DOF SCARA robot. The Neuro-fuzzy system is constructed, where the positions of end-effecter are as inputs to the system and the outputs are the joint angles of the manipulator. The membership functions which are Gaussian membership are tuned using a learning algorithm. Alavandar and Nigam (2008) developed Neuro-Fuzzy based approach for the inverse kinematics solution of industrial robot manipulators. In their paper, the ANFIS is learned from the coordinates and the angles which are used as training data. ANFIS is implemented by a representative fuzzy inference system using a Back-propagation (BP) neural network-like structure. The proposed approach is tested using computer simulations with two examples of 2-DOF and 3-DOF manipulators which are proving the effectiveness of the method. Zhang *et al*. (2012) has been proposed an Adaptive Particle Swarm Optimization method (A-PSO) combined the kinematics equations. The proposed method is suggested to solve the inverse kinematics for serial manipulator. They modified the conventional PSO because it is easily trapped in local optima and need much population size. The fitness function that is used is defined as the combinatorial matrix deviation from the end of the robot to the object. The proposed method is able to reduce the complexity of the analysis of inverse kinematics equations. They are compared their method with the traditional PSO algorithm and the efficiency of the proposed method is demonstrated in the case study.

The Traveling Salesman Problem (TSP) is the task of finding the shortest possible tour through a given set of cities (Narasimhan, 2006). Given a graph G on n_v vertices, a Hamilton cycle in G is a cycle that passes through all the n_v vertices exactly once (Yampolskiy *et al*., 2012). An optimal Hamilton cycle is a cycle of least total cost. The TSP can be translated to the problem of finding an optimal Hamilton cycle. In order to solve the TSP many researchers in the literature have been suggested approaches such as Genetic algorithm (GA) to solve the problem (Khan *et al*., 2009).

Task scheduling refers to the computation of the optimum sequence among different points (Xidias *et al*., 2010). Therefore, the problem of optimal task scheduling is akin to the traveling salesman problem. For example, in the problem of optimal task scheduling for robots, the gauge is time instead of distance, and the problem is more intricate compared to TSP. In optimal-time task scheduling of robot manipulators, certain issues regarding the robots, such as the collision avoidance are considered, and also multiple robot configurations at each task point (city), instead of a single solution.

Corresponding Author: Issa Ahmed Abed, College of Engineering, Universiti Tenaga Nasional, Selangor, Malaysia.
E-mail: issaahmedabd80@yahoo.com, Tel: +60172884287.

In order to reduce the cycle time for the manipulator during the path, many optimization methods are presented in the literature. Addressing this concern, Edan *et al*. (1991) constructed an algorithm based on the Nearest Neighbor (NN) to obtain the near-optimal time path between the fruit locations for the fruit-harvesting robots. Thus, the authors did not take into account the possibility of collision when designing the algorithm; and also not mentioned about the possible configurations that were used to determine the optimal sequence. Petiot *et al*. (1998) succeeded in showing the potential of the elastic-net method to minimize the cycle time of robots. The algorithm schedules the trajectory points in such a way that gives minimum time, although it is incumbent upon the fact that the energy function (E) needs to be minimized. This is achievable via the modification of a gradient method. The proposed algorithm is effective for a two or three degrees of freedom because of the increased computer time cost. However, the possibility of failure is not nonexistent, as a much cluttered environment might lead to failure. GA is one of optimization methods that retains the ability to search in the midst of large complex spaces (Azariadis and Aspragathos, 2005). Based on a genetic algorithm, Zacharia and Aspragathos (2004, 2005) proposed a method that optimizes the sequence of points that are visited by the end effector. The algorithm takes into account the multiple solutions of the IK problem. However, the technique utilizes a non-redundant manipulator in a collision free workspace. Xidias *et al*. (2010) was introduced optimization algorithm based on GA, with the intention of determining the near-optimal sequence of task points without the possibility of collision between the robot's tip and static obstacles. The algorithm is applicable for both categories of redundant and non-redundant robot, and it also integrates the multiple existing solutions at each task point.

In this paper, a method incorporating both Pendulum-like with Attraction-repulsion mechanism algorithm (PA) and GA is proposed to obtain the near optimal time sequence for a robot manipulator. The robot visits the task points in the environment which contains static obstacles. Thus, the problem can be divided into two sub problems: task scheduling and collision free path, where this problem depends on the solutions that are generated via the developed PA algorithm. The paper is organized as follows: Section 2 is the problem statement and the objective function; Section 3 and Section 4 explains the implementations of PA and PA-GA, respectively; while the simulation results are presented in Section 5; and finally, the conclusion is given in Section 6.

Problem Statement:

The geometric method can be used to find the forward equations for the n-DOF planar robot (Yahya *et al*., 2011; Abed *et al*., 2012) as follows:

$$X_{cur} = L_1 * \mathrm{Cos}\,\theta_1 + L_2 * \mathrm{Cos}\,(\theta_1 + \theta_2) + ... + L_n * \mathrm{Cos}\,(\theta_1 + \theta_2 + ... + \theta_n) \quad (1)$$

$$Y_{cur} = L_1 * \mathrm{Sin}\,\theta_1 + L_2 * \mathrm{Sin}\,(\theta_1 + \theta_2) + ... + L_n * \mathrm{Sin}\,(\theta_1 + \theta_2 + ... + \theta_n) \quad (2)$$

where L_n denotes the n-th link length, θ_n is the n-th joint angle, and (X_{cur}, Y_{cur}) is the current solution at any point of the task.

The positional error between the current solution and the goal position (X_{tp}, Y_{tp}) of the end effector according to Yao and Gupta (2007) is given as follows:

$$f_{error}(\theta) = \sqrt{(X_{tp} - X_{cur})^2 + (Y_{tp} - Y_{cur})^2} \quad (3)$$

In task scheduling, the trajectory between point pt_{i-1} and point pt_i in the task is represented by a cubic function for each joint whose value at t_0 is the initial position of the joint and whose value at t_f being its goal position of that joint (Craig, 2005; Ata and Myo, 2005; Ata, 2007; Guo *et al*., 2009) as follows:

$$\theta\,(t) = C_0 + C_1 t + C_2 t^2 + C_3 t^3 \quad (4)$$

where C_0, C_1, C_2, C_3 are constants and the constraints would be as follows:

The initial and final values for the angles are:

$$\theta\ (0) = \theta_0 \tag{5}$$

$$\theta\ (t_f) = \theta_f \tag{6}$$

The initial and final velocity values are zero

$$\dot{\theta}\ (0) = 0 \tag{7}$$

$$\dot{\theta}\ (t_f) = 0 \tag{8}$$

Eqs. (5-8) are considered as constraint equations

Derive Eq. (4) to calculate the velocity

$$\dot{\theta}\ (t) = C_1 + 2C_2 t\ + 3C_3 t^2 \tag{9}$$

Combining Eqs. (4-9) yield

$$\theta_0 = C_0 \tag{10}$$

$$\theta_f = C_0 + C_1 t_f + C_2 t_f^2 + C_3 t_f^3 \tag{11}$$

$$0 = C_1 \tag{12}$$

$$0 = C_1 + 2C_2 t_f + 3C_3 t_f^2 \tag{13}$$

Solve Eqs. (10-13)

$$C_0 = \theta_0 \tag{14}$$

$$C_1 = 0 \tag{15}$$

$$C_2 = \frac{3}{t_f^2}(\theta_f - \theta_0) \tag{16}$$

$$C_3 = -\frac{2}{t_f^3}(\theta_f - \theta_0) \tag{17}$$

After calculating the constants, the cubic polynomial equation (Eq. (4)) can be used to calculate the trajectory between point pt_{i-1} and point pt_i.

It is required that the robot visits the task points once and double back to its point of origin, utilizing less cycle time for this leg of the tour, while accounting for the multiplicity of robot configurations at the task points.

The cycle time during the task from point pt_{i-1} with solution θ^p to the point pt_i with solution θ^q is as follows (Xidias *et al*., 2010):

$$T_{travel} = \sum_{i=2}^{m_\gamma} \max_j \left(\frac{\left|\theta_{ji}^q - \theta_{j(i-1)}^p\right|}{\dot{\theta}_j}\right) \tag{18}$$

where T_{travel} is the time it takes to visit all task points, $i = 1,2,3,....,m_{\gamma}$, where m_{γ} is the number of task points, $j = 1,2,3,....,n$, n is the number of DOF, $p,q = 1,2,3,.....,N_s$, N_s is the number of solutions, θ_{ji} is the joint displacement at the joint j in the task point pt_i, and $\dot{\theta}_j$ is the average velocity of joint j which is assumed to be constant. This is taken to be quite a reasonable approximation, due to the fact that the time belongs to acceleration, and the deceleration is rather negligible. Eq. (18) explains the fact that the time it takes to travel between two successive points is evaluated by a manipulator's joint that is the slowest. Subsequently,

$$T_{return} = \max_{j} \left(\frac{\left| \theta_{j1}^{q} - \theta_{jm_{\gamma}}^{p} \right|}{\dot{\theta}_j} \right) \tag{19}$$

where T_{return} is the return time to the initial task point.

Also, the total cycle time for the manipulator to visit all of the task points and return to the home position is

$$T_{task} = T_{travel} + T_{return} \tag{20}$$

In this work, to avoid collision, a circle method (Lai and Kang, 2009; Nearchou and Aspragathos, 1997; Rana and Zalzala, 1997) is introduced to check the collision. It has the advantage of low computational cost and is sometimes fast in the collision detection computations (Beaumont and Crowder, 1989). It can be also applied in real time collision avoidance (Beaumont and Crowder, 1991). In this method, the links of the manipulator are approximated via tangent circles, while the obstacles are enclosing via one circle, as shown in Fig.1. This makes the collision incumbent upon the sum of the radii of the present circles.

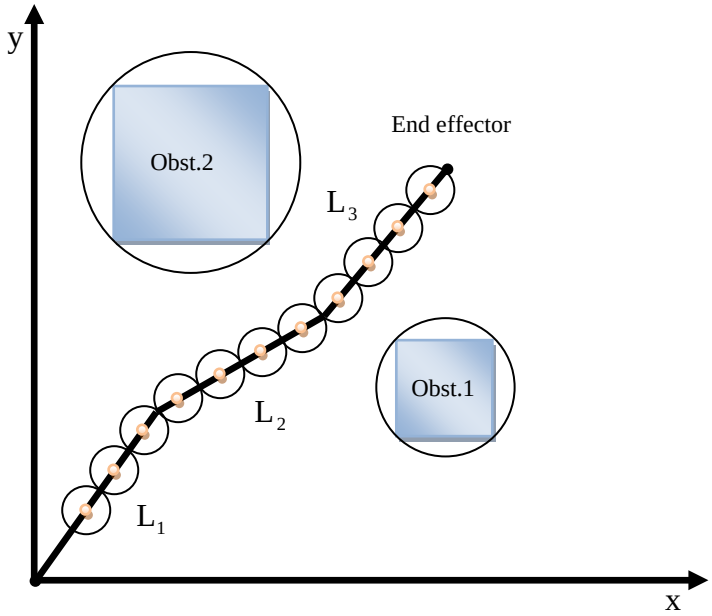

Fig. 1: Circle approximation for robot and obstacles.

The radius of the circles would be calculated as follows (Beaumont and Crowder, 1989):

$$r = \frac{L_n}{2N_c} \tag{21}$$

where r is the radius for each circle and N_c is the number of circles for each link, displayed in Fig. 2.

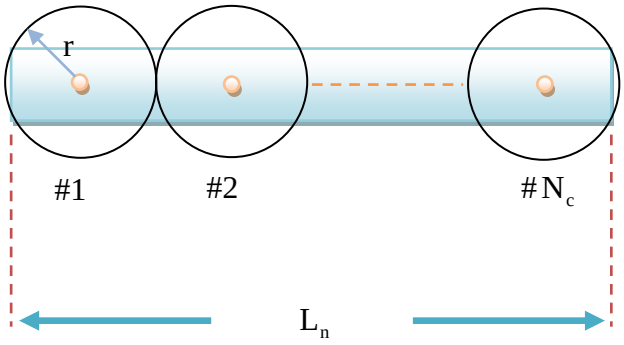

Fig. 2: Representation of the robot link by circles.

The centers of the circles for each link are evaluated according to the midpoint formula. This formula locates the point exactly between two other points, or a line that bisects (divides into equal halves) a given line segment (manipulator's link). Representation of the links by a number of circles and the obstacle by a circle would moderate between the computational cost and the accuracy. Then the collision avoidance objective function is:

$$f_{obst} = \begin{cases} 1 & \text{no collision} \\ c & \text{otherwise} \end{cases} \tag{22}$$

where c is assumed as a large number.

Implementation of PA Algorithm for Multiple Robot Solutions:

Electromagnetism-like algorithm is a global optimization algorithm that adapts the attraction-repulsion mechanism to enable it to move sample points in the direction of optimality (Su and Lin, 2011; Filipović *et al*., 2013). The general scheme for electromagnetism-like algorithm which consists of four main parts is shown in Fig. 3 (Jolai, 2012).

1. Initialization
2. iteration $\leftarrow 1$
3. **While** iteration< Max_iteration **do**
4. Evaluation of objective function
5. Local search (Lsiter, δ)
6. F $\leftarrow$ Calc F()
7. Move (F)
8. iteration $\leftarrow$ iteration +1
9. **End while**

Fig. 3: EM algorithm general scheme.

I. Initialization of the Population:

A population with m points is randomly generated, with n coordinates. Each coordinates are uniformly distributed within both the upper u_k and lower l_k bounds (Chang *et al*., 2009). An objective function value for each sample $f(\theta^i)$ is evaluated after the generation of the samples in the population. Then the point with best objective function is stored in θ^{best} (Miao and Jiang, 2012).

II. Local Search:

The procedure searches for a better solution by collecting the localized information from every single sample point (Birbil and Fang, 2003; Gilak and Rashidi, 2009).

III. Charge and Resultant Force Calculations:

The theory of superposition of electromagnetism posits that a force that is exerted on a point by other points is inversely proportional to the distance between the points and directly proportional to the product of their charges (Birbil and Fang, 2003; Tsou and Kao, 2006).

The first step involves the charge determination of each sample point, which is conducted for each generation based on the objective function of this particular point and the objective function for the best point; detailed in the Eq. (23) (Birbil and Fang, 2003; Lee and Chang, 2010):

$$q^i = \exp\left\{ -n \frac{f(\theta^i) - f(\theta^{best})}{\sum_{k=1}^{m} [f(\theta^k) - f(\theta^{best})]} \right\}, \forall i \tag{23}$$

where $f(\theta^{best})$ is the objective function of the current best solution. It is also worth noting that the charge of a sample point is without sign. An alternative to this is the fact that the direction of a particular force generated between two points would be specified after comparing the objective function value for each existing point (Birbil and Fang, 2003), thus θ^{best} would be the point that attracts all the other points in the population. Hence,

$$F_j^i = \begin{cases} (\theta^j - \theta^i) \dfrac{q^i q^j}{\left\| \theta^j - \theta^i \right\|^2}, & \text{if} \quad f(\theta^j) < f(\theta^i) \\ (\theta^i - \theta^j) \dfrac{q^i q^j}{\left\| \theta^j - \theta^i \right\|^2}, & \text{if} \quad f(\theta^j) \geq f(\theta^i) \end{cases} \quad , \quad i = 1,2,.....m \tag{24}$$

Then the total force is,

$$F^i = \sum_{j \neq i}^{m} F_j^i \quad , \quad i = 1,2,....., m \tag{25}$$

where F^i is the total force exerted on sample point θ^i. A point that has a superior objective function means that it has higher charge and would attract other points, while a point with an inferior objective function value act to repel the others (Birbil *et al.*, 2004).

IV. Movement along the Total Force:

After the completion of the force evaluation, the movement according to force is determined. The particle would update itself according to the direction of the force via random step length (Eq. (26)) (Rocha and Fernandes, 2008). In Eq. (26), the RNG is designated as a vector, whose components denote the allowed feasible movement toward the upper bound, u_k or the lower bound, l_k. In addition, exerted force on each particle would be normalized in order to maintain feasibility. Therefore,

$$\theta^i = \theta^i + \lambda_1 \frac{F^i}{\left\| F^i \right\|} (\text{RNG}), \; i = 1,2,....., m \tag{26}$$

The sample point moves to the direction of upper bound via the random step length when the force is positive, and vice versa (Jhang and Lee, 2009).

In order to increase the capability of the searching algorithm with fewer shortages, pendulum-like algorithm has been proposed in this work. The suggested pendulum-like algorithm is integrated with attraction-repulsion mechanism in order to develop powerful algorithm to solve the complex problems. However, the calculations of

the force and movement of the attraction-repulsion mechanism are similar to the ones in the original EM algorithm. Fig. 4 displays the procedures of the PA algorithm.

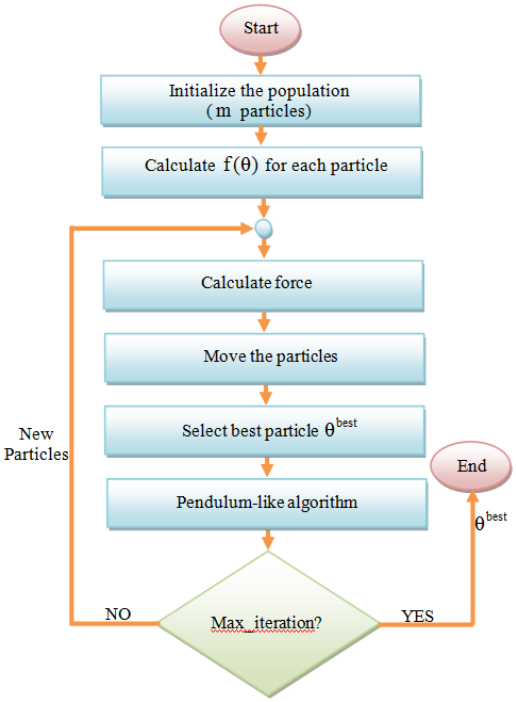

Fig. 4: Flowchart for PA algorithm.

Pendulum-like is a new method which is adopted the principle of the pendulum. The algorithm utilizes the idea of swinging a point mass (m_p) around the zero position as shown in Fig. 5. In the searching process, the current solution moves towards the optimal point. Therefore, by the component of the force which depends on the displacement of the current best solution will always ensure the solution to be directed to the optimal value.

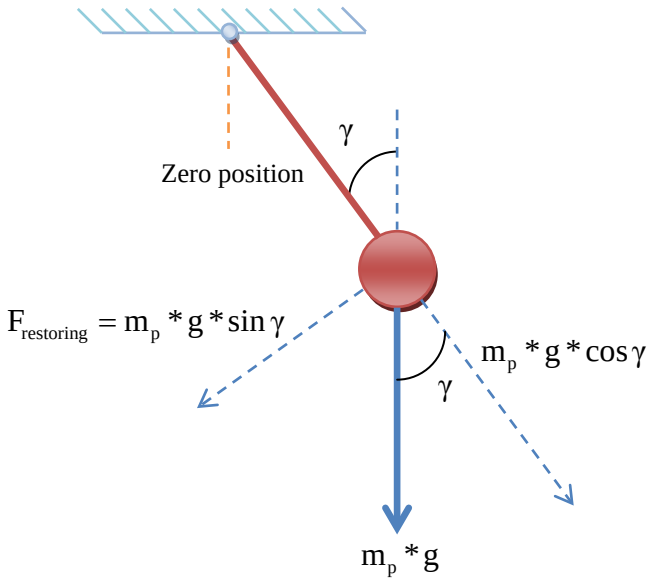

Fig. 5: Restoring force principle.

In this section, some points related to the pendulum-like algorithm would be discussed. Thus, instead of using the term "restoring force" as in Fig. 5, which means restoring the point mass to the equilibrium point; it would be more suitable if it is called the directed force which means directing the current best solution to the global point. So,

$$F_{directed} = m_p * g * \gamma \quad (27)$$

where g is the acceleration of gravity and γ is the angle between the string and the vertical line as shown in Fig. 5.From Eq. (27), the directed force is proportional to the displacement. Hence, the decrease in the value of the displacement would decrease the force component until the current best solution reaches the desired point where the force would be totally absent.

Fig. 6 shows the change in the position of the pendulum during the time. In this figure it is considered the point 4 is the equilibrium point, point 1 as maximum rightward displacement, and point 7 as maximum leftward displacement. When the pendulum at point 1, the displacement would reduce gradually toward the equilibrium point. This because of the force which is directed the movement to the equilibrium point. The same thing occurs in the leftward at point 7.

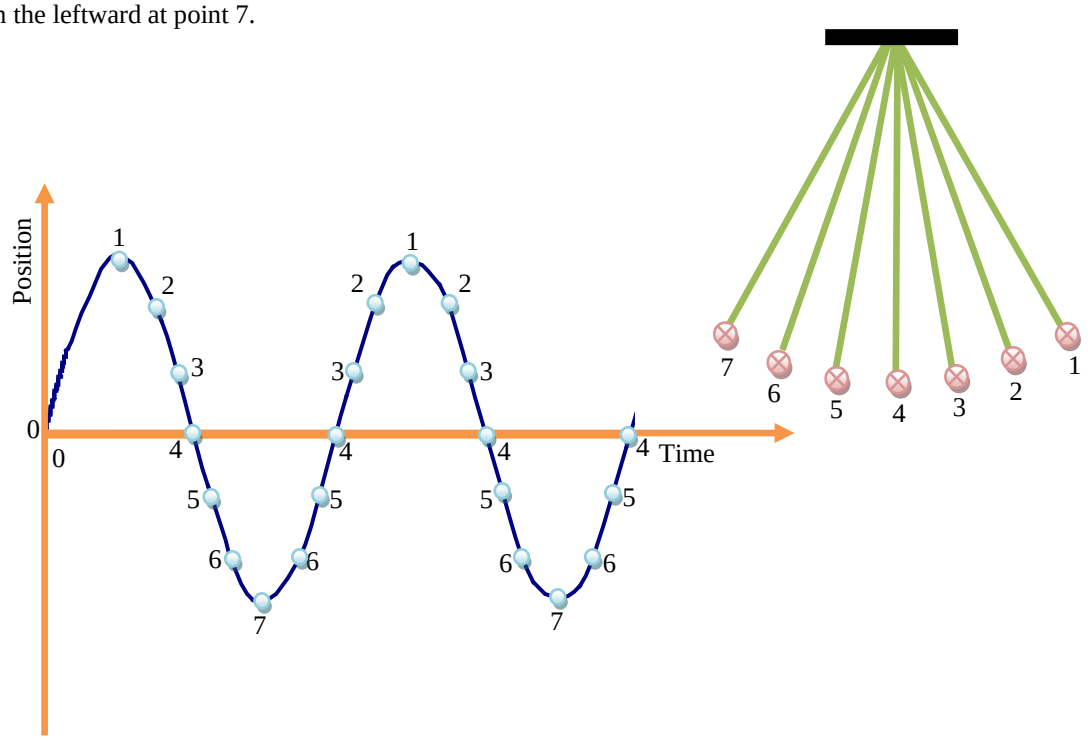

Fig. 6: Oscillating the pendulum through the points.

The displacement equation for single pendulum is

$$\gamma = \gamma_{max} \cos(\omega_p t + \Phi) \quad (28)$$

where γ_{max} is the maximum displacement, ω_p angular frequency, and Φ is constant. As a result, the idea behind the Eq. (28) and Fig. 6 motivated the authors to propose the following adaptive displacement equation:

$$\gamma = 1 / (N_i)^b \quad (29)$$

where $b > 0$ is a user defined constant. With an increase in the number of iterations, the value of $F_{directed}$ would decrease regularly to ensure that the best point is always guided to the global location.

It can also notice that, the idea of frequency of oscillation which is used in the pendulum-like algorithm came from the back and forth oscillation as in Fig. 6 for the pendulum.

Pendulum-like is a search method which is more directed towards the goal point. In other words, the movement along the directed force for the current best solution is left and right. This process would guide the solution towards the optimum instead of randomly choosing the directions that might divert the solution away from the target point. This would increase the probability that one is nearer and is able to reach the global position. In pendulum-like method, the current best solution is separated into different configurations, which

provides the opportunity to generate solutions adjacent to the target point. After that, the use of i_{freq} counter to vibrate and concentrate the search in the neighborhood of the best current solution, would fine tune the best solution and provide better convergence speed. Fig. 7 shows the pseudo code for the pendulum-like algorithm. Pendulum-like algorithm is dependent on the production of two solutions for each dimension, and this is achieved by moving the current best solution with directed force rightward and leftward. Firstly, the value of the force component should be evaluated according to the current stage using Eq. (27) (Fig. 7, lines 1-2). This is followed with procedures involving positive and negative movement for the dimensions according to the value of $F_{directed}$ in order to determine the intermediate solutions (lines 4-15). After that, the best intermediate point is compared with the current best solution, and if it shows any improvements, the current best solution would be updated (lines 16-19), and the process would vibrate by counter frequency i_{freq} .

1. $\gamma \leftarrow 1/(N_i)^b$
2. $F_{directed} = m_p * g * \gamma$
3. **For** $i_{freq} = 1$ **to** f **do**
4. $int\_point \leftarrow 1$
5. **For** $k = 1$ **to** n **do**
6. $temp \leftarrow \theta^{best}$
7. $\lambda_1 \leftarrow U(0, 1)$
8. $temp_k \leftarrow temp_k + \lambda_1 * F_{directed}$
9. $p^{int\_point} \leftarrow temp$
10. $temp \leftarrow \theta^{best}$
11. $\lambda_1 \leftarrow U(0, 1)$
12. $temp_k \leftarrow temp_k - \lambda_1 * F_{directed}$
13. $p^{int\_point+1} \leftarrow temp$
14. $int\_point \leftarrow int\_point + 2$
15. **End for**
16. $p^{best} \leftarrow \arg\min\{f(p^{int\_point}), \forall int\_point\}$
17. **If** $f(p^{best}) < f(\theta^{best})$ **then**
18. $\theta^{best} \leftarrow p^{best}$
19. **End if**
20. **End for**

Fig. 7: Pseudo code of pendulum-like algorithm.

Implementation of PA-GA Algorithm in Single-Robot Manipulator Workspace:

The system proposed here is the optimization of the time of the task scheduling for robot manipulator that works in the environment with and without static obstacles via the utilization of a PA-GA algorithm. The idea behind this is to move towards a more sophisticated implementation for GA to the combinatorial problem.

I. Initialization of the Population and Representation the Chromosome:

The initial population would be randomly generated, and is put through the operators of GA. The algorithm selects the best chromosome for the next generation at the completion of each generation, and randomly regenerates the rest of the population. Each chromosome has been assigned to a particular sequence. The smallest building block of the individual, the gene, is the element that contains the Cartesian coordinates of each

point of the task. During the evolution, the task sequence is the variable that would change. The chromosome can experience crossover and mutations. In the problem of task scheduling, for each gene of the chromosome would possess coordinates in a Cartesian space, there are quite a number of joint angles specified for each position. The number of these joint angles depends on the DOF of the robot manipulator, where each set of the joint angles in one Cartesian coordinate is considered a solution or configuration. These solutions are calculated by solving the problem of IK at each task point. The population and the chromosome representations for m chromosomes are as shown in Fig. 8.

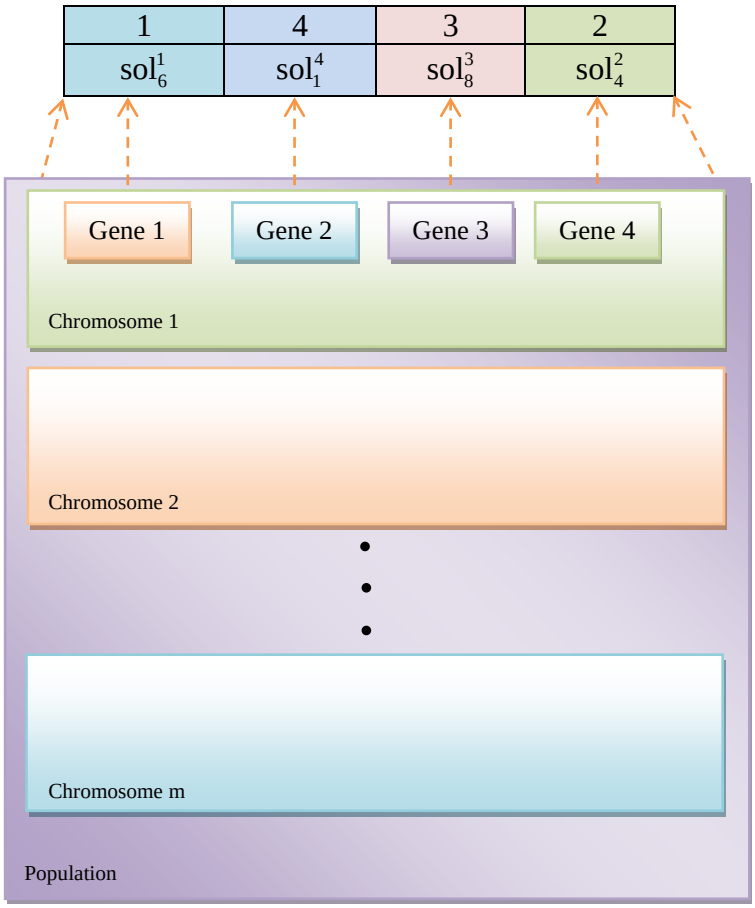

Fig. 8: Sample chromosome within m individual population.

II. Evaluation Function:

A mathematical function used to compute and determine whether the solution is good and it has a vital role because the operators of the algorithm usually use this function. The total objective function which is derived from Eq. (20) and Eq. (22) is:

$$OF_{single} = T_{task} * f_{obst} \tag{30}$$

In case of the workspace without static obstacles

$$OF_{single} = T_{task} \tag{31}$$

Thus, in each generation, the chromosomes are evaluated by Eq. (30) or Eq. (31).

III. Selection Operation:

Using this operator, the chromosomes would be selected for the next generation. In this work, the roulette wheel selection method has been used for the selection procedures. The selection by roulette wheel is used to select the parents for the crossover operation. First, it would distribute the chromosomes to an area, which is

calculated using their objective functions. Next, a random number, N_r in the range from 0-1 is generated. Then, the sum of the area of each chromosome would be examined one by one, until the sum is larger or equal to N_r, and the corresponding chromosome is selected. The process is repeated until the numbers of selected chromosomes are equal to the number of chromosomes that would undergo the crossover operation.

IV. Crossover Operation:

Crossover is defined as the process of producing a child from two selected chromosomes. It is the step that is hoped to give better performance to the offspring. Consider a pair of parent individuals as shown in Fig. 9.

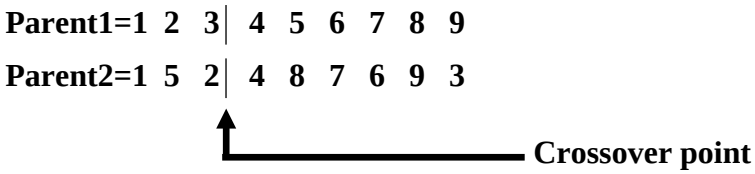

Fig. 9: Two parents with one crossover point.

The crossover point is randomly selected and the section after this point for both parents are exchanged to produce two offspring as shown in Fig. 10. From Fig. 10, it is seen that, some genes are repeated in the offspring that would give illegal tour.

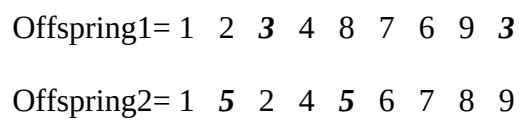

Fig. 10: Offspring with repeated genes after the crossover.

Thus, order crossover has been proposed for the crossover operation (Sivanandam and Deepa, 2008; Carter and Ragsdale, 2006) which is a commonly used TSP operator. In this reproduction operation, two pointers q_1, q_2 to the chromosomes. The range of q_1 is 1 to $m_\gamma - 1$ and the range of q_2 is 2 to m_γ. Then two parents are selected from the pool of selected chromosomes to get the offspring. The two pointers would divide the parents into a left, middle, and right portion. Therefore, the first child inherits its left and right section from the first parent. While middle section would accede to the genes in the middle section of the first parent in the order in which the values appear in the second parent. The second child undergoes similar procedures as shown in Fig. 11.

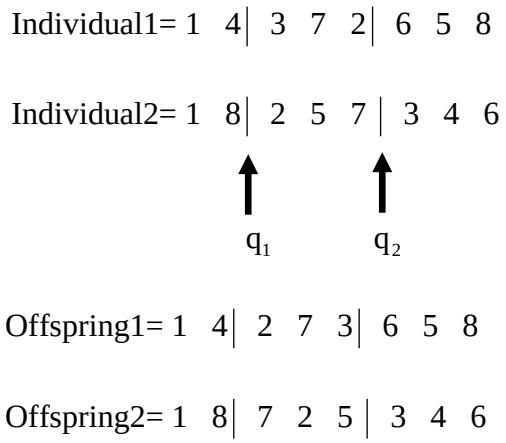

Fig. 11: Order crossover.

V. Mutation Operation:

It is the operation that maintains the diversity of individuals during the generations of GA. Mutation occurs at the gene level in order to avoid the local minima. Suppose that the following legal tour as shown in Fig. 12.

Parent= 2 1 4 3 5

Child= ***2*** 1 ***2*** 3 5

Fig. 12: Mutation example.

Assume that the position 3 is randomly changed with the value of 2, where the value of 4 is replaced with 2. Therefore, illegal tour would be produced in the child because the value of 2 is repeated for two positions.

Consequently, interchanging mutation has been used in this work (Sivanandam and Deepa, 2008; Louis and Li, 2000) which is shown in Fig. 13. In this type of mutation, two random positions were chosen for the parent, and then the genes corresponding to these positions are interchanged, as depicted in Fig. 13. The result of the swap is considered to be valid if it results in an improvement. Otherwise, the tours are discarded. As a result, a good tour never gets lost, and the diversity is always maintained in each generation (Majumdar and Bhunia, 2011).

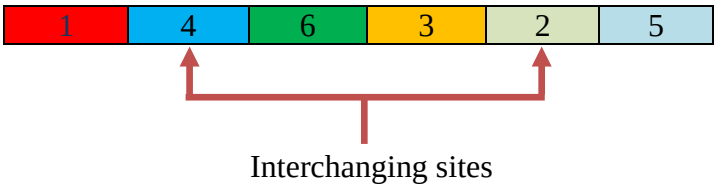

Fig. 13: Interchanging mutation.

Simulation Results:

Two simulators with different numbers and positions of task points have been used to test the effect of the proposed PA-GA algorithm in finding the optimal-time task scheduling for 4-DOF manipulator with the parameters shown in Table 1 and velocity for each joint is being 0.8 rad /s.

Table 1: Manipulator parameters.

Robot parameters	Values
Length for each link	20 cm
Limit for all joint except third joint	[0°,180°]
Limit of third joint	[0°,360°]

I. Six Points Task Scheduling Simulator:

Six task points have been scattered in the environment to represent the Cartesian points that should be visited by the robot. This task has been tested for two scenarios: with static obstacles, and without static obstacles.

a. Six Points Environment Clutter with Static Obstacles:

Different positions and sizes of static obstacles are used for this simulation as depicted in Fig. 14 which shows the full movement for the robot manipulator between the task points. Figure 15 shows the robot configuration at each task point for optimal task found. The optimal or near optimal sequence for this simulation is 1-6-4-5-3-2-1, with cycle time 6.47 s and total error for all points being 3.05E-9 cm. Besides, Fig. 16 shows the movement of the manipulator's tip between the task points. Thus, in this simulator, the robot was able to avoid collision at the task points with the static obstacles scattered inside the workspace.

During the search for the optimal or near optimal solution for the sequence of the tour, PA-GA searches in a large search space to realize that solution. Fig. 17 shows that the PA-GA began with a complex space, which initially has many collisions with the static obstacles. Due to the algorithm being assigned with a large number for the case when there is a collision, the convergence of the objective value begins from a large value towards the minimum time, with an acceptable number of generations.

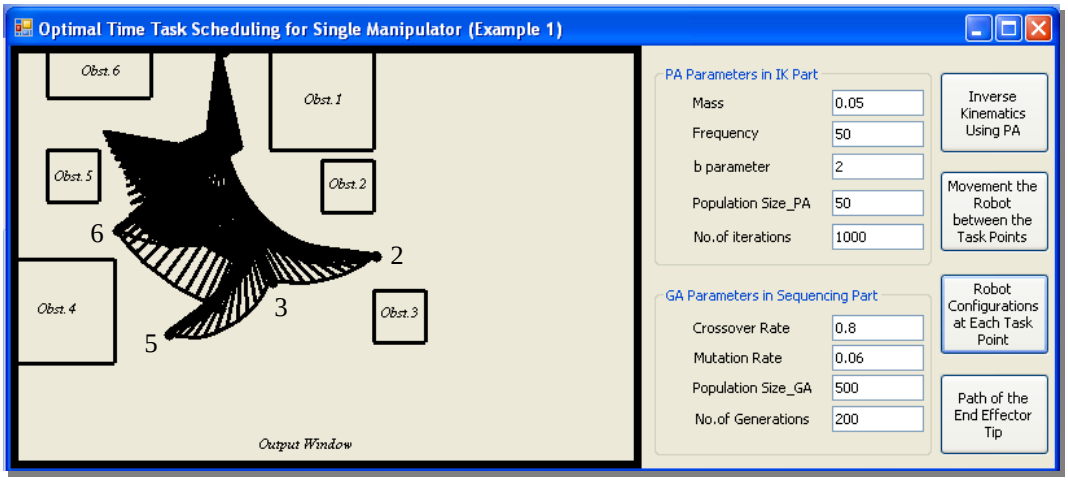

Fig. 14: Simulator window and the parameters of the simulation of six points task in present of static obstacles.

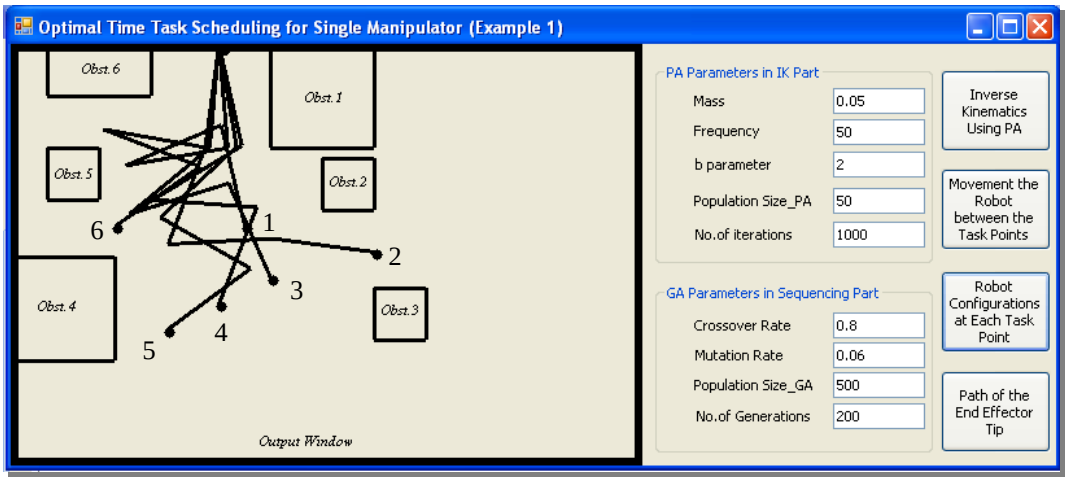

Fig. 15: Manipulator configurations at the tour of six points in present of static obstacles.

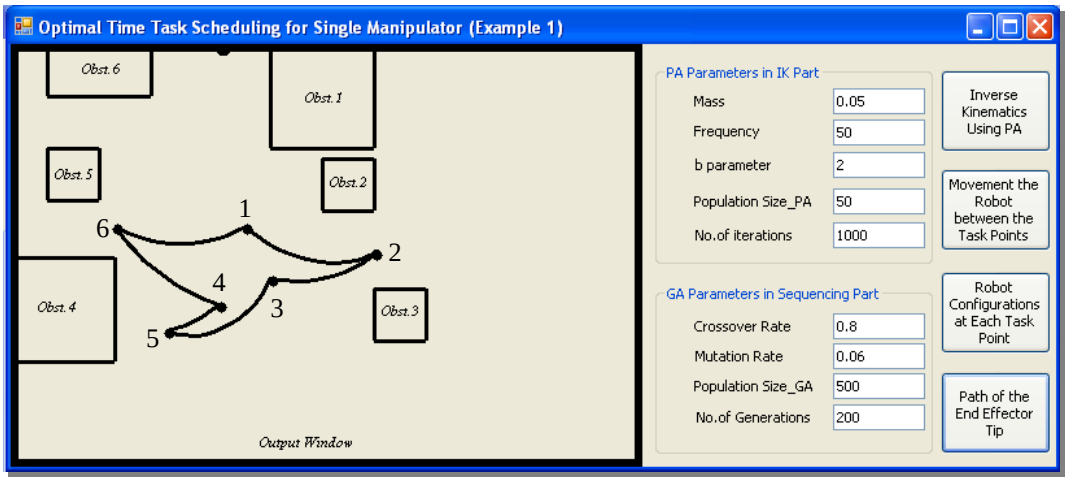

Fig. 16: Near optimal solution for trajectory of the manipulator's end effector along the six points in the present of the obstacles

b. Six Points Collision Free Workspace Simulator:

The second case simulation type was executed when the task points are distributed in a collision free environment. In this workspace, the robot move freely between the task points with less difficulty and shorter time. However, the algorithm searches for the best configurations in order to minimize the cycle time. Figs. 18,

19, and 20 show the full movement, the robot configuration at each task points, and the path of the end effector's tip, respectively. The optimal sequence is 1-6-5-4-3-2-1 with cycle time being 4.06 s and the total error for all the task point is 3.70E-9 cm. Compared to the same task which contained obstacles, the cycle time is decreased by 37.24%.

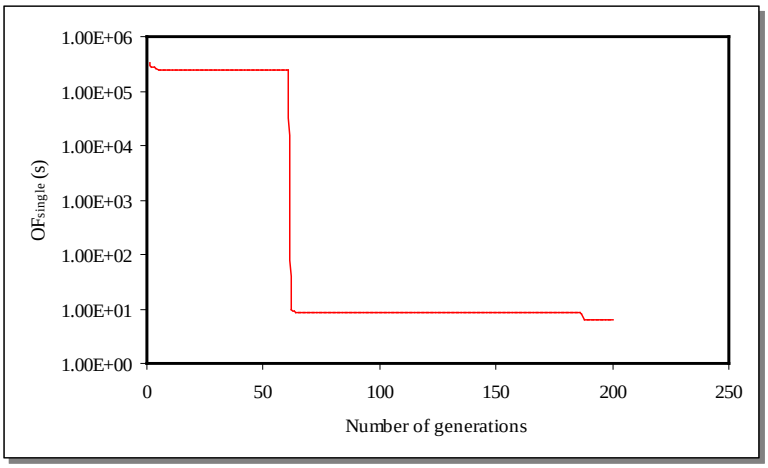

Fig. 17: Objective values decreasing with an increase in generations for six points task and with present of obstacles.

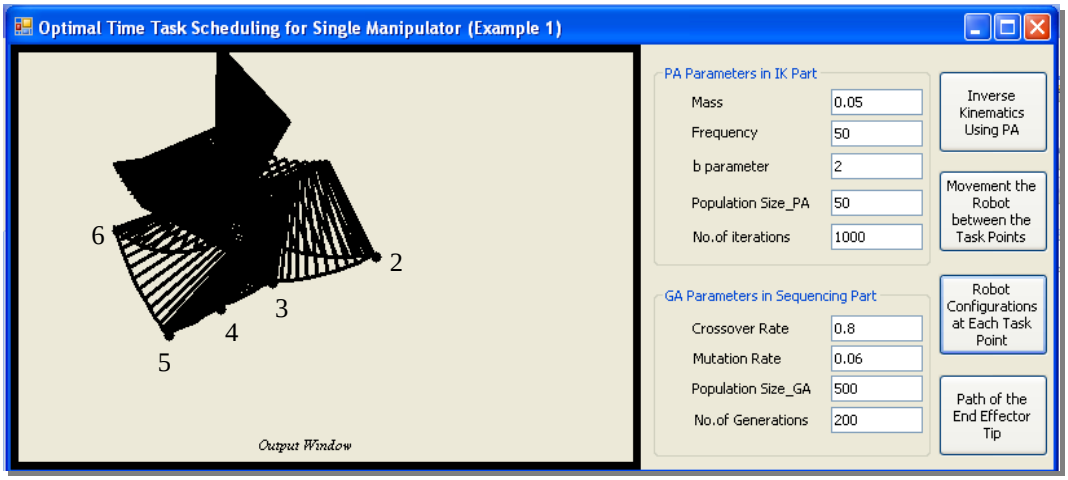

Fig. 18: Simulator window and the parameters of the simulation of six task points without static obstacles.

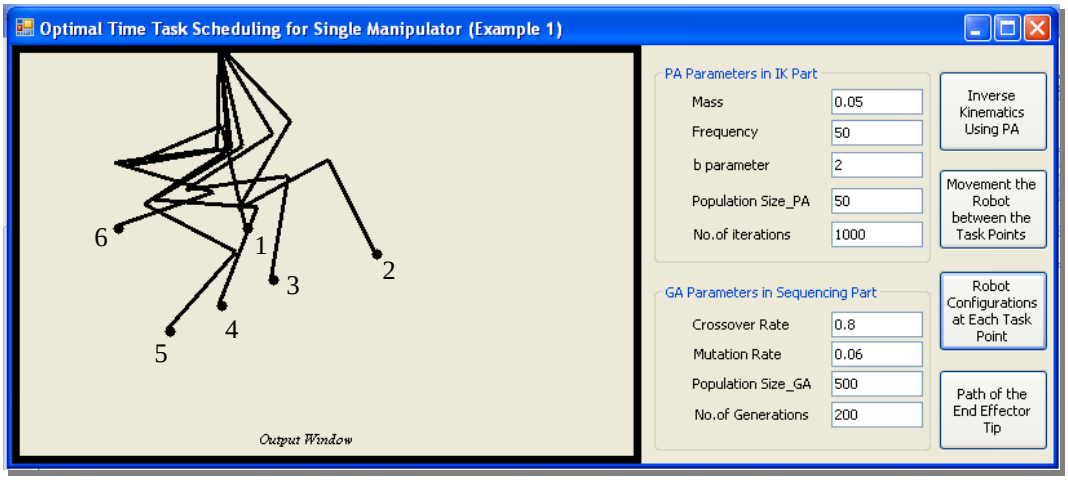

Fig. 19: Manipulator configurations at the tour of six points without static obstacles.

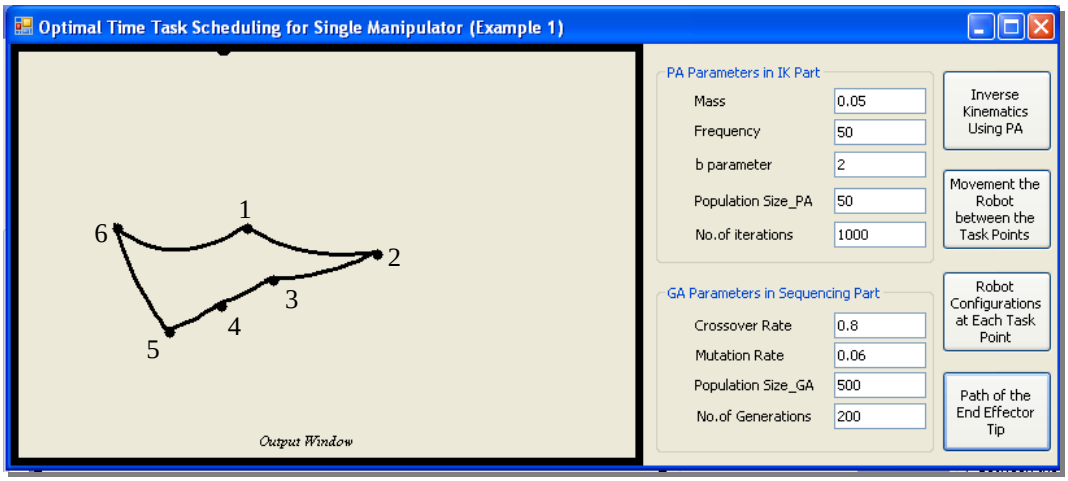

Fig. 20: Near optimal solution for trajectory of the manipulator's end effector along the six points without static obstacles.

The search for a near optimal solution in the absence of static obstacles is easier than the search in the task with obstacles. Fig. 21 shows the convergence of the algorithm towards the optimal time.

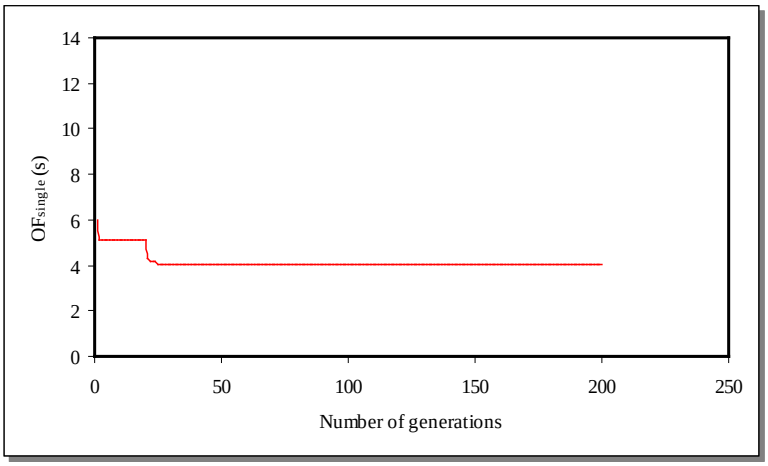

Fig. 21: Objective values deceasing with an increase in generations for six points task and in the absence of obstacles.

II. Ten Points Task Scheduling Simulator:

In this proposed task, ten points have been suggested to be visited by the robot manipulator. The robot would go through these points in present and absence the static obstacles.

a. Ten Points Environment Clutter with Static Obstacles:

Same sized obstacles but with different positions are used for this simulation as illustrated in Fig. 22 which displays the full movement for the robot manipulator between ten task points. Besides, the possible configurations at the task points are shown in Fig. 23. The optimal or near optimal sequence for this simulation is 1-9-6-5-7-8-3-4-2-10-1, while the total cycle time and total error for all points being 9.59 s and1.04E-8 cm, respectively. During this tour, the end effector tip passes through the ten Cartesian points as shown in Fig. 24.

The increase of the number of task points is one of the factors that increase the cycle time. Fig. 25 shows the capability of PA-GA for the search in the workspace with more task points. The objective value is high at the beginning of the search but it gradually reduces towards the near optimal solution.

b. Ten Points Collision Free Workspace Simulator:

Ten points task has also been tested in the absence of the static obstacles. In this case, the environment is less complex which helps the proposed method to achieve less cycle time. Figs. 26, 27, and 28 described this simulator. The near optimal sequence is 1-3-8-6-7-5-4-9-2-10-1 with a total time of 7.88 s, while total error for all points is 5.19E-9 cm. Thus, the time is reduced by 17.83% compared to the simulator with static obstacles.

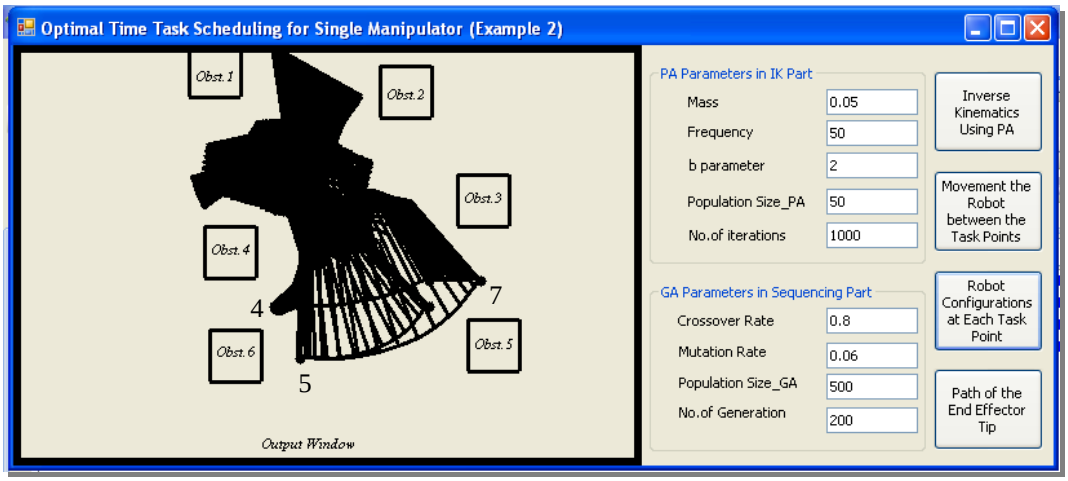

Fig. 22: Simulator window and the parameters of ten points task with static obstacles.

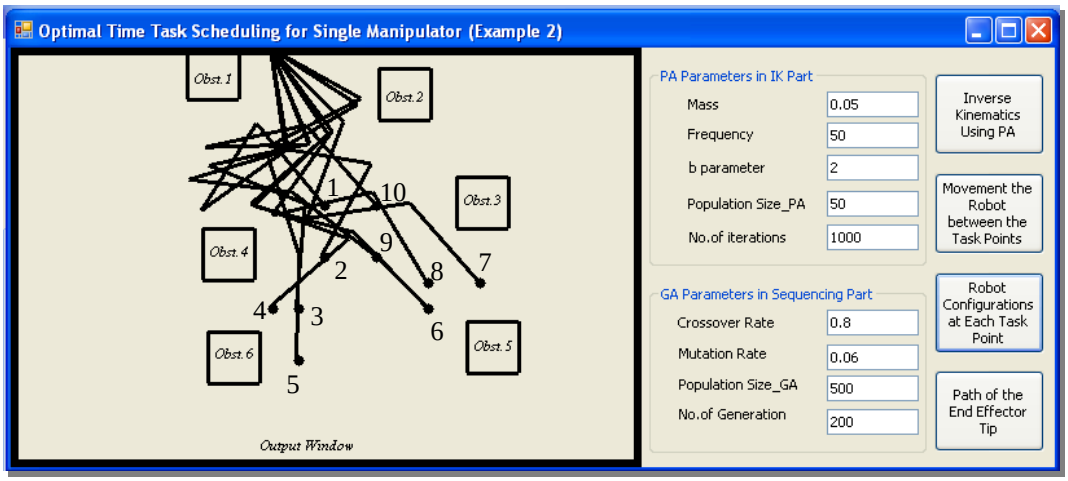

Fig. 23: Manipulator configurations at the tour of ten points with static obstacles.

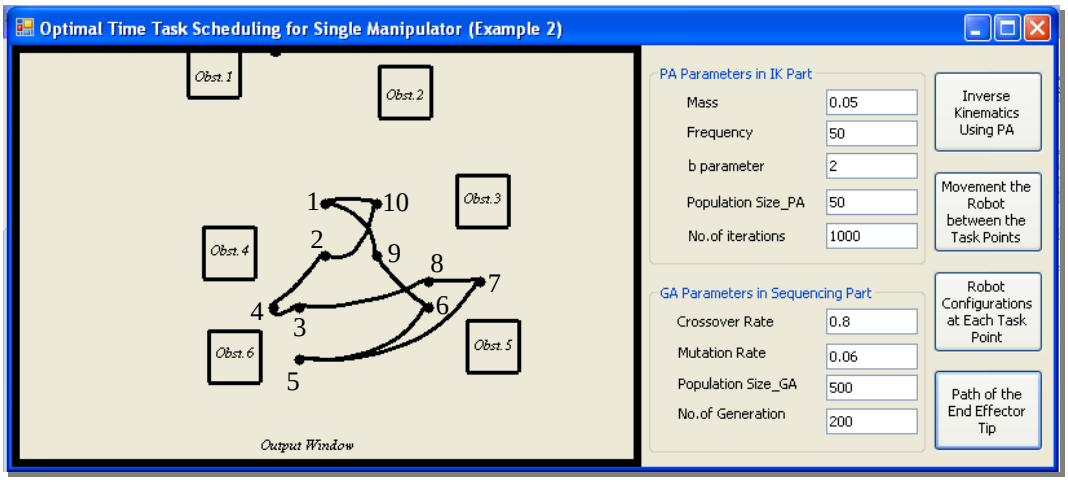

Fig. 24: Near optimal solution for trajectory of the manipulator's end effector along the ten points with static obstacles.

Even if the workspace does not contain static obstacles, the starting time is still higher than the simulator of six task points without static obstacles. This is probably due to the increase in the number of task points for this simulator as depicted in Fig. 29.

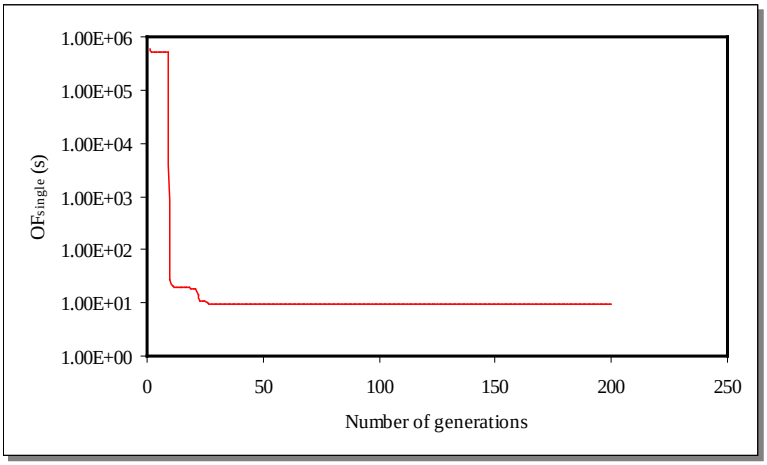

Fig. 25: Objective values decreasing with an increase in generations for ten points task and with obstacles.

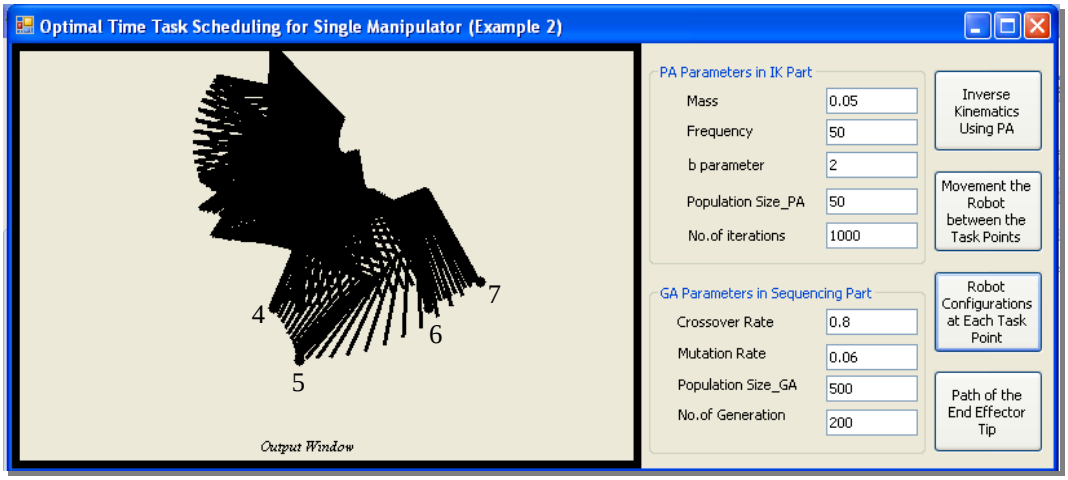

Fig. 26: Simulator window and the parameters of simulation with ten points in absence of static obstacles.

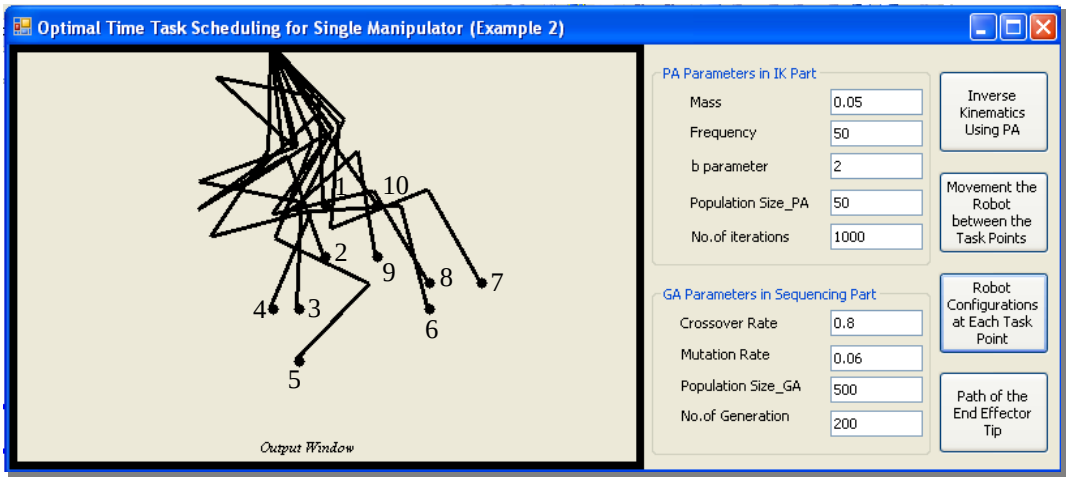

Fig. 27: Manipulator configurations at the tour of ten points in the absence of obstacles.

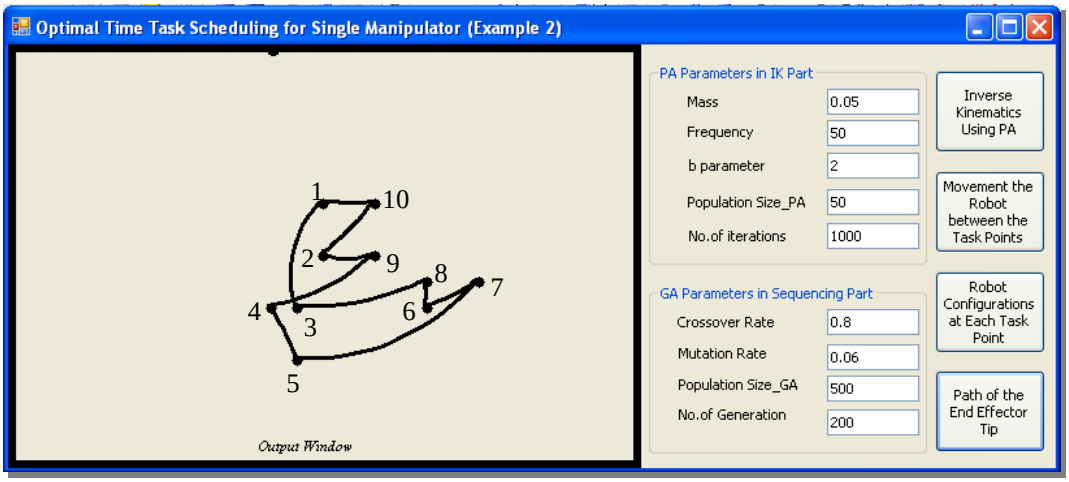

Fig. 28: Near optimal solution for trajectory of the manipulator's end effector along the ten points in the absence of static obstacles.

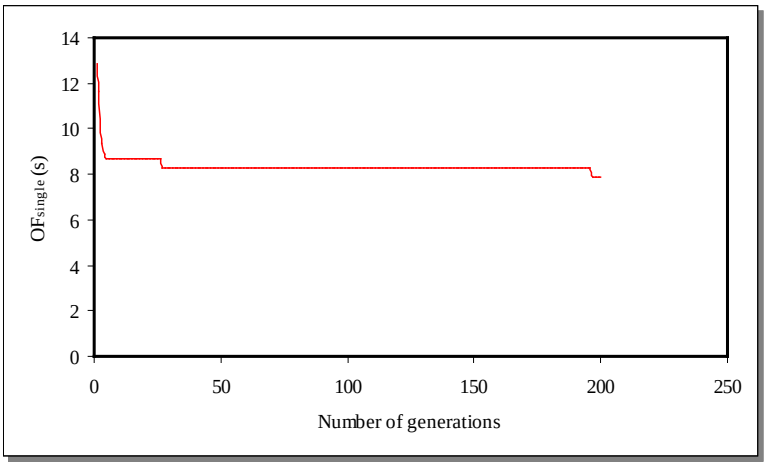

Fig. 29: Objective values decreasing with an increase in generations for ten points task and with absence of obstacles.

Conclusion:

In this paper, the proposed pendulum-like algorithm which is derived from the idea of the simple physical pendulum has been applied to solve different optimization scenarios in IK problem. Eventually, the PA algorithm is combined with GA in order to solve the overall optimal-time task scheduling problem for robot manipulators in the present and absent of static obstacles. The performance optimization of the single-robot manipulator is based on the TSP algorithm. However, it is more complicated due to having multiple solutions at the task points and the possibility of collisions. From the results, the range of the cycle time for single-robot manipulator with obstacles is from 6.47 s to 9.59 s, while the range of cycle time without static obstacles is from 4.06 s to 7.88 s, which show the effect of increasing the task points and the present of static obstacles. As a conclusion, the PA-GA algorithm can perform a more precise position, where the accuracy is achieved up to 3.05E-9 cm for the given task points.

REFERENCES

Abed, I.A., S.P. Koh, K.S.M. Sahari, S.K. Tiong and D.F. Yap, 2012. Comparison Between Genetic Algorithm and Electromagnetism-Like Algorithm for Solving Inverse Kinematics. World Applied Sciences Journal, 20(7): 946-954.

Alavandar, S. and M.J. Nigam, 2008. Neuro-fuzzy based approach for inverse kinematics solution of industrial robot manipulators. Int. J. of Computers, Communications & Control, 3(3): 224-234.

Ata, A.A. and T.R. Myo, 2005. Optimal Point-to-Point Trajectory Tracking Redundant Manipulators using Generalized Pattern Search. International Journal of Advanced Robotic Systems, 2(3): 239-244.

Ata, A.A., 2007. Optimal trajectory planning of manipulators: a review. Journal of Engineering Science and Technology, 2(1): 32-54.

Azariadis, P.N. and N.A. Aspragathos, 2005. Obstacle representation by bump-surfaces for optimal motion-planning. Robotics and Autonomous Systems, 51(2): 129-150.

Beaumont, R.G. and R.M. Crowder, 1989. Two-armed robot systems-a review of current theory and the development of algorithms for real-time collision avoidance. In IEE Colloquium on Controllers for Robotic Applications-Concepts and Implementations. IET, pp: 1-4.

Beaumont, R.G. and R.M. Crowder, 1991. Real-time collision avoidance in two-armed robotic systems. Computer-aided engineering journal, 8(6): 233-240.

Birbil, Ş.İ. and S.C. Fang, 2003. An electromagnetism-like mechanism for global optimization. Journal of Global Optimization, 25: 263-282.

Birbil, Ş.İ., S.C. Fang and R.L. Sheu, 2004. On the convergence of a population-based global optimization algorithm. Journal of global optimization, 30(2-3): 301-318.

Carter, A.E. and C.T. Ragsdale, 2006. A new approach to solving the multiple traveling salesperson problem using genetic algorithms. European journal of operational research, 175(1): 246-257.

Chang, P.C., S.H. Chen and C.Y. Fan, 2009. A hybrid electromagnetism-like algorithm for single machine scheduling problem. Expert Systems with Applications, 36(2): 1259-1267.

Craig, J.J., 2005. Introduction to robotics: mechanics and control. 3rd Edn., Pearson Prentice Hall.

Edan, Y., T. Flash, U.M. Peiper, I. Shmulevich and Y. Sarig, 1991. Near-minimum-time task planning for fruit-picking robots. IEEE Transactions on Robotics and Automation, 7(1): 48-56.

Filipović, V., A. Kartelj and D. Matić, 2013. An electromagnetism metaheuristic for solving the Maximum Betweenness Problem. Applied Soft Computing, 13: 1303-1313.

Gilak, E. and H. Rashidi, 2009. A New Hybrid Electromagnetism Algorithm for Job Shop Scheduling. 3rd UK Sim European Symposium on Computer Modeling and Simulation. IEEE, pp: 327-332.

Guo, D., H. Ju, Y. Yao, F. Ling and T. Li, 2009. Efficient Algorithms for the Kinematics and Path Planning of Manipulator. In International Conference on Artificial Intelligence and Computational Intelligence. IEEE, pp: 282-287.

Jasim, W.M., 2011. Solution of Inverse Kinematics for SCARA Manipulator Using Adaptive Neuro-Fuzzy Network. International Journal Soft Computing, 2(4): 59-66.

Jhang, J.Y. and K.C. Lee, 2009. Array pattern optimization using electromagnetism-like algorithm. AEU-International Journal of Electronics and Communications, 63(6): 491-496.

Jolai, F., R. Tavakkoli-Moghaddam, A. Golmohammadi and B. Javadi, 2012. An Electromagnetism-like algorithm for cell formation and layout problem. Expert Systems with Applications, 39(2): 2172-2182.

Khan, F.H., N. Khan, S. Inayatullah and S.T. Nizami, 2009. Solving TSP problem by using genetic algorithm. International Journal of Basic & Applied Sciences IJBAS, 9(10): 79-88.

Lai, K.C. and S.C. Kang, 2009. Collision detection strategies for virtual construction simulation. Automation in construction, 18(6): 724-736.

Lee, C.H. and F.K. Chang, 2010. Fractional-order PID controller optimization via improved electromagnetism-like algorithm. Expert Systems with Applications, 37(12): 8871-8878.

Louis, S.J. and G. Li, 2000. Case injected genetic algorithms for traveling salesman problems. Information Sciences, 122(2): 201-225.

Majumdar, J. and A.K. Bhunia, 2011. Genetic algorithm for asymmetric traveling salesman problem with imprecise travel times. Journal of computational and applied mathematics, 235(9): 3063-3078.

Marcos, M.D.G., J.A. Tenreiro Machado and T.P. Azevedo-Perdicoúlis, 2012. A multi-objective approach for the motion planning of redundant manipulators. Applied Soft Computing, 12(2): 589-599.

Miao, M. and J. Jiang, 2012. Electromagnetism-like Mechanism Algorithm Based on Normalization and Adaptive Move Operator. Journal of Computational Information Systems, 8(18): 7449-7455.

Narasimhan, M.V., 2006. Optimization Of The Tool Path In A Robotic Environment, PhD Thesis, The University of Texas at Arlington.

Nearchou, A.C. and N.A. Aspragathos, 1997. A genetic path planning algorithm for redundant articulated robots. Robotica, 15(2): 213-224.

Petiot, J.F., P. Chedmail and J.Y. Hascoët, 1998. Contribution to the scheduling of trajectories in robotics. Robotics and Computer-Integrated Manufacturing, 14(3): 237-251.

Rana, A.S. and A.M.S. Zalzala, 1997. Collision-free motion planning of multiarm robots using evolutionary algorithms. Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering, 211(5): 373-384.

Rocha, A.M.A. and E.M. Fernandes, 2008. Implementation of the electromagnetism-like algorithm with a constraint-handling technique for engineering optimization problems. 8th International Conference on Hybrid Intelligent Systems. IEEE, pp: 690-695.

Sivanandam, S.N. and S.N. Deepa, 2008. Introduction to Genetic Algorithms. Springer-Verlag Berlin Heidelberg, New York.

Su, C.T. and H.C. Lin, 2011. Applying electromagnetism-like mechanism for feature selection. Information Sciences, 181(5): 972-986.

Tsou, C.S. and C.H. Kao, 2006. An electromagnetism-like meta-heuristic for multi-objective optimization. IEEE Congress on Evolutionary Computation. IEEE, Sheraton Vancouver Wall Centre Hotel, Vancouver, Canada, pp: 1172-1178.

Xidias, E.K., P.T. Zacharia and N.A. Aspragathos, 2010. Time-optimal task scheduling for two robotic manipulators operating in a three-dimensional environment. Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering, 224(7): 845-855.

Xidias, E.K., P.T. Zacharia and N.A. Aspragathos, 2010. Time-optimal task scheduling for articulated manipulators in environments cluttered with obstacles. Robotica, 28(03): 427-440.

Yahya, S., M. Moghavvemi and H.A. Mohamed, 2011. Geometrical approach of planar hyper-redundant manipulators: Inverse kinematics, path planning and workspace. Simulation Modelling Practice and Theory, 19(1): 406-422.

Yampolskiy, R.V., L. Ashby and L. Hassan, 2012. Wisdom of Artificial Crowds-A Metaheuristic Algorithm for Optimization. Journal of Intelligent Learning Systems and Applications, 4(2): 98-107.

Yao, Z. and K. Gupta, 2007. Path planning with general end-effector constraints. Robotics and Autonomous Systems, 55(4): 316-327.

Zacharia, P.T. and N.A. Aspragathos, 2004. Optimization of industrial manipulators cycle time based on genetic algorithms. In 2nd IEEE International Conference on Industrial Informatics. IEEE, pp: 517-522.

Zacharia, P.T. and N.A. Aspragathos, 2005. Optimal robot task scheduling based on genetic algorithms. Robotics and Computer-Integrated Manufacturing, 21(1): 67-79.

Zhang, P., X. Mu, Z. Ma and F. Du, 2012. An adaptive PSO-based method for inverse kinematics analysis of serial manipulator. International Conference on Quality, Reliability, Risk, Maintenance, and Safety Engineering, IEEE, pp: 1122-1126.