Arab J Sci Eng (2014) 39:6269–6285
DOI 10.1007/s13369-014-1250-0

RESEARCH ARTICLE - ELECTRICAL ENGINEERING

Optimization of the Time of Task Scheduling for Dual Manipulators using a Modified Electromagnetism-Like Algorithm and Genetic Algorithm

Issa Ahmed Abed · S. P. Koh · Khairul Salleh Mohamed Sahari · P. Jagadeesh · S. K. Tiong

Received: 7 December 2012 / Accepted: 31 August 2013 / Published online: 1 July 2014

Abstract A method based on a modified electromagnetism-like with two-direction local search algorithm (MEMTDLS) and genetic algorithm (GA) is proposed to determine the optimal time of task scheduling for dual-robot manipulators. A GA is utilized to calculate the near-optimal task scheduling for both robots, and the MEMTDLS is recommended as a suitable alternative in obtaining multiple solutions at each task point for both manipulators with minimal error. During the course of the tour, the robots move from point to point with a short cycle time, while ensuring that no collision occurs between the two manipulators themselves or between the dual manipulators and the static obstacles in the workspace. The movement and the configurations of the manipulators at the task points were illustrated using a simulator that was developed via Visual Basic .Net. The method is verified using two simulators that are used as examples for two identical four-link planar robots that work in the environment, with square-shaped obstacles cluttered at different locations.

Keywords Attraction–repulsion mechanism · Genetic algorithm · Task scheduling · Two robots

الخلاصة

تم عرض طريقة تعتمد على خوارزمية معدّلة على غرار الكهرومغناطيسية مع بحث محلي باتجاهين وخوارزمية جينية لتحديد الوقت المثالي لجدولة المهام لمناولات روبوت مزدوجة. وقد استُخدمت الخوارزمية الجينية لحساب جدولة المهام القريبة من المثالية لكلا الروبوتين ، ويوصى بالخوارزمية المعدّلة على غرار الكهرومغناطيسية مع بحث محلي باتجاهين كبديل مناسب للحصول على حلول متعددة عند كل نقطة مهام لكلا المناولين وبأدنى خطأ. ويتحرك الروبوتان خلال الجولة من نقطة إلى أخرى في دورة زمنية صغيرة بينما يتم التأكد من عدم حدوث تصادم بين المناولين نفسيهما أو بين المناولين المزدوجين وبين العوائق الثابتة في مساحة العمل. وقد تم توضيح حركة وشكل المناولين عند نقاط المهام باستخدام برنامج محاكاة تم تطويره عبر برنامج فيجوال بيسك. نت. وقد تم التأكد من الطريقة باستخدام محاكيين تم استخدامهما كمثالين لروبوتي مستوى أربع وصلات متماثلين يعملان في البيئة مع عوائق مربعة الشكل موزعة عشوائياً في مواقع مختلفة.

1 Introduction

The use of two robotic manipulators guarantees particular advantages over the use of one robotic system. These advantages include increase in productivity, reduction of costs, improved product quality, and the simplification of complex tasks [1].

Many researchers have utilized artificial intelligence, such as genetic algorithms (GAs) and neural networks, to solve the inverse kinematics (IK) problem at any given point. In this regard, Karlik and Aydin [2] proposed an improved approach by finding the best configuration of the artificial neural network (ANN) to solve the inverse kinematics of a six-degree of freedom (DOF) robot manipulator. They proposed the ANN approach to control the motion of a robot manipulator, and the learning equations that they used were those of the back-propagation algorithm. Two different configurations for the ANN were constructed. In the first configuration, the neurons were fully connected to each output with one hidden layer. In the second configuration, each output of a neural

I. A. Abed (✉)
Technical College Basrah, Foundation of Technical Education, Baghdad, Iraq
e-mail: issaahmedabd80@yahoo.com

S. P. Koh · K. S. M. Sahari · P. Jagadeesh · S. K. Tiong
College of Engineering, Universiti Tenaga Nasional, Selangor, Malaysia

network was designed with two hidden layers. The result of their second configuration was better than that of the first. ANN architecture took into account the Cartesian coordinates for position, Euler angles to represent orientation as inputs, and joint angles as the outputs. However, a four-layer neural network has already been proposed for predicting IK solutions, and an appropriate computer program has already been developed using the Borland C++ language for the ANN architectures that are being considered in their study.

The iterations in the study of Karlik and Aydin were performed on a PC P-90 computer, and 6,000 iterations were used for teaching the ANN. However, their work utilized a very large data set without mentioning the possibility of multiple solutions for IK. A back-propagation neural network equipped with a sigmoidal activation function was designed by Köker et al. [3] to address the problem of IK for a three-joint robot. The authors selected specific points from the workspace of the manipulator. After selecting those points, they used them in the cubic polynomial to generate the set of angles according to each (x, y, z) point. All of these values were then recorded in a file, which formed the learning set of the neural network. They sampled 6,000 for the trajectory of a given job, where 5,000 were used during training, and the rest were used in testing. They then designed the neural network to solve the IK for the robot. However, the training process was stopped by checking the error, which comprised the difference between the end effector and the target point according to the distance equation. The designed neural network thus gave the angles according to the given (x, y, z) Cartesian coordinates. The authors designated large data sets as a requirement for acquiring better performances for the purpose of learning. Kalra et al. [4] suggested the use of an evolutionary approach based on single-level genetic algorithm to solve the IK for industrial robots. The proposed method used a fitness function, which was defined in a manner that required the separate evaluation of the positional error of the robot and the total joint displacement. The authors obtained an acceptable distribution of population members around the multiple IK solutions by using a niche strategy that involves the use of a binary tournament selection operator and mating restriction. This technique was tested on two robotic configurations, namely the SCARA and PUMA robots. The SCARA-configured robot managed to generate one or two solutions for each target point, whereas the PUMA-configured robot generated four solutions for each wrist location. They also introduced minimal positional errors of the wrist for the obtained values of the joint variables by using two niche strategies, namely niching strategy 1 and niching strategy 2. The position errors obtained from the niche strategies 1 and 2 for the SCARA robot were in the ranges of 0.40–2.19 and 0.14–0.69 mm, respectively. Thus, fewer positional errors were obtained in niche strategy 2 than niche strategy 1. The positional errors for the PUMA robot when the niche strategy 2 was used fell within the range of 0.36–1.91 mm. Overall, the simulation results proved that the best solutions for the SCARA robot were obtained after 100 generations, whereas the best solutions for the PUMA robot were obtained after 300 generations. The emergence of electromagnetism-like (EM) algorithm as a relatively recent artificial intelligence method manages to solve the IK problem. Nonetheless, there are very few researchers who used EM algorithm to do so. Feng et al. [5] came up with a hybrid algorithm that is based on a modified Davido–Fletcher–Powell (MDFP) and EM algorithm for computing numerical solutions to the IK problem for robotic manipulators. They posit that the problem can be easily solved using the EM algorithm, and that the EM algorithm can easily be combined with other optimization methods. Based on the approximation solution given by the EM algorithm, a modified DFP algorithm was developed with the main purpose of enhancing this solution. This group modified the DFP algorithm such that it randomly chooses the search step size between 0 and 1. The algorithm was tested for some test functions, and due to the IK problem could be transformed into an equivalent minimization problem, which allowed them to test the method by solving the inverse kinematics for the PUMA robot. Their algorithm was executed in the C/C ++ environment.

In summary, in methods that used ANN, several problems are encountered in solving the IK problem by using ANN. The first problem is the selection of the appropriate type of neural network and the generation of a suitable training data set. In order to map the Cartesian configuration into corresponding joint angles, the ANN is used to approximate the IK relations of the robot manipulator. A large number of training data and iterations are used to improve learning performance. Thus, the provision of such large data set is difficult, and the data, which is obtained from deriving the IK equations, might contain mapping error due to the nonlinear mapping between the joint angle coordinates and Cartesian coordinates, leading to inaccuracies in the predicted IK solutions [5]. Some of these methods are capable of providing the inverse kinematics solution for 2 and 3 DOF, but they demanded high-performance computing systems and complex computer programming for obtaining the solutions of more DOF [5]. In the methods that utilized the gradient approach, the selection of the step size is important in improving the convergence because an inferior choice for this parameter will compel it to go through an infinite number of iterations before ending up with a solution, thereby making the process excruciatingly slow. On the other hand, a larger step size will increase the speed of convergence; however, it may also result in a large error.

Many optimization methods for reducing the cycle time of the manipulator have been presented in previous studies. Addressing this concern, Edan et al. [6] 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. However, the authors did not take into account the possibility of collision when designing the algorithm and they did not mention the possible configurations used to determine the optimal sequence. Petiot et al. [7] succeeded in showing the potential of the elastic-net method to minimize the cycle time of robots. The algorithm scheduled the trajectory points in such a way that it gave minimum time, although it was 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 DOF robot. However, a highly cluttered environment could still lead to failure. A GA is an optimization method that retains the ability to search in the midst of large complex spaces [8]. Obtaining the derivative of the function is unnecessary in GA. Furthermore, GA is applicable to nonlinear and non-continuous functions, which makes it a good algorithm for efficiently solving the path planning of a redundant robot [9]. GA is also recommended as the most suitable algorithm for finding the minimum distance of a collision-free path for two-arm robots [10]. Zacharia and Aspragathos [11] proposed a method based on GA that optimizes the sequence of points that are visited by the end effector. Their algorithm takes into account the multiple solutions of the IK problem. However, their technique utilizes a non-redundant manipulator in a collision-free workspace. An optimization algorithm based on GA was introduced by Xidias et al. [12] with the intention of determining the near-optimal sequence of task points with no possibility of collision between the robot and static obstacles. The algorithm is applicable for both redundant and non-redundant robots, and it also integrates multiple existing solutions at each task point. Jiang et al. [13] suggested an assembly-scheduling method, which produces optimal time solutions for a two-robot cell. In order to completely reduce the possibility of collision, only one robot is allowed to enter the assembly area at any given time. This constraint is assumed when deriving the assembly-planning algorithm. The problem of optimal task scheduling for dual manipulators is akin to the traveling salesman problem (TSP); this problem describes the dilemma of a salesman visiting a number of cities once and returning to the first city via the shortest possible route [14]. In the problem of optimal task scheduling for dual robots, the gauge is time instead of distance, and the problem is more intricate than the TSP. In optimal task scheduling of dual manipulators, particular issues regarding the robots, such as collision avoidance, are considered. Two salesmen (robot manipulators) are then used in the workspace, and multiple robot configurations at each task point (city) exist, instead of a single solution. An approach based on GA has been achieved by Xidias et al. [15] in obtaining the near-optimal task scheduling for two robots. Multiple solutions for each task point for non-redundant robots were taken into account. The two robots moved between the task points in a 3D environment, but they ensured that no collisions between the two manipulators would occur at these task points.

This work discusses a method that combines modified electromagnetism-like with two-direction local search algorithm (MEMTDLS) and GA in determining the near-optimal time sequence for dual-robot manipulators. Both robots divide the task throughout the environment, which contains static obstacles. This study aims to determine the minimum time for the total task and simultaneously avoid collision between the two manipulators or between the manipulators and the static obstacles. The calculations of the minimum total cycle time hinges on the IK solutions determined by the utilization of the MEMTDLS algorithm. The remainder of the paper is organized as follows: Sect. 2 defines the problem and objectives of the work; Sects. 3, 4, and 5 provide details of the EM, MEMALS and MEMTDLS algorithms; the GA is implemented in Sect. 6; the simulation results are showcased in Sect. 7; and finally, Sect. 8 concludes the study.

2 Definition of the Problem

2.1 Calculation of the IK Solutions

In robotics, the design of any task, such as path planning, task scheduling, and the control of the robot manipulators requires forward and inverse kinematics. The solution of the forward kinematics is simple, because it depends on knowing the link parameters and joint variables. The geometric method is applicable for solving the forward equations of the n-DOF planar robot, shown in Fig. 1 as follows [16]:

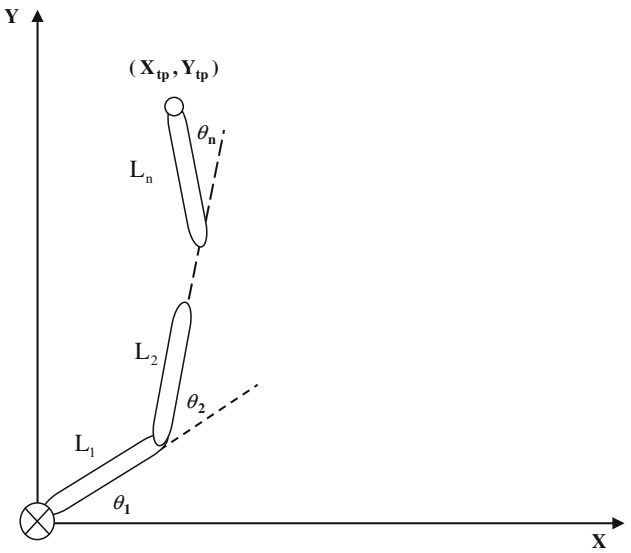


Fig. 1 Top view of the n links planar robot

$$X_{\text{cur}} = L_1 * \cos\theta_1 + L_2 * \cos(\theta_1 + \theta_2) + \cdots + L_n * \cos(\theta_1 + \theta_2 + \cdots + \theta_n) \quad (1)$$

$$Y_{\text{cur}} = L_1 * \sin\theta_1 + L_2 * \sin(\theta_1 + \theta_2) + \cdots + L_n * \sin(\theta_1 + \theta_2 + \cdots + \theta_n) \quad (2)$$

where L_n denotes the nth link length, θ_n is the nth joint angle, and $(X_{\text{cur}}, Y_{\text{cur}})$ is the current solution at any point of the task. Obtaining $(\theta_1, \theta_2, \ldots, \theta_n)$ in terms of the $(X_{\text{cur}}, Y_{\text{cur}})$ is aptly named the inverse kinematics. The IK is challenging to solve because the forward equations are nonlinear and generate infinite solutions at given target (X_{tp}, Y_{tp}) for redundant robots. Therefore, the MEMTDLS method is used to minimize the error of using the forward equations for the robot manipulators.

The positional tracking error between the current solution and the position of the target according to Yao and Gupta [17] is:

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

The EM algorithm solves for different solutions at each task point [5,18]. Hence, the fitness function for the MEMTDLS is:

$$\text{fitness\_MEMTDLS} = \frac{1}{1 + f_{\text{error}}(\theta)} \quad (4)$$

The trajectory between point (i) and point $(i+1)$ in the task is derived as follows [19,20]:

$$\theta_{i,i+1}(t) = C_{i0} + C_{i1}t + C_{i2}t^2 + C_{i3}t^3 \quad (5)$$

where $C_{i0}, C_{i1}, C_{i2}, C_{i3}$ are constants, and the constraints are as follows:

$$\theta_i = C_{i0} \quad (6)$$

$$\theta_{i+1}(t_f) = C_{i0} + C_{i1}t_f + C_{i2}t_f^2 + C_{i3}t_f^3 \quad (7)$$

$$\dot{\theta}_i = C_{i1} \quad (8)$$

$$\dot{\theta}_{i+1}(t_f) = C_{i1} + 2C_{i2}t_f + 3C_{i3}t_f^2 \quad (9)$$

where t_f is the time interval between point (i) and point $(i+1)$.

In using the cubic polynomial equation (Eq. 5), the constants of Eqs. (6–9) must be determined by solving these equations. The velocity at point (i) and point $(i+1)$ is zero.

2.2 Task Scheduling Problem

After determining the manipulator configurations via the MEMTDLS algorithm, the dual manipulators divide m_t task points and move between all of these task points exactly once before returning to the initial point of launch.

The following equation is used to calculate the cycle time for each robot during the task from point$_{i-1}^{\alpha}$ with solution θ^p to the task point$_i^{\alpha}$ with solution θ^q [12]:

$$T_{\text{travel}}^{\alpha} = \sum_{i=2}^{m_\alpha} \max_j \left(\frac{\left|\theta_{ji}^q - \theta_{j(i-1)}^p\right|}{\dot{\theta}_j} \right) \quad (10)$$

where $T_{\text{travel}}^{\alpha}$ is the travel time for the α-robot in visiting m_α task points, $\alpha = 1, 2$, and m_α is the number of task points for the α-robot, $j = 1, 2, 3, \ldots, n$, where n is the number of DOF, $p, q = 1, 2, 3, \ldots, s$, where s is the number of solutions; θ_{ji} is the joint displacement at joint j in the task point i, and $\dot{\theta}_j$ is the average velocity of joint j. Thus,

$$T_{\text{return}}^{\alpha} = \max_j \left(\frac{\left|\theta_{j1}^q - \theta_{jm_\alpha}^p\right|}{\dot{\theta}_j} \right) \quad (11)$$

where $T_{\text{return}}^{\alpha}$ is the return time to the initial task point of the α-robot.

The total time for the α-robot to travel from the initial point to the m_α and return to the initial point is

$$T_{\text{task}}^{\alpha} = T_{\text{travel}}^{\alpha} + T_{\text{return}}^{\alpha} \quad (12)$$

According to Xidias et al. [15]

$$T_{\text{task\_total}} = \max\left(T_{\text{task}}^1, T_{\text{task}}^2\right) \quad (13)$$

where $T_{\text{task\_total}}$ is the function that should be minimized.

2.3 Collision Avoidance

In obtaining the most viable results, collision must be avoided at all costs to ensure free motion. Therefore, a circle method [9,10,21] is introduced to prevent collisions. The first collision might occur between the links of two manipulators working in an overlapping workspace, where one manipulator might impede the other. The second potential collision may involve the scattered obstacles throughout the workspace. The links of the manipulator are approximated via tangent circles, and the obstacles are enclosed via one circle, as shown in Fig. 2. This procedure makes the collision incumbent upon the sum of the radii of the present circles.

$$f_{\text{obst}} = \begin{cases} 1 & \text{no collision} \\ c & \text{otherwise} \end{cases} \quad (14)$$

where c is assumed to be a large number.

The total function (TF) is:

$$\text{TF} = T_{\text{task\_total}} * f_{\text{obst}} \quad (15)$$

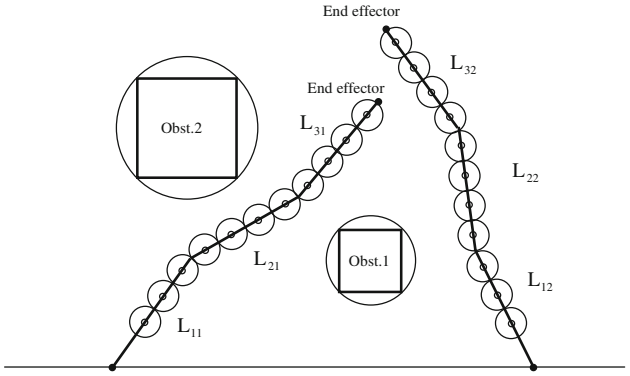


Fig. 2 Circle method for dual robots and obstacles

```
1. Length ← δ(max_k{u_k − l_k})
2. For i = 1 To m do
3.   For k = 1 To n do
4. Counter ← 1
5.   DO
6. λ1 ← U(0,1)
7. y ← θ^i
8. λ2 ← U(0,1)
9.     If λ1 > 0.5 then
10. y_k ← y_k + λ2 *Length
11.    Else
12. y_k ← y_k − λ2 *Length
13.    End if
14.      If f(y) < f(θ^i) then
15. θ^i ← y
16. Counter ← Lsiter -1
17.      End if
18. Counter ← counter +1
19.    Loop while counter < Lsiter
20.  End for
21. End for
22. θ^best ← arg min{f(θ^i), ∀i }
```

Fig. 3 Pseudo code for local search procedures

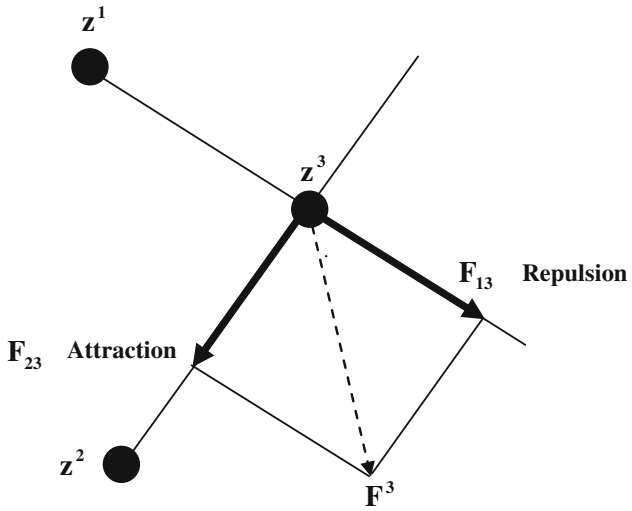


Fig. 4 Attraction–repulsion mechanism

```
1. For i = 1 To m do
2.   If i ≠ best then
3.     λ ← U(0,1)
4. Norm calculation for the force ||F^i||
5.          F^i ← F^i / norm
6.     For k = 1 To n do
7.       If F^i_k > 0 then
8.         θ^i_k ← θ^i_k + λ F^i_k (u_k − θ^i_k)
9.       Else
10.        θ^i_k ← θ^i_k + λ F^i_k (θ^i_k − l_k)
11.      End if
12.    End for
13.  End if
14. End for
```

Fig. 5 Pseudo code of movement procedures

The fitness function for the task scheduling problem is

$$\text{fitness\_GA} = \frac{1}{\text{TF}} \tag{16}$$

3 Review of EM Algorithm

The EM algorithm is derived from the attraction–repulsion theory of physics and is classified as a population-based algorithm [22]. It differs from both the GA and simulated annealing (SA) in terms of the exchange of information among individuals of a population, but bears a striking resemblance to both the particle-swarm optimization (PSO) and ant-colony optimization (ACO), because the particles affect each other in the population [23,24]. The EM algorithm is an appropriate choice for solving continuous problems such as those of IK without taking into consideration the geometry and the DOF of the robots [5]. It approximates a solution more quickly than other algorithms [25], and it is also fairly accurate [26]. The algorithm is divided into four phases, which are:

3.1 Initialization of the Population

A population with m points is randomly generated with n coordinates. The coordinates are uniformly distributed within both the upper and lower bounds [27]. An objective function

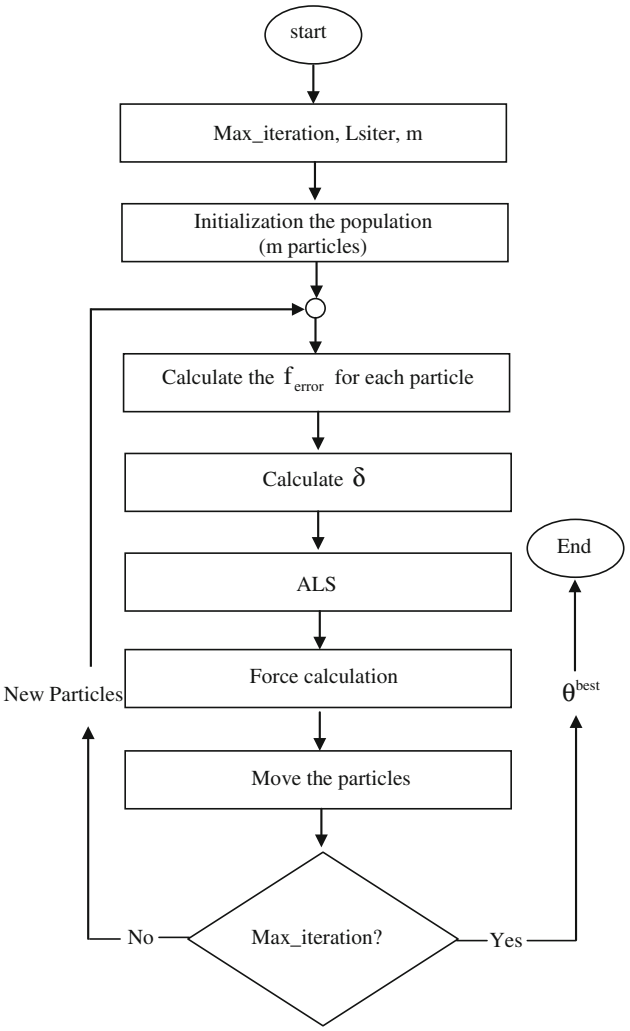


Fig. 6 Flow chart for MEMALS

value for each sample is evaluated after the generation of the samples in the population.

3.2 Local Search

The procedure that searches for a better solution by collecting the localized information from each sample point. Figure 3 displays the pseudo code of the local search procedure [26].

In Fig. 3, m is designated as the number of sample points (population size), n is the dimension for each sample; Lsiter represents the local search iterations, and δ is the local search parameter, with a probable value between 0 and 1. The algorithm begins with the local search procedure, which is initiated after the initialization of the population. The first step is the evaluation of the maximum feasible step length (Length) by using parameter δ (line 1). The second step analyzes the fact that for each i, the initial point θ^i is stored in the temporary point of y. Third, y moves in accordance

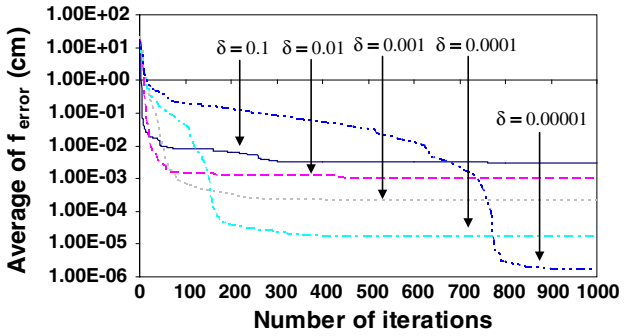


Fig. 7 Average of f_{error} with the iterations for different local search parameter

to the randomly selected number coordinate by coordinate (lines 6–13). Fourth, if the new point in y fares better than θ^i within the Lsiter iterations, then θ^i will be replaced by y, and the search for this θ^i will come to an end. If this does not occur, the counter increases a step, and the loop is restarted for another Lsiter iteration. Finally, θ^{best} is updated (line 22), and it is the sample point that contains the best objective function [24].

3.3 Charge and Resultant Force Calculations

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, as shown in the Eq. (17) [26]:

$$q^i = \exp\left\{-n\frac{f_{\text{error}}(\theta^i) - f_{\text{error}}(\theta^{\text{best}})}{\sum_{k=1}^{m}\left[f_{\text{error}}(\theta^k) - f_{\text{error}}(\theta^{\text{best}})\right]}\right\}, \quad \forall i \tag{17}$$

where $f_{\text{error}}(\theta^i)$ is the objective function value of sample point θ^i, and $f_{\text{error}}(\theta^{\text{best}})$ is the objective function of the current best solution. It is also worth noting that the charge of a sample point is without sign. Alternatively, the direction of a particular force between two points is specified after comparing the objective function value for each existing point [26]. Thus,

$$F^i = \sum_{j\neq i}^{m}\left\{\begin{array}{l}(\theta^j - \theta^i)\dfrac{q^i q^j}{\|\theta^j - \theta^i\|^2} \\ \text{if} \quad f_{\text{error}}(\theta^j) < f_{\text{error}}(\theta^i) \\ (\theta^i - \theta^j)\dfrac{q^i q^j}{\|\theta^j - \theta^i\|^2} \\ \text{if} \quad f_{\text{error}}(\theta^j) \geq f_{\text{error}}(\theta^i)\end{array}\right\}, \quad \forall i \tag{18}$$

where F^i is the total force exerted on sample point θ^i. A point with a superior objective function attracts other points

and vice versa. For example, Fig. 4 shows the force applied by z^1 on z^3 as a repulsion force (F_{13}) because the objective function of z^1 is worse than that of z^3. Moreover, the force F_{23} that is applied by z^2 on z^3 is regarded as an attraction force because the objective function of z^2 is better than that of z^3, which shifts z^3 along the total force F^3 [28].

3.4 Movement Along the Total Force

The final step after force evaluation is the determination of movement in accordance to the force. The particle updates itself according to the direction of the force via a random step length. The sample's point moves to the direction of the upper bounds via a random step length when the force is positive, whereas it moves toward the lower bounds when the force is negative [24] (Fig. 5, lines 7–11).

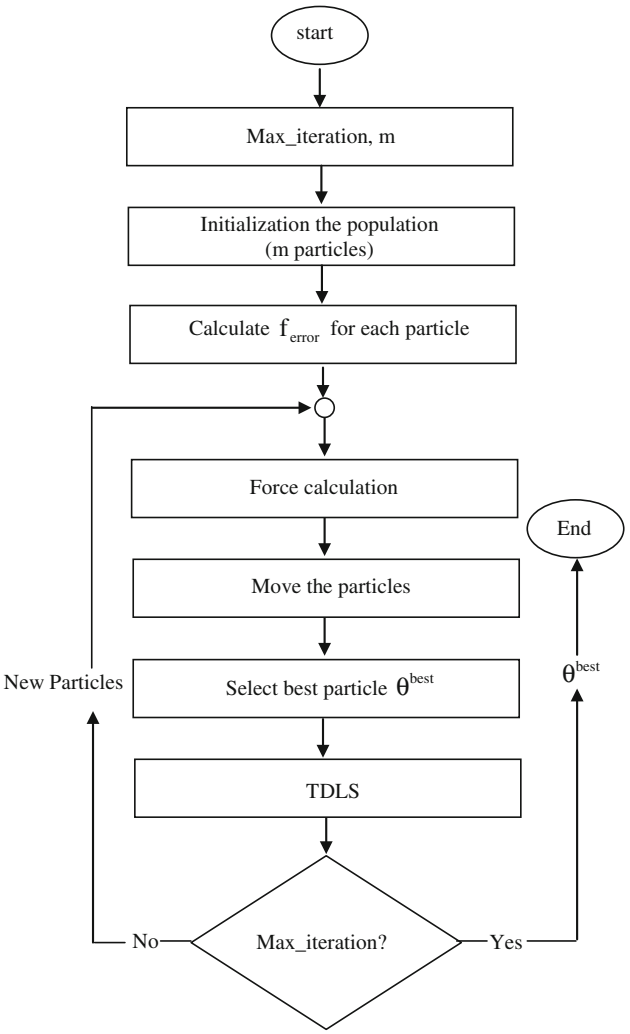


Fig. 8 Flow chart for MEMTDLS

1. $\beta \leftarrow 1/(N_i)^2$
2. **For** $i_{TDLS} = 1$ To m **do**
3. int_particle $\leftarrow 1$
4. **For** k = 1 **To** n **do**
5. temp $\leftarrow \theta^{best}$
6. $\lambda \leftarrow U(0,1)$
7. $temp_k \leftarrow temp_k + \lambda * \beta$
8. $p^{int\_particle} \leftarrow$ temp
9. temp $\leftarrow \theta^{best}$
10. $\lambda \leftarrow U(0,1)$
11. $temp_k \leftarrow temp_k - \lambda * \beta$
12. $p^{int\_particle+1} \leftarrow$ temp
13. int_particle $\leftarrow$ int_particle + 2
14. **End for**
15. $p^{best} \leftarrow \arg\min\{f(p^{int\_particle}), \forall$ int_particle$\}$
16. **If** $f(p^{best}) < f(\theta^{best})$ **then**
17. $\theta^{best} \leftarrow p^{best}$
18. **End if**
19. **End for**

Fig. 9 Pseudo code of TDLS

4 Modified EM Algorithm with Adaptive Local Search (MEMALS)

In this method, all of the parts are similar to the original EM algorithm except for the local search. The random line search in the original EM algorithm is replaced by an adaptive local search (ALS). The algorithm of the MEMALS is explained in Fig. 6.

The local search is an important step in the EM algorithm. Without it, the particles lack local information. Thus, obtaining the local optima becomes difficult [26,29]. The previous local search in the original EM is a random line search, and one of its drawbacks is its inability to evade the local optima [30]. Consequently, the ALS is one of the ways that is used to enhance the performance of the EM algorithm. In this paper, we discuss how this aim is realized by controlling the value of the parameter δ with the number of iterations as follows:

$$\delta = \frac{1}{(N_i)^2} \tag{19}$$

where the N_i is the current number of iteration. Based on Eq. (19), an increase in the number of iterations decreases the value of δ because in earlier generations, the current solutions were usually not the best, and a particular value of δ must be added to the dimensions of each particle to improve the solution. After several iterations, the value of δ gradually decreases which renders the search adjacent to the global

Fig. 10 Sample chromosome

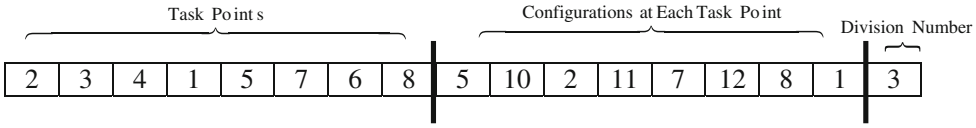


Table 1 Dual manipulator parameters

Parameters of the dual robots	Values
Length for each link	20 cm
Velocity for each joint	0.8 rad/s
Limit for the all joint except third joint	[0°, 180°]
Limit of third joint	[0°, 360°)

solution instead of being distant from the desired solution. On the other hand, this process accelerates the convergence of the algorithm compared with the local search with a fixed δ. Figure 7 illustrates the relation between the average error and the number of iterations in the context of the problem of IK for a 4-DOF planar robot and with different values of δ. The figure shows that a decrease in the value of δ enhances the accuracy of the solution but also slows the speed of convergence.

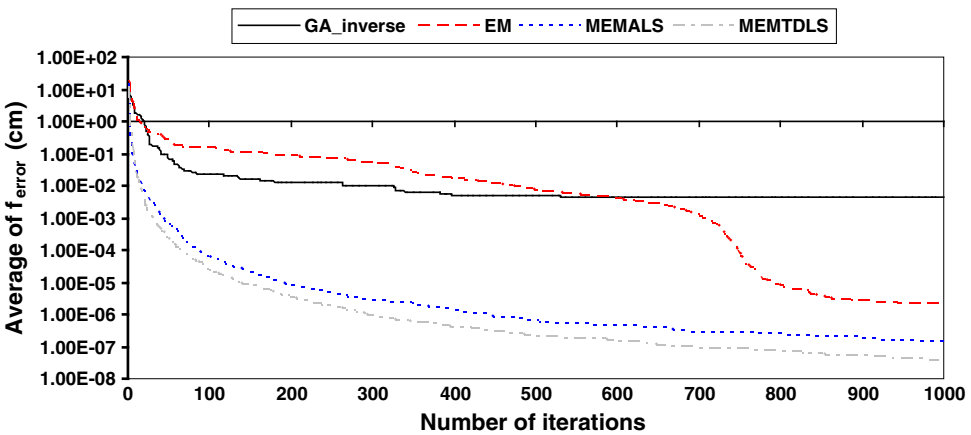
5 Modified EM Algorithm with Two-Direction Local Search (MEMTDLS)

This algorithm is a different type of modified EM algorithm, which utilizes a local search called the two-direction local search (TDLS). However, the calculations of the force and movement in this algorithm are similar to the ones in the original EM. Figure 8 displays the procedures of the MEMTDLS algorithm.

The most important part in the MEMTDLS algorithm is TDLS, which is regarded as the main contribution of this study. TDLS is proposed to improve the performance of the EM algorithm. The main advantages, which made this algorithm better than the other methods, are the fact that the algorithm directs toward the goal point in a more straightforward manner. In other words, the addition of the movement for the best current solution is a random value, but with specific directions, namely left and right. This process guides the solution toward the optimum instead of randomly choosing the directions that might divert the solution away from the global solution. Furthermore, the use of the directions of left and right for each dimension and in each i_{TDLS} iteration ensures that one of the directions is near and can still reach the global point. The TDLS method spreads of the best solution throughout different levels of configurations or solutions, which provides opportunities to generate solutions adjacent to the target point. The second point is that, this method uses the adaptive displacement that is proved in the ALS algorithm to be better than the fixed one in the enhancing the results. Moreover, the use of i_{TDLS} counter to vibrate and concentrate the search in the neighborhood of the best current solution, which is obtained after the force calculations the movement, fine tunes the best solution and provides better convergence speed.

Figure 9 shows the pseudo code for the TDLS algorithm. TDLS is dependent upon the production of two solutions for each dimension, which are obtained by adding and subtracting the adaptive displacement β. First, the value of displacement is calculated using Eq. (19) at the current iteration (Fig. 9, line 1). Procedures involving positive and negative movements for the dimensions based on the value of β are then performed to determine the intermediate points (lines 4–14). The best intermediate point is then compared with the current best solution and, if it is better, replaces the best current solution. Thus, the loop of i_{TDLS} increases (lines 15–19).

Fig. 11 The approximation of the different techniques with the iterations (simulator 1)



6 Proposed GA to Solve the Time Optimal Task Scheduling Problem

GA is capable of solving continuous, discrete and multi-objective problems [31]. The GA method incorporates and utilizes various operators, which are used as genetic operators that produce the best individuals from an initial random population [32]. These operators are imperative to the convergence of the solution. The major steps of the genetic algorithm are described below:

6.1 Initialization and Representation

A batch of the initial population is randomly generated and passed through the GA operators. At the end of each iteration, the algorithm picks the best chromosome for the next generation and randomly regenerates the remainder of the population. The representation of the chromosomes is shown in the following example:

Take as an example two 4-DOF planar robots attempting to visit eight task points in the workspace, as shown in Fig. 10.

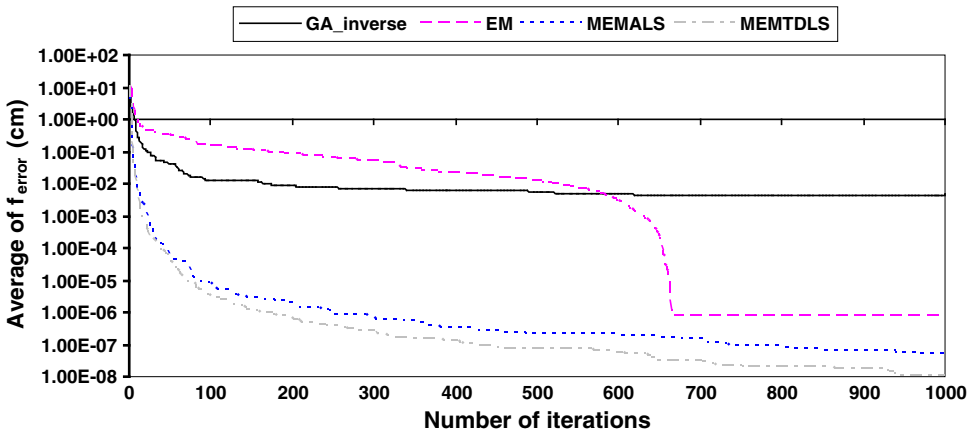


Fig. 12 The approximation of the different techniques with the iterations (simulator 2)

Table 2 The performance of different methods

Algorithm	Target position (100, 50) cm			Target position (50, 60) cm		
	Simulator 1			Simulator 2		
	Average f_{error} (cm)	SD	Best f_{error} (cm)	Average f_{error} (cm)	SD	Best f_{error} (cm)
GA_inverse	4.15E−3	2.15E−3	1.87E−3	4.65E−3	1.95E−3	1.36E−3
EM	1.95E−6	2.37E−6	4.58E−7	7.65E−7	6.28E−7	1.13E−7
MEMALS	1.40E−7	1.40E−7	2.87E−8	5.39E−8	1.98E−8	2.18E−8
MEMTDLS	3.71E−8	3.73E−8	3.15E−9	1.14E−8	4.42E−9	3.90E−9

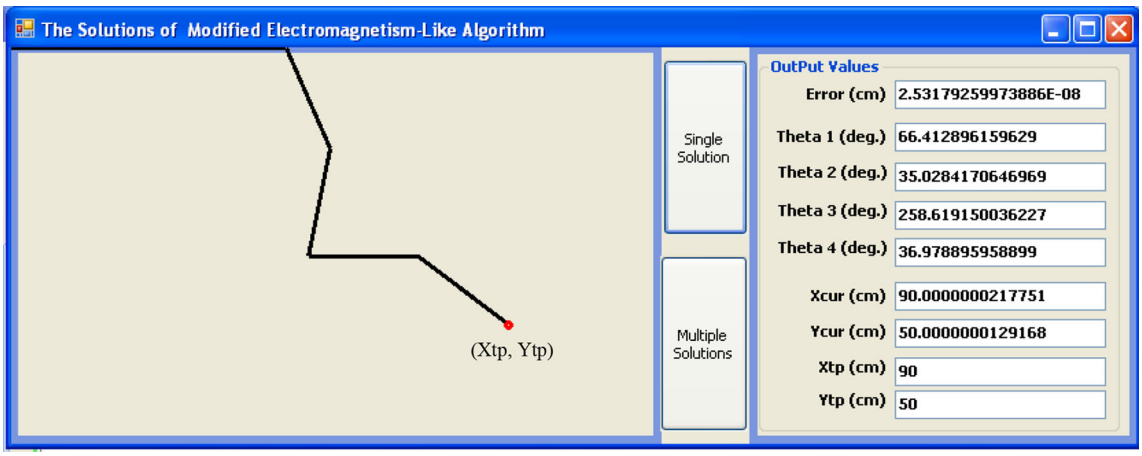


Fig. 13 Single solution example for simulator 1

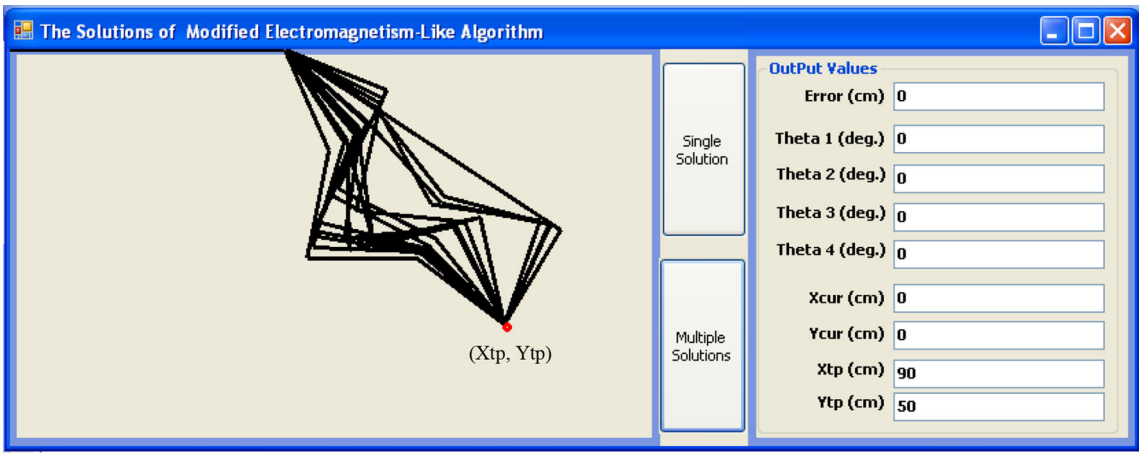


Fig. 14 Multiple robot configurations (simulator 1)

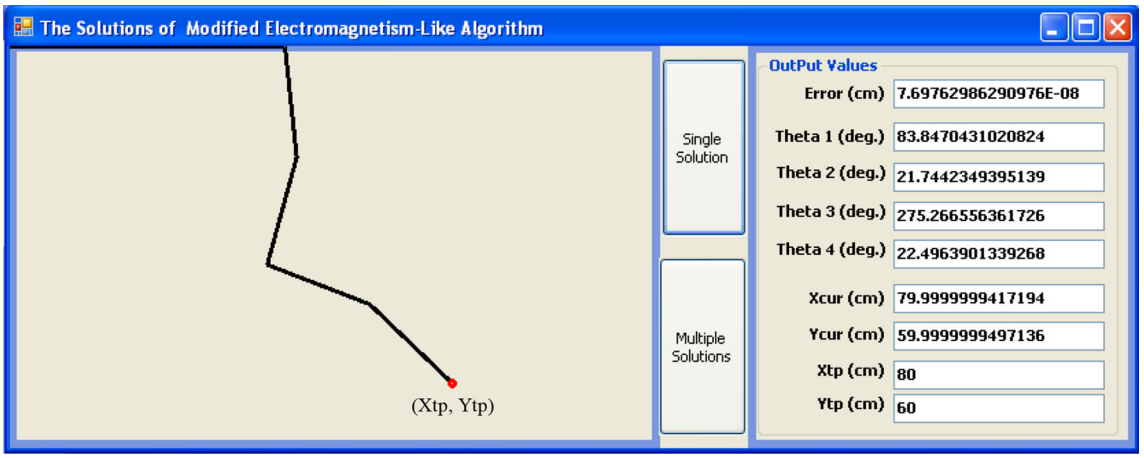


Fig. 15 Single solution example for simulator 2

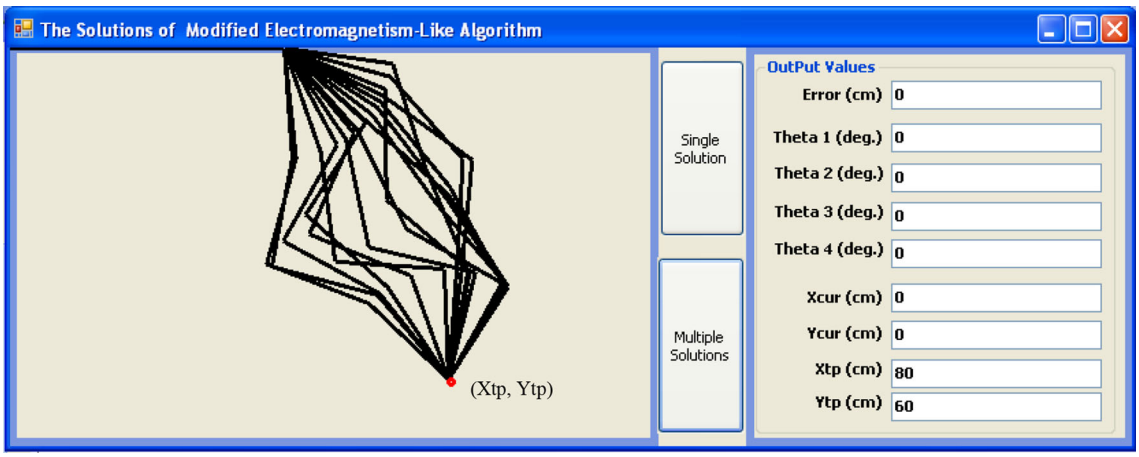


Fig. 16 Multiple robot configurations (simulator 2)

The first portion of the chromosome represents the task points, which are eight in this example. The second portion of the chromosome represents the robot solutions corresponding for every task point. For instance, if we take into account the second portion of the chromosome, 5 indicates the solution numbered 5 for task point 2. The third part of the chromosome represents the value that divides the task points between the dual manipulators. For example, the value 3 means that

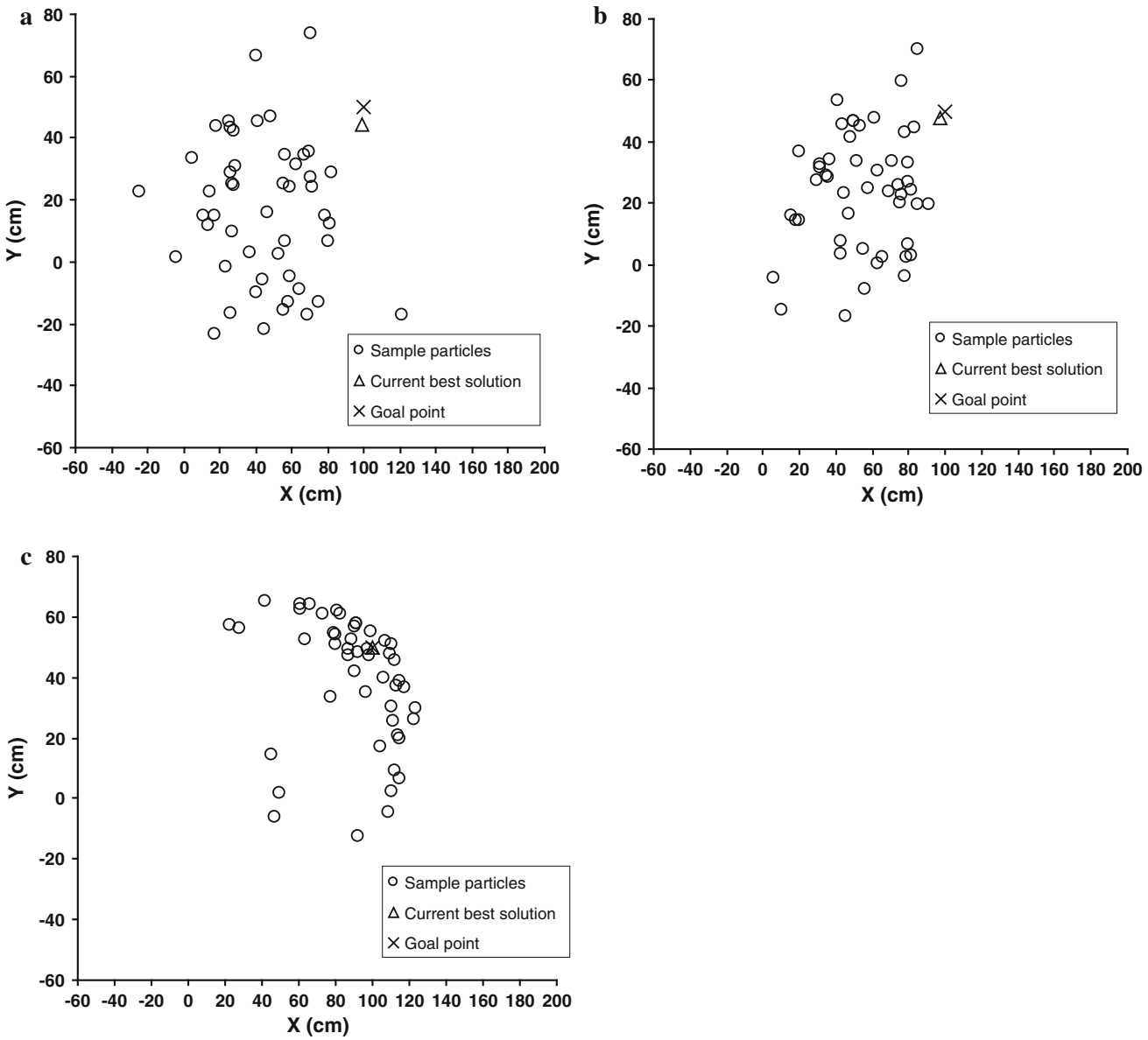


Fig. 17 Attraction–repulsion between the sample particles (simulator 1). **a** Distribution of initial particles (simulation 1). **b** Particles movement toward the target (simulator 1). **c** Current best solution is very near to the goal (simulator 1)

the first robot visits the task points {2, 3, 4}, whereas the second robot visits the task points {1, 5, 7, 6, 8}.

6.2 Evaluation

In each generation, the chromosomes are evaluated by Eq. (15) in order to measure the value of the solution [33].

6.3 Selection

This operator selects and isolates the chromosomes for use in the next generation. In this paper, the roulette wheel selection (RWS) method was used for the selection procedures.

6.4 Crossover

This reproduction operation involves two parents chosen from a pool of selected chromosomes producing an offspring.

The present study uses the order crossover method [31,34] to produce new genetic material for the first portion of the chromosome. For example, if two parent chromosomes are present, then the two selected random points divide the respective parents into a left, middle and right portion. The division results in the first child inheriting the respective left and right section from the first parent, whereas the middle section is determined by genes in the middle section of first

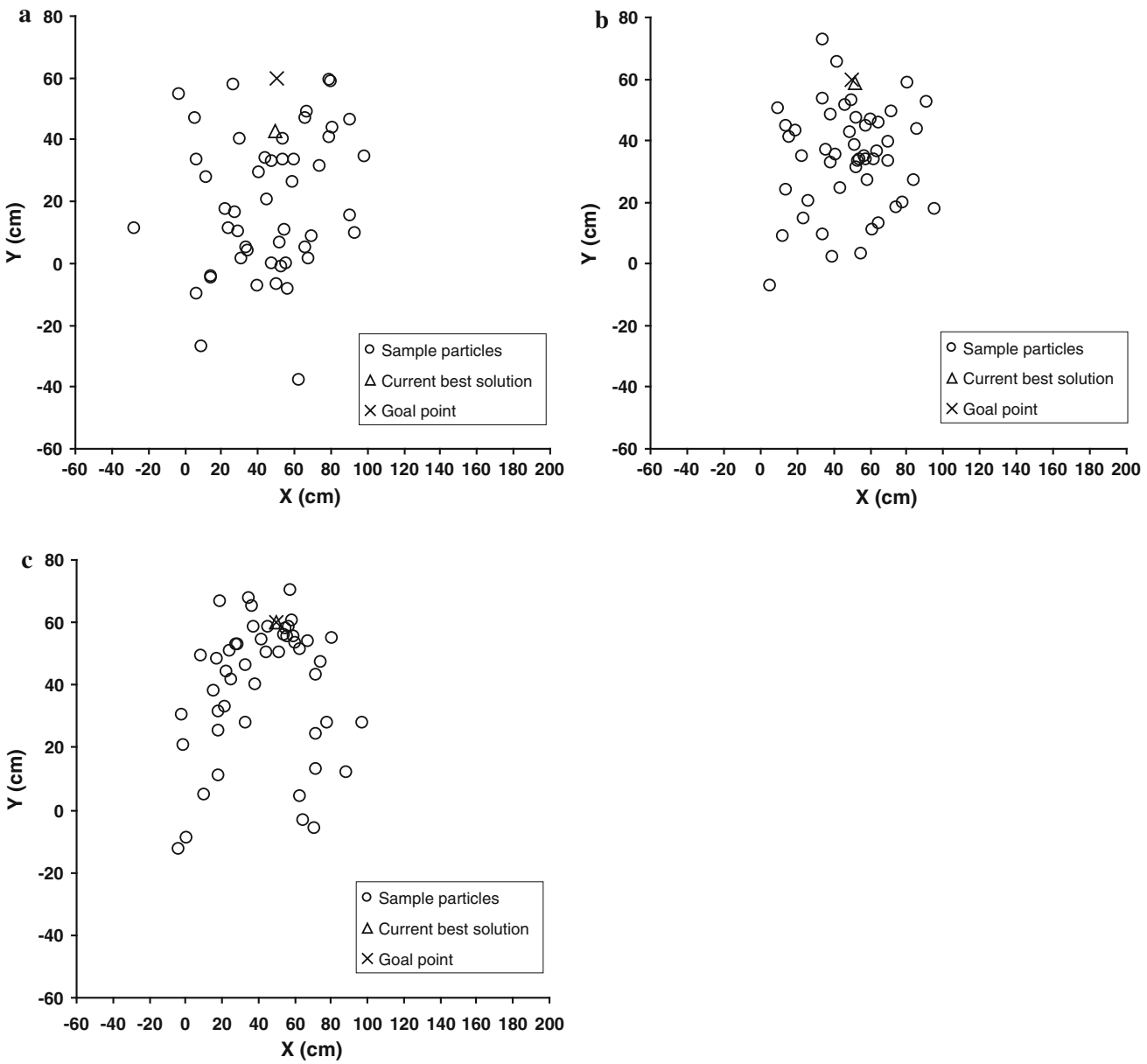


Fig. 18 Attraction–repulsion between the sample particles (simulator 2). **a** Distribution of initial particles (simulation 2). **b** Particles movement toward the target (simulator 2). **c** Current best solution is very near to the goal (simulator 2)

parent in the order in which the values appear in the second parent. Similar procedures are carried out for the second child. The second and third parts of the chromosome use the two-point crossover [31,35].

6.5 Mutation

This operation maintains the diversity of individuals during GA generations. Mutation is generally applied to avoid the local minima. The first portion of the chromosome uses interchanging mutation [31,36]. Two random positions are selected for the parent in this particular type of mutation, and the genes corresponding to these positions are interchanged. For the second part, the mutation is carried out according to the method mentioned in [37]. Where a point from the chromosome is randomly selected. After that, the gene at this point is swapped with the following gene. Boundary mutation is used in the third portion [15], where the gene changes with a random number selected from a range of allowable values.

7 Simulation Results

In all of the simulations, both robots are identical and set up at different positions. The task points and the static obstacles are

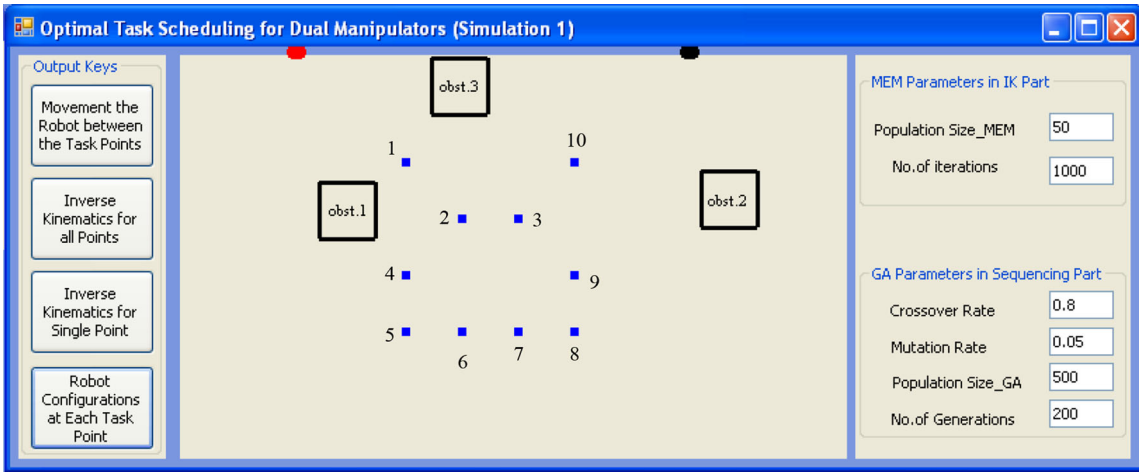


Fig. 19 Workspace (simulation 1)

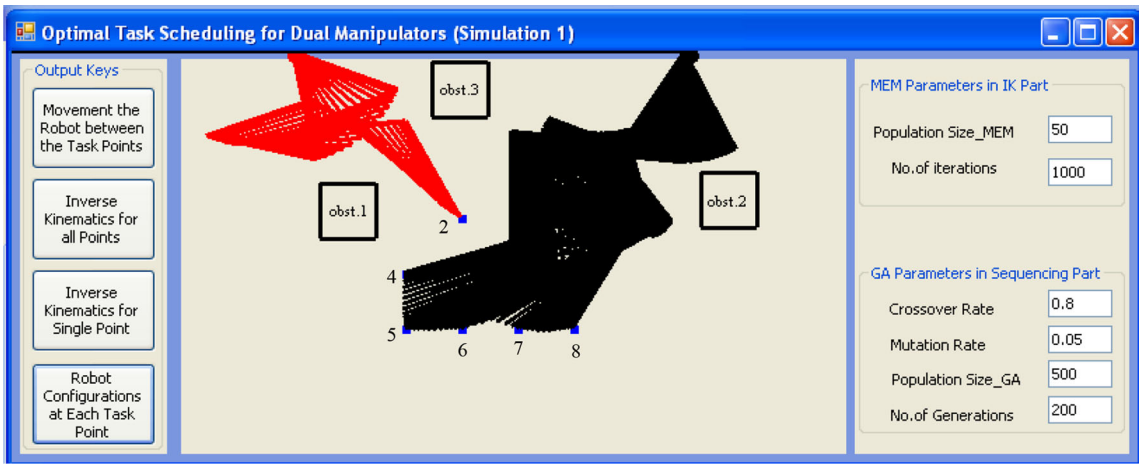


Fig. 20 Control parameters and simulator window of full movement for dual manipulators (10 points)

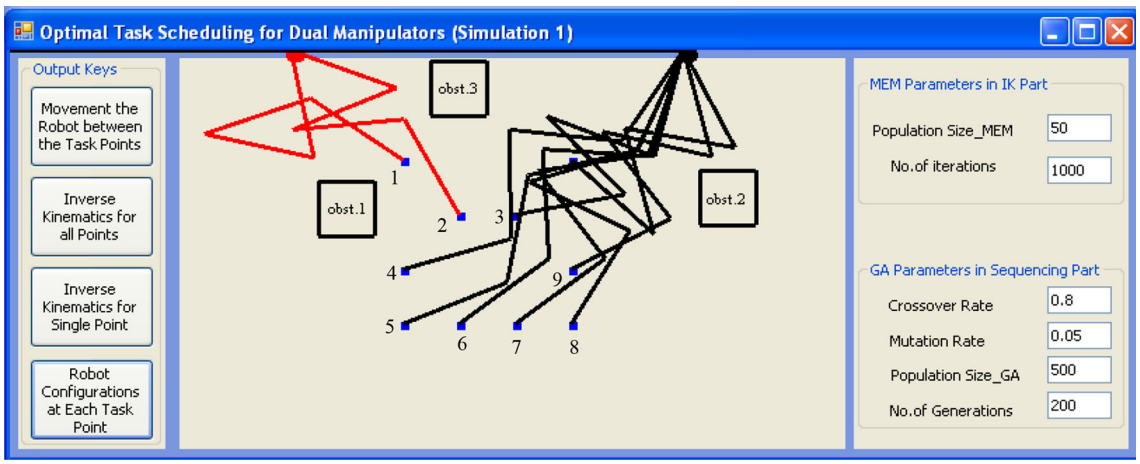


Fig. 21 Configurations of the dual manipulators for the task of simulation 1

predefined and scattered in the workspace. Two planar robots with 4 DOF were utilized to assess the proposed technique, and the parameters of the robots are shown in Table 1.

Figures 20 and 23 show the parameters of the algorithm for examples 1 and 2. Figures 11 and 12 were captured after many tests for different points and robot positions for each simula-

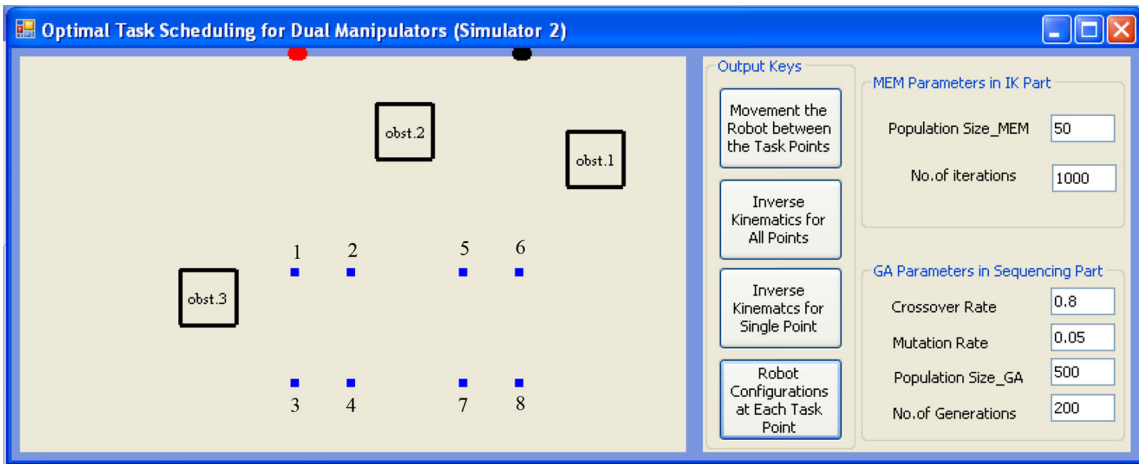


Fig. 22 Workspace (simulation 2)

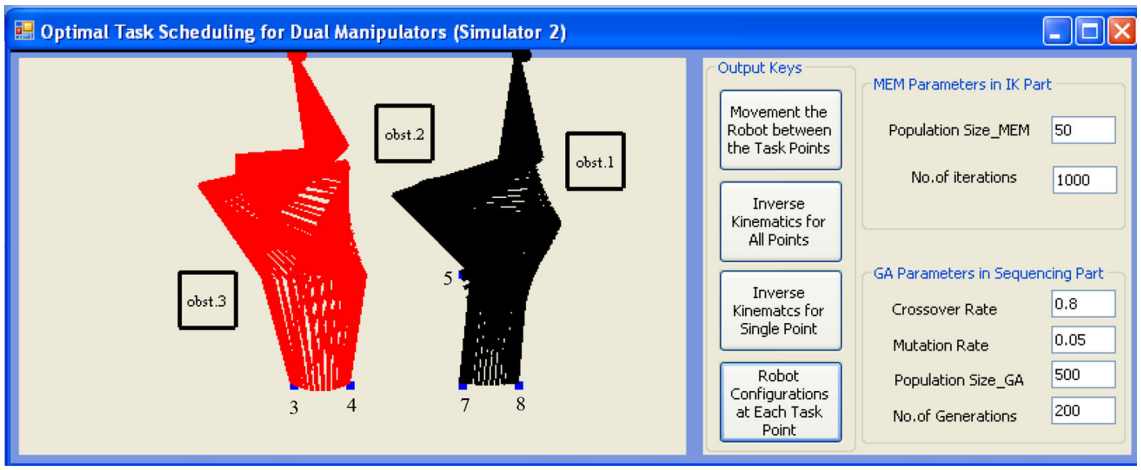


Fig. 23 Control parameters and simulator window of full movement for dual manipulators (8 points)

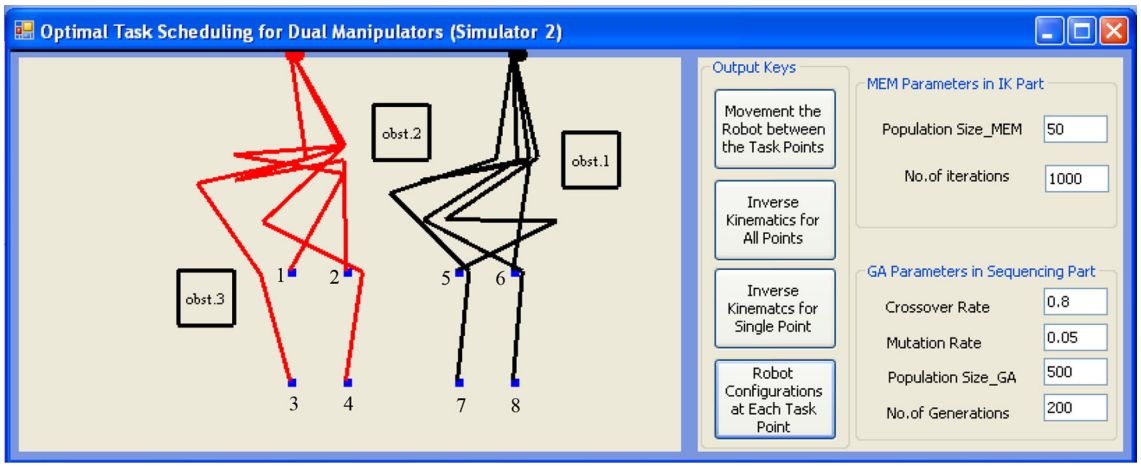


Fig. 24 Configurations of the dual manipulators for the task of simulation 2

tion. From these figures, the average iterations required for the MEMTDLS algorithm to reach the acceptable accuracy is very efficient compared with other methods; thus, it can rapidly converge to the desired position. Table 2 provides further results that illustrate the performance of the GA_inverse, EM, MEMALS and MEMTDLS for both simulators.

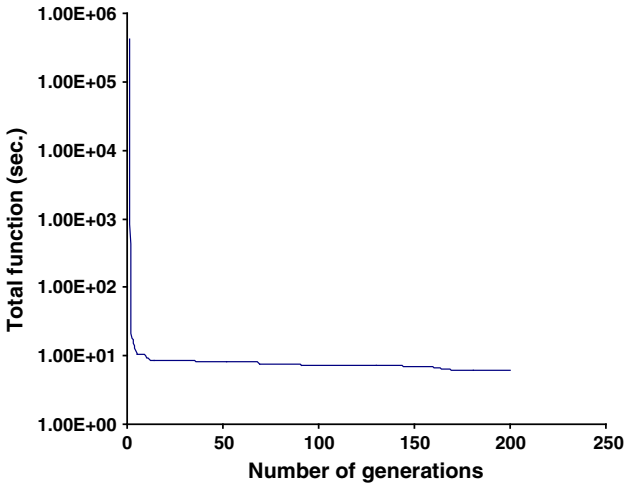


Fig. 25 The performance of GA (simulator 1)

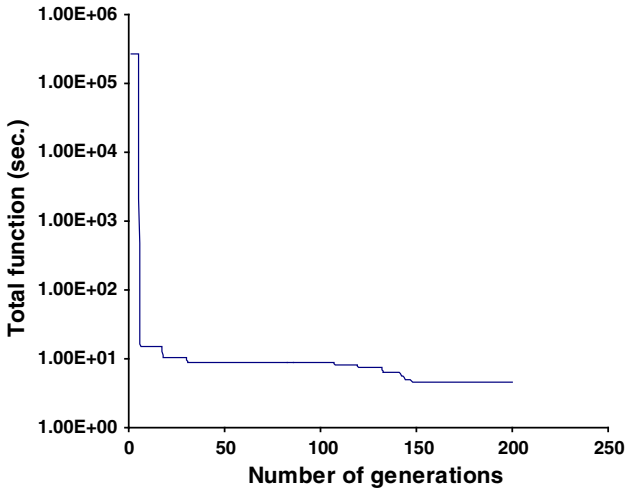


Fig. 26 The performance of GA (simulator 2)

A self-developed simulator package is designed to show the capability of the algorithm to solve efficiently the problem of IK with acceptable results and different configurations, as shown in Figs. 13, 14, 15 and 16. Figures 13 and 15 describe one solution for the given target point. Furthermore, the same figures provide the angles of each link that are calculated from the forward equations by using the MEMTDLS algorithm. The simulators also give the positional errors for the selected targets. For example, in Fig. 13, the error between the target and the best current solution is 2.5317E−8 cm. On the other hand, Figs. 14 and 16 demonstrate the ability of the algorithm to search and discover multiple robotic configurations.

The behavior of the MEMTDLS algorithm is explained in Fig. 17a–c for simulator 1, and Fig. 18a–c for simulator 2. Initially, the population is randomly generated in the workspace. After several iterations and according to the attraction–repulsion mechanism, some particles and the best current solution move toward to the desired position. Thus, in each iteration, if the new solution is better than the current best solution, this current best solution is replaced, and the process continues until a specified number of iterations in order to obtain a very close approximation of the solution to the desired point.

Figures 19, 20 and 21 show the setup of the workspace, the movement of the dual manipulators, and the configurations of the two robots at the task points for the first simulation. However, in simulation 1, the dual manipulators divide the task during the tour to obtain a near-optimal solution. Thus, 1-robot visits the task points 2–1–2, whereas 2-robot visits points 7–4–5–6–9–10–3–8–7. The total error for all of the task points is 1.1357E−7 cm. The second simulation is shown in Figs. 22, 23 and 24. In this simulation, the manipulators equally divide the tour. Hence, 1-robot visits 1–2–3–4–1, whereas 2-robot visits 8–5–6–7–8. The total error for simulator 2 is 1.1239E−7 cm. In both simulators, the dual manipulators pass the task points with near-optimal cycle time, and completely avoid collision between the manipulators and the static obstacles inside the environment. Consequently, the static obstacles in the dual manipulators environment increase the complexity of the current movement and after that the time. In these workspaces, not only does one manipulator avoid the other, but must also be shielded from static obstacles.

The optimization discussed here is with regards to the time consumption which is determined from the joint–space perspective and not travel distance. The solution might not suggest optimal path for the robot to follow, but actually requires shorter cycle time, especially when it is resulting from simultaneous operation of several actuators at the joints.

Thus, the results confirm that the suggested GA is capable of searching in a workspace containing two robots and static obstacles, where multiple solutions for each task point are taken into account to determine accurately the near-optimal solution.

Figures 25 and 26 show the total function approximation with the number of generations of the GA for simulations 1 and 2, respectively. The total function has already obtained a collision-free solution after a number of iterations, but the GA continues to enhance the solution further.

8 Conclusions

The two manipulators system has numerous advantages, which motivate researchers to enhance its performance. A method based on MEMTDLS and GA can solve the near-optimal task scheduling problem. Multiple robot configurations at each point of the task are obtained via the MEMTDLS

algorithm with excellent accuracy. The MEMTDLS algorithm is a powerful algorithm because it possesses advantages in both accuracy and speed of convergence compared with original EM algorithm. The optimal task is calculated by the proposed GA, because GA is capable of searching in a large search space. Simulators constructed using Visual Basic 2008 showed the results and the control parameters of the algorithm. All of the computations were conducted on a Celeron R CPU 2.2 GHz PC. The algorithm can find the optimal task, which is visited and interacted with the two redundant manipulators. The two robots divide the tour points with main purpose of minimizing the total cycle time without any collisions between the manipulators, or between the manipulators and the static obstacles. Future studies should explore applications of the suggested method in the workspace with static and dynamic obstacles, equipped with two or three manipulators.

References

1. Chang, C.; Chung, M.J.; Lee, B.H.: Collision avoidance of two general robot manipulators by minimum delay time. IEEE Trans. Syst. Man Cybern. **24**, 517–522 (1994)
2. Karlik, B.; Aydin, S.: An improved approach to the solution of inverse kinematics problems for robot manipulators. Eng. Appl. Artif. Intell. **13**, 159–164 (2000)
3. Köker, R.; öz, C.; Çakar, T.; Ekiz, H.: A study of neural network based inverse kinematics solution for a three-joint robot. Robot. Auton. Syst. **49**, 227–234 (2004)
4. Kalra, P.; Mahapatra, P.B.; Aggarwal, D.K.: An evolutionary approach for solving the multimodal inverse kinematics problem of industrial robots. Mech. Mach. Theory **41**, 1213–1229 (2006)
5. Feng, Y.; Nan, W.Y.; Ning, W.S.: Inverse kinematic solution for robot manipulator based on electromagnetism-like and modified DFP algorithms. Acta Autom. Sinica. **37**(1), 74–82 (2011)
6. Edan, Y.; Flash, T.; Peiper, U.M.; Shmulevich, I.; Sarig, Y.: Near-minimum-time task planning for fruit-picking robots. IEEE Trans. Robot. Autom. **7**(1), 48–56 (1991)
7. Petiot, J.-F.; Chedmail, P.; Hascoët, J.-Y.: Contribution to the scheduling of trajectories in robotics. Robot. Comput. Integr. Manuf. **14**, 237–251 (1998)
8. Azariadis, P.N.; Aspragathos, N.A.: Obstacle representation by Bump-surfaces for optimal motion-planning. Robot. Auton. Syst. **51**, 129–150 (2005)
9. Nearchou, A.C.; Aspragathos, N.A.: A genetic path planning algorithm for redundant articulated robots. Robotica **15**, 213–224 (1997)
10. Rana, A.S.; Zalzala, A.M.S.: Collision-free motion planning of multiarm robots using evolutionary algorithms. In: Proceedings of the Institution of Mechanical Engineers. Part I: Journal of Systems and Control Engineering, pp. 373–384 (1997)
11. Zacharia, P.Th.; Aspragathos, N.A.: Optimal robot task scheduling based on genetic algorithms. Robot. Comput. Integr. Manuf. **21**, 67–79 (2005)
12. Xidias, E.K.; Zacharia, P.Th.; Aspragathos, N.A.: Time-optimal task scheduling for articulated manipulators in environments cluttered with obstacles. Robotica **28**, 427–440 (2010)
13. Jiang, K.; Seneviratne, L.D.; Earles, S.W.E.: Assembly scheduling for an integrated two-robot workcell. Robot. Comput. Integr. Manuf. **13**(2), 131–143 (1997)
14. Samanlioglu, F.; Ferrell, W.G.; Kurz, M.E.: A memetic random-key genetic algorithm for a symmetric multi-objective traveling salesman problem. Comput. Ind. Eng. **55**, 439–449 (2008)
15. Xidias, E.K.; Zacharia, P.Th.; Aspragathos, N.A.: Time-optimal task scheduling for two robotic manipulators operating in a three-dimensional environment. Proc. IMechE Part I J. Syst. Control Eng. **224**, 845–855 (2010)
16. Yahya, S.; Moghavvemi, M.; Mohamed, H.A.F.: Geometrical approach of planar hyper-redundant manipulators: inverse kinematics, path planning and workspace. Simulat. Model. Pract. Theory **19**, 406–422 (2011)
17. Yao, Z.; Gupta, K.: Path planning with general end-effector constraints. Robot. Auton. Syst. **55**, 316–327 (2007)
18. Abed, I.A.; Koh, S.P.; Sahari, K.S.M.; Tiong, S.K.; Yap, D.F.W.: Comparison between genetic algorithm and electromagnetism-like algorithm for solving inverse kinematics. World Appl. Sci. J. **20**(7), 946–954 (2012)
19. Selig, J.M.: Introductory Robotics. Prentice Hall, Englewood Cliffs, NJ (1992)
20. Ata, A.A.; Myo, T.R.: Optimal point-to-point trajectory tracking of redundant manipulators using generalized pattern search. Int. J. Adv. Robot. Syst. **2**(3), 239–244 (2005)
21. Lai, K.-C.; Kang, S.-C.: Collision detection strategies for virtual construction simulation. Autom. Construct. **18**, 724–736 (2009)
22. Khalili, M.; Moghaddam, R.T.: A multi-objective electromagnetism algorithm for a bi-objective flowshop scheduling problem. J. Manuf. Syst. **31**, 232–239 (2012)
23. Yurtkuran, A.; Emel, E.: A new hybrid electromagnetism-like algorithm for capacited vechicle routing problems. Exp. Syst. Appl. **37**, 3427–3433 (2010)
24. Jhang, J.-Y.; Lee, K.-C.: Array pattern optimization using electromagnetism-like algorithm. Int. J. Electron. Commun. (AEÜ). **63**, 491–496 (2009)
25. Lee, C.-H.; Chang, F.-K.; Lee, Y.-C.: An improved electromagnetism-like algorithm for recurrent neural fuzzy controller design. Int. J. Fuzzy Syst. **12**, 280–290 (2010)
26. Birbil, Ş.İ.; Fang, S.-C.: An electromagnetism-like mechanism for global optimization. J. Global Optimiz. **25**, 263–282 (2003)
27. Chang, P.-C.; Chen, S.-H.; Fan, C.-Y.: A hybrid electromagnetism-like algorithm for single machine scheduling problem. Exp. Syst. Appl. **36**, 1259–1267 (2009)
28. Debels, D.; Reyck, B.D.; Leus, R.; Vanhoucke, M.: A hybrid scatter search/electromagnetism meta-heuristic for project scheduling. Eur. J. Oper. Res. **169**, 638–653 (2006)
29. Guan, X.; Dai, X.; Qiu, B.; Li, J.: A revised electromagnetism-like mechanism for layout design of reconfigurable manufacturing system. Comput. Ind. Eng. **63**, 98–108 (2012)
30. Lin, J.-L.; Wu, C.-H.; Chung, H.-Y.: Performance comparison of electromagnetism-like algorithms for global optimization. Appl. Math. **3**, 1265–1275 (2012)
31. Sivanandam, S.N.; Deepa, S.N.: Introduction to Genetic Algorithms. Springer, Berlin (2008)
32. Ganesan, H.; Mohankumar, G.: Optimization of machining techniques in CNC turing centre using genetic algorithm. Arab. J. Sci. Eng. **38**, 1529–1538 (2013)
33. Lee, M.: Evolution of behaviors in autonomous robot using artificial neural network and genetic algorithm. Inf. Sci. **155**, 43–60 (2003)
34. Carter, A.E.; Ragsdale, C.T.: A new approach to solving the multiple traveling salesperson problem using genetic algorithms. Eur. J. Oper. Res. **175**, 246–257 (2006)
35. Chen, S.-M.; Chien, C.-Y.: Solving the traveling salesman problem based on the genetic simulated annealing ant colony sys-

tem with particle swarm optimization techniques. Exp. Syst. Appl. **38**, 14439–14450 (2011)
36. Louis, S.J.; Li, C.: Case injected genetic algorithms for traveling salesman problems. Inf. Sci. **122**, 201–225 (2000)
37. Asadzadeh, L.; Zamanifar, K.: An agent-based parallel approach for the job shop scheduling problem with genetic algorithms. Math. Comput. Model. **52**, 1957–1965 (2010)