The International Journal of Advanced Manufacturing Technology (2021) 115:927–939
https://doi.org/10.1007/s00170-020-06166-0

ORIGINAL ARTICLE

Optimal robot task scheduling based on adaptive neuro-fuzzy system and genetic algorithms

E. Xidias [1] · V. Moulianitis [1] · P. Azariadis [1]

Received: 19 August 2020 / Accepted: 24 September 2020 / Published online: 12 October 2020

Abstract
Industrial manipulators should be able to execute difficult tasks in the minimum cycle time in order to increase performance in a robotic work cell. This paper is focused on determining the near optimum route of a manipulator's end-effector which is requested to reach a predefined set of demand points in a robotic work cell. Two subproblems are related with this goal: the motion planning problem and the task scheduling problem. A new approach is presented in this paper for simultaneously planning collision-free motion and scheduling time near optimum route along the demand points. A combination of a geometrical approach and an adaptive neuro-fuzzy system is employed to consider the multiple manipulator's configurations, while a special genetic algorithm is designed to solve the derived optimization problem. The experiments show that the proposed method has the capacity to determine both the near optimum manipulator configurations and the near optimum sequence of demand points.

Keywords Task scheduling · Manipulator · Adaptive neuro-fuzzy system · Genetic algorithms · Collision avoidance · Robotic work cell

1 Introduction

Nowadays, robotics and more specifically industrial manipulators play an important role in Industry 4.0. Industrial manipulators have several features such as flexibility and adaptability which make them attractive for various assignments within large industrial environments. Moreover, they must be able to perform their tasks as fast as possible in order to keep productivity in a high level and simultaneously to decrease production costs [1, 2]. Thus, one of the major topics of the robotics research community is to develop methodologies that enable the manipulators to perform the requested tasks, such as spot welding, as quickly as possible, considering the limits imposed by (a) their characteristics and (b) the working environment [3].

The presented work is motivated by industrial applications where a manipulator is requested to visit a set of demand points with no predefined order. Some examples of such applications are the laser cutting, the multiple drilling and the spot welding [4]. There are numerous industrial activities where the manipulator is requested to reach several demand points and return to the initial demand point. Clearly, in these problems, the sequence of demand points has a strong effect on the total cycle time.

The problem of optimal multi-tasking motion planning for an industrial manipulator which is requested to operate in a robotic work cell can be considered the coupling of two NP-hard problems: (a) the motion planning problem (MPP) [5] and (b) the task scheduling problem (TSP) [6]. Several approaches have been implemented for the solution of TSP as well as for motion planning (MP) of manipulators. However, only few researchers have worked on the solution of the combined problem (MPP and TSP) for manipulators operating in robotic work cells.

Xidias et al. [7] investigate the combinatorial problem of MPP and TSP for an articulated robot which is operating in a 2D robotic work cell. The objective is to obtain the shortest cycle time by considering the multiple solutions of the inverse

✉ E. Xidias
xidias@aegean.gr

V. Moulianitis
moulianitis@aegean.gr

P. Azariadis
azar@aegean.gr

[1] Department of Product & Systems Design Engineering, University of the Aegean, Ermoupolis, Greece

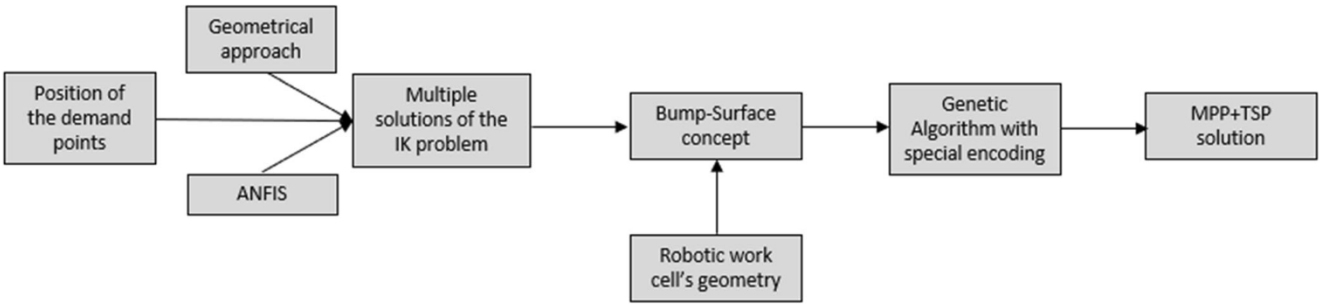

Fig. 1 A schematic representation of the proposed approach

kinematic (IK) problem and robotic work cell's structure. The proposed concept adapts the genetic algorithms (GA), proposed by Zacharia et al. [4], and the bump-surface concept [8] for the collision avoidance process. However, the method cannot guarantee that the robot's end-effector will reach the required task point. Then, Zacharia et al. [9] extend the previous approach by adapting it to the 3D robotic work cell for a PUMA manipulator. The derived optimization problem was solved by using a GA. The main advantage of the proposed approach is that task scheduling and motion planning are incorporated into the objective function. However, the length and complexity of the chromosome used in the proposed GA make the underlying method very time-consuming.

Huang et al. [10] study the problem of selecting a specific manipulator for the particular multi-goal task planning in a manipulator system. The input parameters were (a) the number and the position of demand points and (b) the specifications of the candidate manipulator such as the Denavit-Hartenberg (D-H) parameters and the joint limitations. The output was the selection of an appropriate manipulator system.

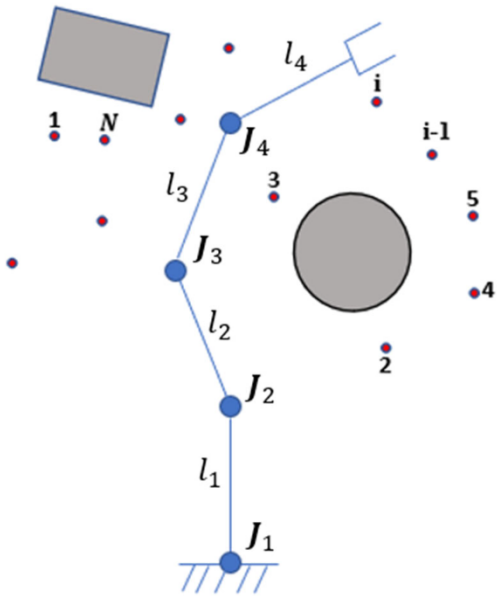

Fig. 2 A schematic representation of the 2D robotic work cell with N demand points denoted with small circles and two obstacles

With this approach, it is not always feasible to obtain an optimal solution due to the complexity of the problem.

Lattanzi et al. [11] propose an approach which was motivated by a visual inspection operation in a robotic work cell. The objective was to determine an efficient and safe path for the manipulator in order to visit a predefined number of demand points. The authors claim that the total execution time of the robot task depends on the length and complexity of the path.

Baizid et al. [12] propose an approach which is based on GAs by exploiting CAD capabilities to simulate and optimize the cycle time in performing manufacturing operations. The objective is to determine the shortest travel distance of a 6-DoF manipulator by considering the multiple solutions of the IK problem. However, they did not consider the work cell's structure. In [13], there is an extensive review of the combinatorial problem of MP and TSP.

In this paper, we consider a combined motion planning and task scheduling problem by considering simultaneously (a) the entire manipulator's work cell, (b) the location of the demand points, (c) the multiple solutions of the IK problem at each demand point and (d) the manipulator's structure. First, by using an adaptive neuro-fuzzy system (ANFIS) and a geometrical approach, we solve the IK problem at each demand point. Then, we generate a safe motion between each pair of demand points by using the flexibility of the bump-surface concept. Finally, the overall problem is solved by using a modified GA with special encoding in order to derive a near

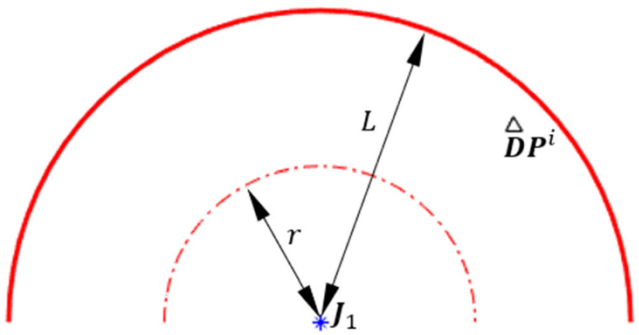

Fig. 3 The robotic work cell with the circle (J_1, r) denoted with a dashed line

optimum solution. The architecture of the proposed approach is presented in Fig. 1.

The main innovations and contributions of the proposed approach are the following:

- The representation of the robot's workspace by using the bump-surface concept is fast and independent from the manipulator's number of DoFs. Thus, there is no need for high storage capacity.
- Any shape of the robot can be included. The collision checking procedure is not combinatorial.
- The combination of ANFIS system and a geometrical approach allows us to solve the IK for a given task point in seconds.
- In contrast with [7, 9], the proposed approach takes into account the desired orientation at a task point. Furthermore, the proposed GA uses a chromosome which its length is much shorter from those on [7, 9], which helps the GA's convergence.

The rest of the paper is organized as follows. In Section 2, the problem definition and notations are described. Section 3 presents the proposed optimization algorithm, while in Section 4 the experiments are presented. Section 5 summarizes the contribution of the paper.

2 Problem statement, assumptions and notations

2.1 Problem statement

In a work cell, the time required by an industrial manipulator to carry out a requested operation (cycle time) depends on (a) the morphological characteristics of the manipulator, (b) the route of the end-effector, (c) the travel distance of the end-effector, (d) the inverse kinematic solution at each demand point and (e) the structure of the robotic work cell. The approach proposed in this paper differs from the works reported in [4], since we are considering all the above issues simultaneously.

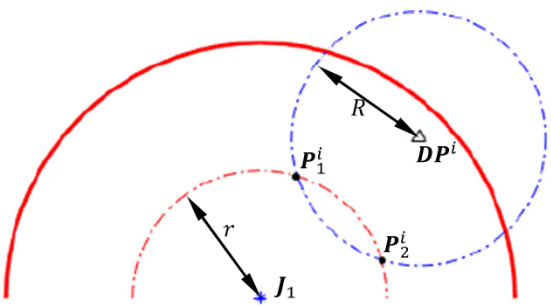

Fig. 4 The intersection of circles (DP^i, R) and (J_1, r)

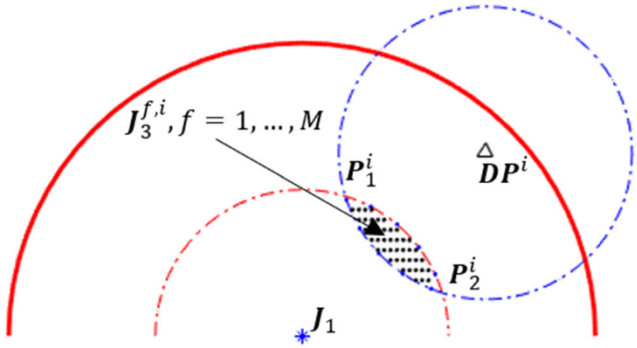

Fig. 5 The possible positions of $J_3^{f,i}$ (black dots)

Let a planar 4-link manipulator with rotational joints, having $l = \{l_1, l_2, l_3, l_4\}$ as link lengths and $\theta^i = \{\theta_1^i, \theta_2^i, \theta_3^i, \theta_4^i\}$, $\theta_1^i \in [0, \pi], \theta_j^i \in [-\pi, \pi], j = 2, \ldots, 4$ as joint angles. The Cartesian coordinates of the end-effector (x_E^i, y_E^i) at i demand point are given by the forward kinematic equations:

$$\begin{cases} x_E^i = x_B + \sum_{j=1}^{4} l_j \cos\left(\sum_{j=1}^{4} \theta_j^i\right) \\ y_E^i = y_B + \sum_{j=1}^{4} l_j \sin\left(\sum_{j=1}^{4} \theta_j^i\right) \end{cases}, i = 1, \ldots, N \quad (1)$$

where (x_B, y_B) defines the manipulator's base position.

The manipulator is requested to visit N demand points $(x_E^i, y_E^i), i = 1, \ldots, N$ in a robotic work cell which is cluttered with obstacles (see Fig. 2). Furthermore, we consider that:

- The robotic work cell geometry is known.
- The location of all demand points (x_E^i, y_E^i) is known and they should be visited by the manipulator only once.

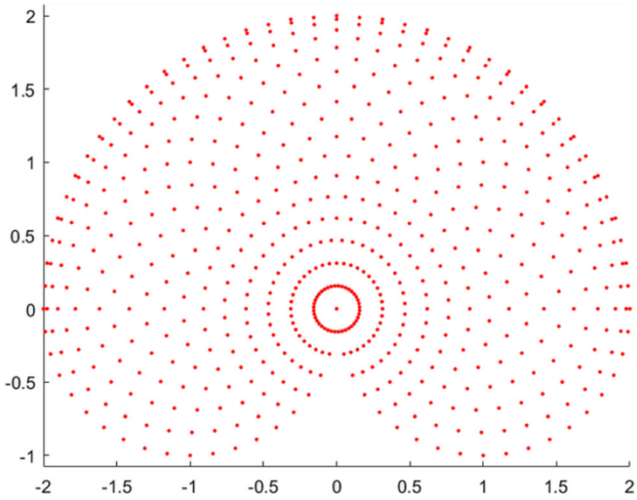

Fig. 6 The workspace for the configuration of the 2-link manipulator

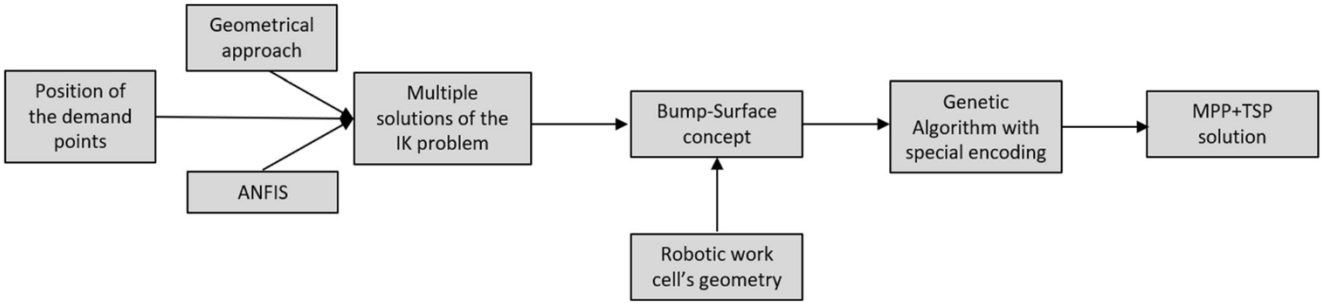

Fig. 7 ANFIS flowchart

The overall problem is formulated by the combination of the following problems:

A. The manipulator should avoid colliding with obstacles while moving in the robotic work cell.
B. The end-effector should reach all the given demand points, each one exactly once, and return to the initial demand point in the minimum cycle time.

- It is necessary to be pointed that at each demand point (x_E^i, y_E^i), we consider all possible configurations of the manipulator.

2.2 The inverse kinematics of a 4-link manipulator

It is computationally difficult to solve the presented problem as it leads to a large and multidimensional search space. Thus, we propose a combination of ANFIS and a geometric approach for the solution of the IK problem for a 4-link planar manipulator with rotational joints. Its goal is to perform a fast-local search in order to determine the position of the 3rd joint instead of applying slow global optimization techniques to the whole problem. The proposed approach can be explained in the following steps:

Let DP^i denote the i demand point (x_E^i, y_E^i). In order to determine the manipulator's configuration at DP^i, we first define the position J_3 of the 3rd joint as follows:

- Let the first joint J_1 be located at the base of the manipulator.
- Let a circle of radius $r = l_1 + l_2$ with its centre at point J_1, while L is the maximum working area defined by $L = \sum_{j=1}^{4} l_j$, as shown in Fig. 3.

- Let a circle with radius $R = l_3 + l_4$ and centre at DP^i. We have three cases:

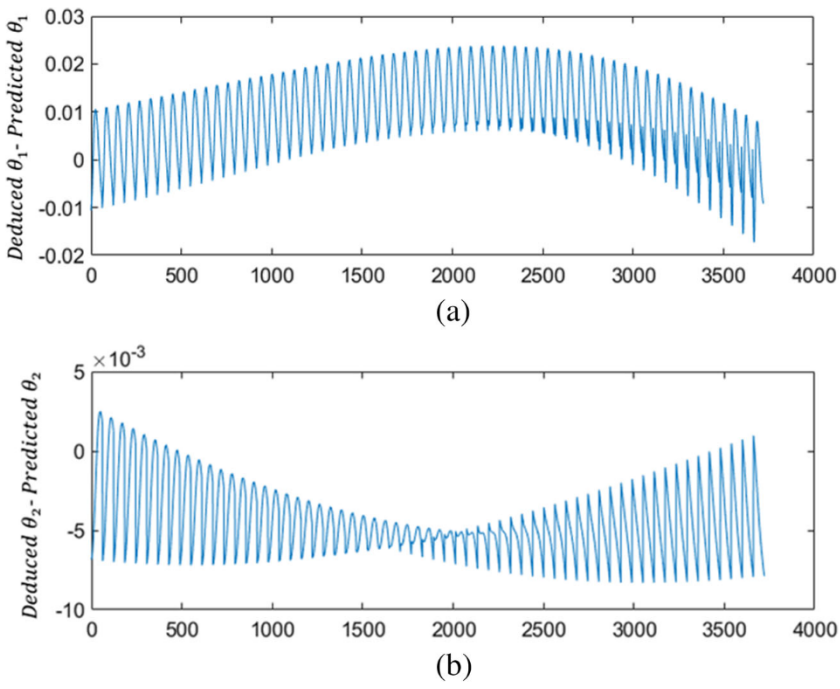

Fig. 8 **a** The error range (in rad) between the deduced θ_1 and the predicted θ_1. **b** The error range (in rad) between the deduced θ_2 and the predicted θ_2

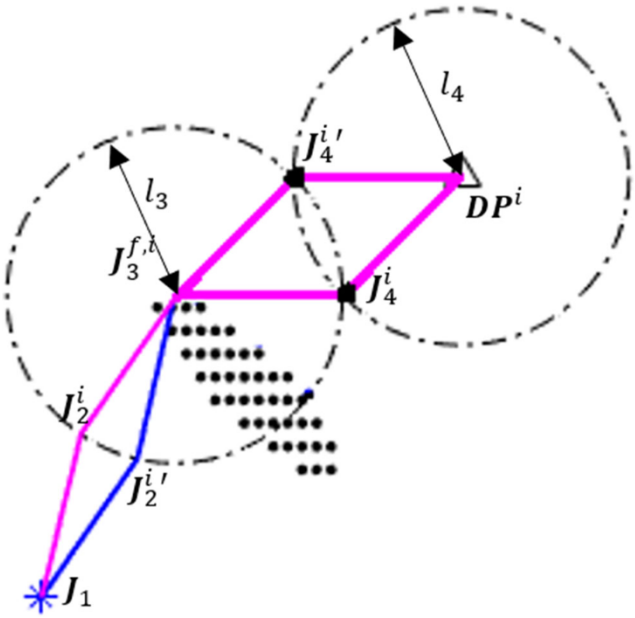

Fig. 9 The four solutions of the IK problem for a given position of the 3rd joint

- If $d(J_1, DP^i) = R + r$, then the two circles (J_1, r) and (DP^i, R) touch each other externally. The touching point defines the position of the 3rd joint J_3^i.
- If $d(J_1, DP^i) > R + r$, then the point DP^i is outside of the manipulator's working area.
- If $d(J_1, DP^i) < R + r$, then the circle (DP^i, R) intersects the circle (J_1, r) at two points P_1^i and P_2^i, as shown in Fig. 4.

- For the case where $d(J_1, DP^i) < R + r$, the possible positions $J_3^{f,i}, f = 1, \ldots, M, i = 1, \ldots, N$ of the 3rd joint are located at the intersection area of two circle disks (J_1, r) and (DP^i, R), as shown in Fig. 5. It should be noted that the number M of points J_3^f depends on the desired dexterity of the manipulator and it is set by the user. After extensive experimentation, we set $M = 10$ random points in the intersection area.

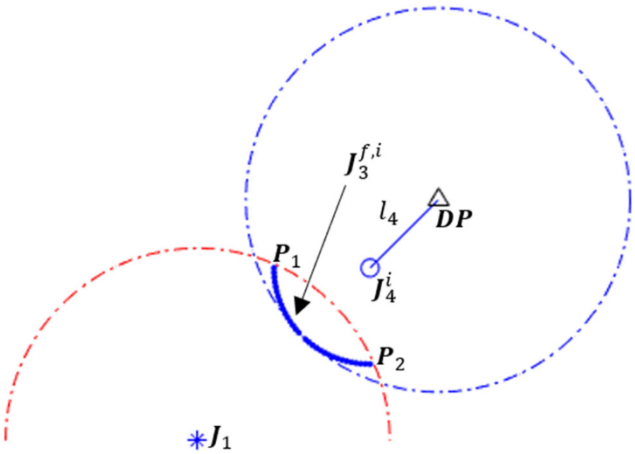

Fig. 10 The locus of points of the possible position of the 3rd joint (thin blue dots), for a given position and orientation of the end-effector

Then, in order to find the multiple manipulator's configurations, we follow the procedure, which is described below;

First, using the ability of ANFIS [14] to learn from training data, we create two systems (one for each solution of the IK) to solve the IK problem of a 2-link planar manipulator with rotational joints. Generally, ANFIS is a neuro-fuzzy technique that combines the learning capabilities of neural networks with fuzzy inference systems. In MATLAB Fuzzy Logic Toolbox, this is achieved through a learning process that combines the least squares method with the backpropagation gradient descent method. The learning algorithm tunes the membership functions of a Sugeno-type fuzzy inference system using the available training data.

In this paper, the coordinates of the manipulator's end-effector (which actually are the coordinates of the 3rd joint for a 4-link manipulator) are the inputs to the two ANFIS and the angles (θ_1, θ_2) are the outputs. The training data of each ANFIS system are determined by two disjoint sets (aspects) using the Jacobian determinant. For a 2R geometry, each aspect contains one of the two solutions of the inverse kinematics. The Jacobian matrix of the two planar manipulator is:

$$J = \begin{bmatrix} -l_1 \sin\theta_1 & -l_2 \sin(\theta_1 + \theta_2) \\ l_1 \cos\theta_1 & -l_2 \cos(\theta_1 + \theta_2) \end{bmatrix} \tag{2}$$

The determinant of the 2-link planar manipulator is $|J| = l_1 l_2 \sin\theta_2$ and does not depend on the first joint variable. The configuration variable that affect the singularity locus depends only on θ_2, and the singularity locus that defines the aspects is determined by $\theta_2 = 0$. Therefore, for the training of the 1st ANFIS, we consider the joint angle constraints $\theta_1 \in [0, \pi]$ and $\theta_2 \in [0, \pi]$, while for the training of the 2nd ANFIS, we consider $\theta_1 \in [0, \pi]$ and $\theta_2 \in [-\pi, 0]$. For both ANFIS, we consider that $l_1 = l_2 = 1$. The end-effector's coordinates are calculated for two joints using forward kinematics. Figure 6 shows the generated workspace. Then, the duplicates in the input data are identified, and the corresponding training set is removed. This way, each configuration that ANFIS learns refers to a unique mapping from Cartesian space to joint space.

Each of the ANFIS uses seven membership functions of a Sugeno-type for each of the two input variables (the (x, y) end-effector coordinates) meaning a rule base made of $7^2 = 49$ rules is generated for each ANFIS. The flowchart of the ANFIS procedure is shown in Fig. 7.

The proposed ANFIS structure is the result of several training scenarios with different configurations of the FIS. These scenarios considered (for each ANFIS) combinations of 4, 5,

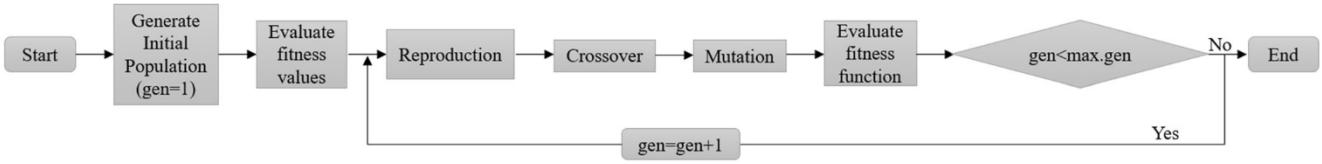

Fig. 11 The flowchart of the developed genetic algorithm

6, 7, 8 membership functions per input, training set ranging from 100 to 1000 input/output with step 50 data samples and up to 150 training epochs.

After training, an important follow-up step is to validate the resulted ANFIS. In our case, with a 2-link planar manipulator with rotational joints, we have tested the ANFIS output (predicted theta) with the results of the closed form inverse kinematics solution. The training of ANFIS was done by using 80% of total data set. The remaining 20% of data set was used for testing and validating the performance of ANFIS. Figure 8 shows for the 1st ANFIS the error range between the deduced and predicted values for θ_1 (Fig. 8a) and θ_2 (Fig. 8b). In both cases, the error range is under 3% which is accepted for the given application. Each ANFIS is trained using a training set of 1000 input/output data samples for 150 epochs.

Then, for a given position of the 3rd joint $J_3^{f,i} = \left(x_3^{f,i}, y_3^{f,i}\right), f = 1, \ldots, M, i = 1, \ldots, N$ and a given demand point $DP^i = \left(x_E^i, y_E^i\right)$, we determine the possible positions $J_4^{q,i}, q = 1, \ldots, 2M$ (two solutions J_4^i and $J_4^i\prime$ for each $J_3^{f,i}$) of the 4th joint by finding the intersection points of the two circles $\left(J_3^{f,i}, l_3\right)$ and (DP^i, l_4). Figure 9 shows a schematic representation of the manipulator's configurations. The positions of the points J_2^i and $J_2^i\prime$ are determined by the generated ANFIS. Furthermore, it should be mentioned that, for a given point DP^i, we have $4 * M$ solutions of the IK problem.

For a requested orientation of the manipulator's end-effector, the manipulator's configurations are reduced significantly: the position of the 4th joint is predetermined and the points $J_3^{f,i}$ are located within the intersection area of three circles (J_1, r), (DP^i, R) and $\left(J_4^i, l_3\right)$; see the blue arc in Fig. 10.

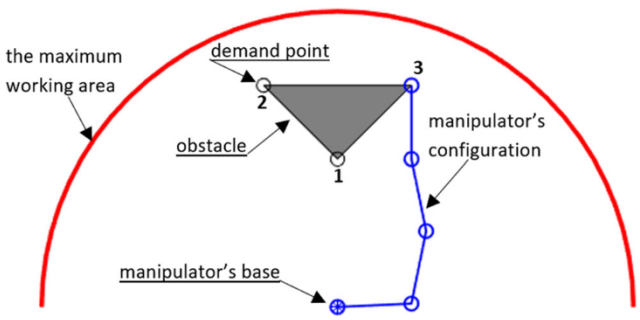

Fig. 12 An example of a robotic work cell with 3 demand points and an initial configuration

2.3 Motion planning

Following the results from [9], in order to provide a safe motion for the manipulator, we used the bump-surface concept [8]. The bump-surface S can represent all feasible configurations through a set of one-parametric curves $C(t)$ which are lying on S. A safe motion for the manipulator should be searched in the *flat* regions of S. The *flat* image of the manipulator's configuration on S is given by the following integral:

$$flat = \int_0^1 S_z(C(t))dt \tag{3}$$

where $S_z(C(t))$ denotes the third coordinate of the image $C(t)$ on S.

2.4 Computing the cycle time

The cycle/travel time t_{cycle} which is needed to reach the N demand points, as well as the intermediate configurations corresponding to points which are used to assist the collision avoidance manoeuvre in the robotic work cell, is given by [9]:

$$t_{\text{cycle}} = \sum_{i=1}^{N+1} \left(t_A^i + t_B^i + t_C^i\right) \tag{4}$$

where

- t_A^i is the travel time from the $(i-1)$ demand point to the 1st intermediate travel point.
- t_B^i is the travel time from the 1st intermediate point to the m intermediate travel point.
- The value m depends from the robotic work cell complexity. In this paper, we set $m = 2$.
- t_C^i is the travel time from the m^{st} intermediate travel point to i demand point.

It is worth noting that, in general, the cycle/travel time which is required by the manipulator to move from point a using the k^{th} configuration to point b using the l^{th} configuration is given by [4]:

$$t = \max\left(\frac{\left|\theta_j^{b,l} - \theta_j^{a,k}\right|}{\dot{\theta}_j}\right), j = 0, \ldots, 3 \tag{5}$$

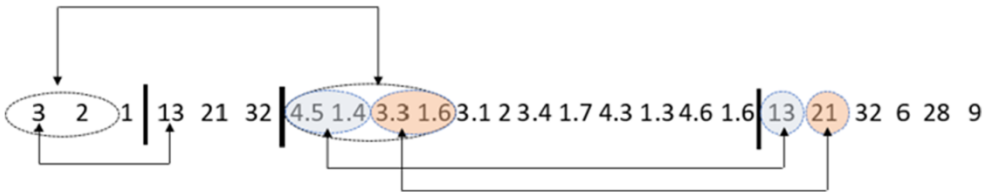

Fig. 13 The proposed chromosome architecture

where $k = 1, \ldots, 4M$ denotes the possible configurations at point a (similar for b), $\theta_j^{b,l}$ and $\theta_j^{a,k}$ are displacements of joint j at point b and a, respectively, and $\dot{\theta}_j$ is the velocity of j−th joint.

2.5 The objective function

In this paper, the combined problem of routing and motion planning is expressed by the following objective function:

$$F_{\text{obj}} = w^{*}t_{\text{cycle}} + (1-w)^{*}\text{flat}, \quad w \in [0, 1] \tag{6}$$

The minimization of F_{obj} with respect to the joints' angles provides the desired solution.

3 The proposed optimization algorithm

In order to solve the combined problem of routing and motion planning, we adopt GA [15] which has been proved that can work efficiently with problems with increasing complexity. GAs are one of the most powerful algorithms for search and optimization inspired by Darwin's evolutionary theory. Nowadays classified as metaheuristic search algorithms, GAs employ a parallel random yet directed search for finding the global optimum solution in the problem's solution space. GAs have been empirically found to perform better than the classical heuristics, gradient-based methods as well as other well-known metaheuristics especially when addressing large-sized combinatorial optimization problems.

3.1 The architecture

The proposed GA is developed as shown in Fig. 11 and includes the following components: (a) the chromosome syntax, (b) the decoding mechanism, (c) the evaluation mechanism, (d) the mechanism for population's generation and (e) the termination criterion.

3.1.1 The chromosome

Let us consider that the 4-link manipulator must visit N demand points in a robotic work cell. As described in Section 2, the solution of the IK for each demand point and each intermediate travel point gives $4 * 10 = 40$ possible configurations. Each chromosome consists of four parts. The first part is composed by N int. numbers and represents the unknown sequence of the demand points. The second part is composed by N integer labels where each label represents a possible configuration for each demand point and takes an integer value from $\{1, \ldots, 40\}$. The third part is composed by $2 * N * m$ real numbers, where $m = 2$ denotes the number of intermediate points between each pair of demand points. The fourth part is composed by $N * 2$ integer labels, where each label represents a possible configuration for each intermediate point and takes an integer value from $\{1, \ldots, 40\}$.

Figure 12 shows an example where a manipulator must visit 3 demand points which are positioned on the boundary of an obstacle in a robotic work cell which contains a triangular obstacle. A possible chromosome is shown in Fig. 13. Here, the first part of the chromosome determines that the order of the demand points should be $3 \rightarrow 2 \rightarrow 1 \rightarrow 3$. The second part determines the manipulator's configurations at demand points 3, 2 and 1, respectively, using the labels 13, 21 and 32. The third part determines the intermediate travel points between every pair of demand points, e.g. points (4.5, 1.4) and (3.3, 1.6) are the intermediate points between demand points 3 and 2. The fourth part determines the manipulator's configuration at

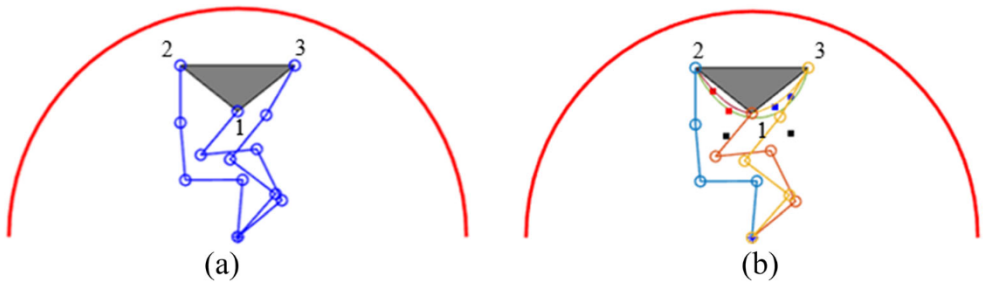

Fig. 14 **a** The proposed configurations at the demand points. **b** The end-effector's trajectory while it is moving between the demand points

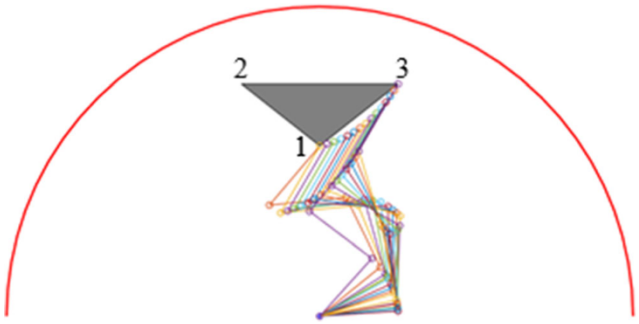

Fig. 15 The manipulator's motion while is moving from 1 to 3

each intermediate travel point. For example, the manipulator at the intermediate point (4.5, 1.4) should have the configuration labelled as 13.

3.1.2 The fitness function

The fitness function is needed as an evaluation mechanism to assess the quality of each chromosome. In this paper, the fitness function is given by:

$$\text{fitness} = \frac{1}{F_{\text{obj}}}, \tag{7}$$

3.1.3 Genetic operations

In the proposed GA, *reproduction* is based on the roulette wheel scheme, where the chromosomes that will be copied are selected with rates proportional to their fitness. For the first, the second and the fourth part of the chromosome, we used the order crossover, while for the third part of the chromosome, we used the one-point crossover. The mutation operator, which is used for the first, the second and the fourth part of the chromosome, is the inversion, while for the third part of the chromosome, we used the single bit mutation.

3.1.4 The population and the termination condition

In this paper, our GA starts with an initial random population, and the result of this run is used to 'seed' the initial population of the next run in the hopes of starting the evolution in a more useful region of the search space. The seeding percentage is set to be 10% of the initial population. Furthermore, the proposed GA terminates either when the maximum number of generations is achieved.

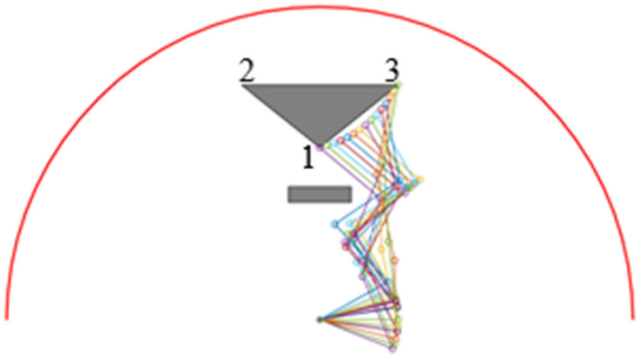

Fig. 17 The manipulator's configurations while it is moving from 1 to 3

4 Experiments

The proposed approach for optimal robot task scheduling has been tested with five sets of experiments, which are implemented in MATLAB and run on a core i5-8265 CPU @ 1.60 Hz, 8.0GB RAM. The GA's control parameters were experimentally determined after several preliminary investigations and by following the indicators of the literature. Their values are pop. size = 250, generations = 500, cross rate = 0.75 and mut. rate= 0.035. In order to evaluate and select the most appropriate values for the proposed control parameters, thirty runs were carried out for each experiment. The dimensionless kinematic parameters are as follows: $l_i = 1$, $i = 1, \ldots, 4$ and $\dot{\theta}_i = 0.5, i = 1, \ldots, 4$.

Experiment I The manipulator is requested to visit three demand points in a robotic work cell with one triangular-shape obstacle. The demand points are positioned on the obstacle's vertices (see Fig. 14). The produced sequence of demand points is $2 \rightarrow 1 \rightarrow 3 \rightarrow 2$. Figure 14 a illustrates the derived configurations at the demand points, while Fig. 14 b shows the produced intermediate points (with rectangular-shaped symbols) which are used to produce a collision-free motion of the

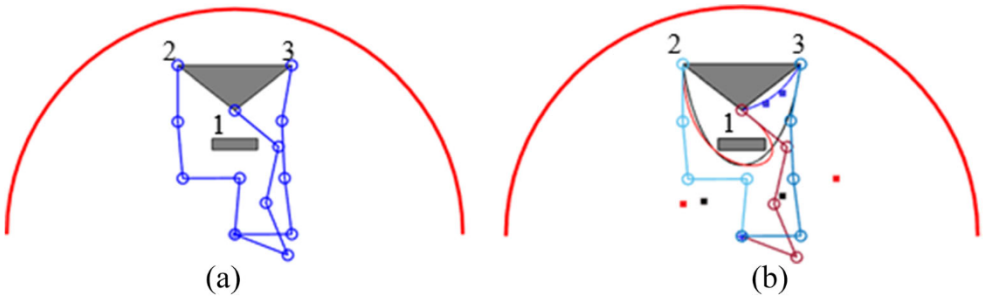

Fig. 16 **a** The proposed configurations at the demand points. **b** The end-effector's trajectory while it is moving between the demand points

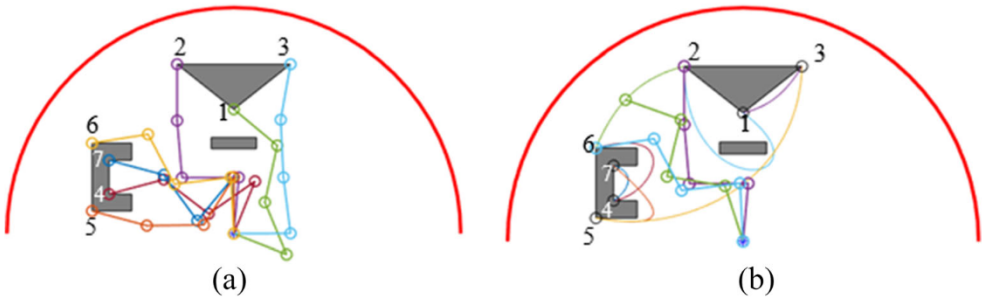

Fig. 18 **a** A schematic representation of the robotic work cell and the derived manipulator's configurations at the demand points. **b** The end-effector's trace while the manipulator is moving between the demand points

manipulator (red points are located between the demand points $2 \rightarrow 1$, blue points are located between the demand points $1 \rightarrow 3$, black points are located between the demand points $3 \rightarrow 2$). Furthermore, Fig. 14 b shows the end-effector's trajectory, presented by coloured curve, while it is moving between the demand points. The brown curve shows the end-effector's trajectory between the demand points $2 \rightarrow 1$, the green curve shows the end-effector's trajectory between the demand points $3 \rightarrow 2$, while the gold curve shows the end-effector's trajectory between the demand points $1 \rightarrow 3$. Figure 15 shows a set of manipulator's configurations corresponding to the end-effector travel from demand point 1 to 3.

Experiment II In this experiment, we add one more obstacle with rectangular shape in the robotic work cell as it is shown in Fig. 14. The new robotic work cell is presented in Fig. 16. The produced sequence of the demand points is $2 \rightarrow 1 \rightarrow 3 \rightarrow 2$. Figure 16 a shows the derived manipulator's configurations at the demand points, while Fig. 16 b shows end-effector's trajectory while the manipulator is moving and the proposed position of the intermediate points which are which are used to produce a collision-free motion of the manipulator (red points are located between the demand points $2 \rightarrow 1$, blue points are located between the demand points $1 \rightarrow 3$, black points are located between the demand points $3 \rightarrow 2$). The red curve shows the end-effector's trace between the demand points $2 \rightarrow 1$, the blue curve shows the end-effector's trace between the demand points $1 \rightarrow 3$, while the black curve shows the end-effector's trace between the demand points $3 \rightarrow 2$. Figure 17 shows a set of manipulator's configurations while the manipulator is requested to move from demand point 1 to demand point 3.

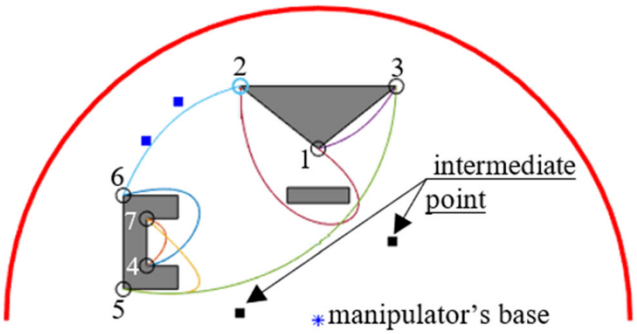

Fig. 19 The position of the intermediate points between the demand points $2 \rightarrow 6$ and $5 \rightarrow 3$

Experiment III In this experiment, one more non-convex obstacle has been added. Furthermore, the manipulator is requested to visit 7 different demand points as shown in Fig. 18. The proposed tour is $2 \rightarrow 6 \rightarrow 4 \rightarrow 7 \rightarrow 5 \rightarrow 3 \rightarrow 1 \rightarrow 2$. Figure 18 a shows the derived manipulator's configurations at the demand points, while Fig. 18 b shows (i) the end-effector's trajectory while the manipulator is moving between the demand points and (ii) the manipulator's configurations while it is moving between the demand points $2 \rightarrow 6$. Furthermore, we use two intermediate points between each pair of demand points. Indicatively, in Fig. 19, we show the intermediate points' positions between the demand points $2 \rightarrow 6$, with blue colour and between the demand points $5 \rightarrow 3$ with black colour.

In order to increase the complexity of the problem, we add in the work cell one more rectangular-shaped obstacle (see Fig. 20). The produced sequence of the demand points $2 \rightarrow 6 \rightarrow 4 \rightarrow 7 \rightarrow 5 \rightarrow 3 \rightarrow 1 \rightarrow 2$. Figure 20 a shows the derived manipulator's configurations at the demand points, while Fig. 20 b shows the end-effectors trajectory (using coloured curves) while the manipulator is moving between the demand points in the robotic work cell. In Fig. 20b, the intermediate points which are tagged with red rectangles are used to produce a collision-free motion while the manipulator is moving between the demand points $1 \rightarrow 2$.

Experiment IV The fourth experiment is even more complicated. Here, the robotic work cell includes three convex obstacles and two non-convex obstacles as shown in Fig. 21. The free space is limited restricting the manipulator's movements in the robotic work cell. The proposed solution tour is $1 \rightarrow 2 \rightarrow 6 \rightarrow 4 \rightarrow 7 \rightarrow 5 \rightarrow 9 \rightarrow 8 \rightarrow 3 \rightarrow 1$. Figure 21 a shows the derived manipulator's configurations at the demand point, while Fig. 21 b shows the end-effectors trace while the manipulator is moving between the demand points $8 \rightarrow 3$ which is defined by the two black rectangles. Furthermore,

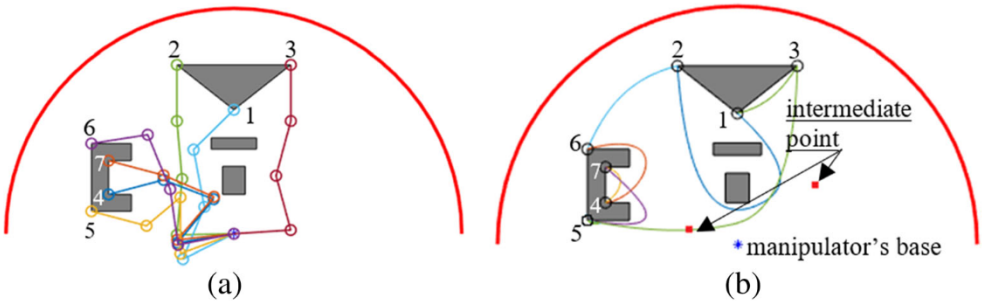

Fig. 20 **a** A schematic representation of the robotic work cell and the derived manipulator's configurations at the demand points. **b** The end-effector's trace while the manipulator is moving between the demand points

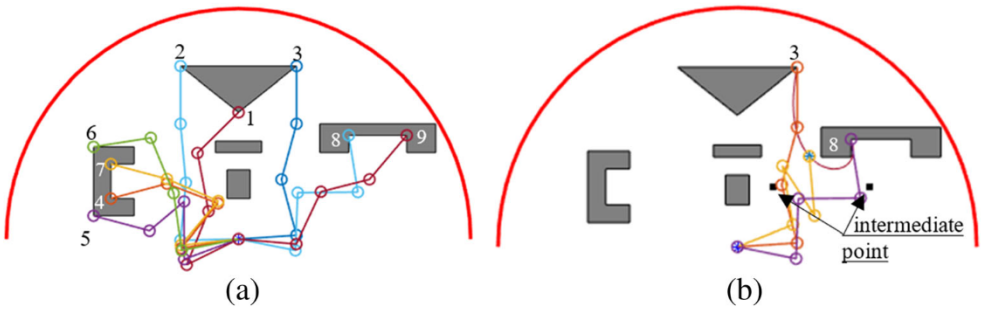

Fig. 21 **a** A schematic representation of the robotic work cell and the derived configurations at the demand points. **b** The proposed collision-free motion between the demand points $8 \rightarrow 3$

indicatively we present one intermediate configuration while the manipulator follows the proposed collision-free motion between the demand points $8 \rightarrow 3$.

Experiment V In this last experiment, we assume that the manipulator's end-effector should reach the demand points with specific orientation. We use the robotic work cell of the first experiment where the manipulator is requested to reach the demand point 2 with orientation 120°, the demand point 3 with orientation 60° and the demand point 1 with orientation 40°. The proposed tour is $3 \rightarrow 1 \rightarrow 2 \rightarrow 3$. Figure 22 a shows the proposed manipulator's configurations at the demand points, while Fig. 22 b illustrates the end-effector's trace while the manipulator is moving between the demand points and the position of the derived intermediate points (rectangular coloured points). The two black rectangles define the end-effector's trace between the demand points $1 \rightarrow 2$, the two red rectangles define the end-effector's trace between the demand points $2 \rightarrow 3$, and the two blue rectangles define the end-effector's trace between the demand points $3 \rightarrow 1$. Figure 23 illustrates two intermediate configurations while the manipulator is moving between the demand points $1 \rightarrow 2$.

4.1 The influence of M

To examine the influence of $M \in \{1, 6, 10\}$ in the dexterity of the manipulator, we used as an indicator the results taken from the fitness function. The results of all our experiments are summarized in Table 1, from which one can see that the selection of $M = 10$ random points in the intersection area (see sub-Section 2.2) gives the best values for the fitness function. We did experiments with

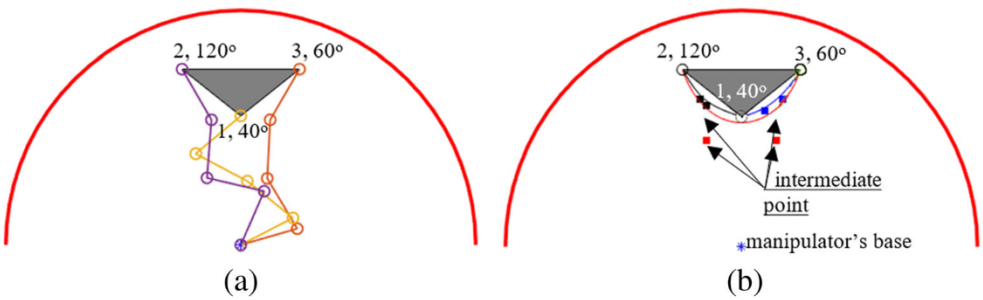

Fig. 22 **a** The derived configurations at the demand points. **b** The proposed collision-free motion

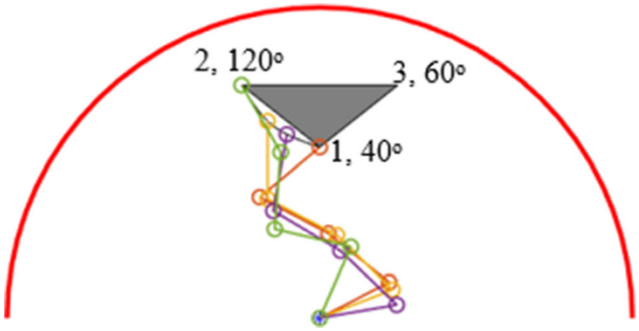

Fig. 23 Two intermediate configurations between the demand points $1 \rightarrow 2$

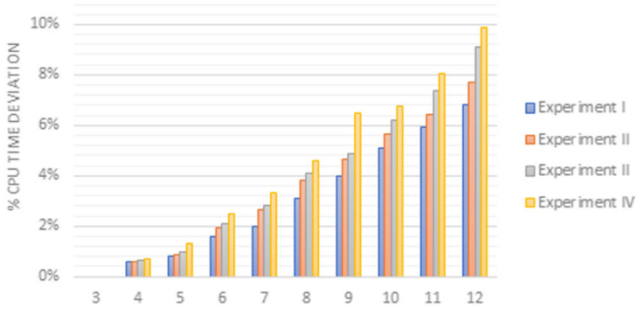

Fig. 24 The effect of demand points in CPU time

$M > 10$, but the improvement in the resulted fitness function was smaller than 1%.

4.2 Experimental study on performance of time

Computational time results of previous approaches are very limited in the relevant literature. So, it is difficult to present an extended comparison. An interesting indication of the computational time performance is the variation of the calculation versus the number of demand points, which is investigated experimentally. In the above experiments, we consider that the manipulator is assigned to reach consecutively $\{3, 4, 5, \ldots, 11, 12\}$ demand points. Figure 24 shows the percentage increase of the CPU time corresponding to 3 demand points (reference time) up to 12 demand points. The increase of the CPU time is almost linear.

Furthermore, we investigate by experiments the influence of M in computational time performance. We consider that the indicator M is consecutively increased by $\{1, 6, 10\}$ random points in the intersection area (sub-Section 2.2). Figure 25 shows the percentage increase from the CPU time corresponding to $M = 1$ random points (reference time) until to $M = 10$ random points.

It should be mentioned that the complexity of the robotic cell influences the flexibility of the manipulator to reach the desired task points. The general rule is that a big number for M gives the ability to the manipulator to determine easier the set of configurations which give us the minimum cycle time but on the other hand increases the CPU time. For this reason, after extensive experimental investigation, we concluded that $M = 10$ gives a balance between the CPU time and resulted cycle time. We did experiments with $M > 10$, but the improvement in the resulted cycle time was smaller than 1%, while the increased on the CPU time was exponential.

4.3 Comparative study

It is difficult to compare different techniques because they were tested on different types of environment, using different underlying libraries and implemented on different machines. However, one can compare the proposed method with a two-phase approach which considers the two operations (task scheduling and motion planning) separately. For the two-step approach in order to solve the TSP (determining the minimum cycle time), we used the approach which is proposed in [4] ignoring the obstacles which are located in the robotic cell. Then, by taking into account the proposed sequence of demand points, we apply the bump-surface concept [7, 8] in order to determine the collision-free paths between each pair of demand points. In order to evaluate the performance of the approaches, a series of experiments were carried out concerning the total cycle time which is needed to reach the requested demand points, as well as the intermediate configurations corresponding to points which are used to assist the

Table 1 The influence of choice M

No. Experiment	I			II			III (robotic work cell, Fig. 18)			IV		
Fitness function	M=1	M=6	M=10	M=1	M=6	M=10	M=1	M=6	M=10	M=1	M=6	M=10
	19.21	16.42	10.25	28.12	21.2	16.92	40.5	33.1	23.21	51.3	34.6	29.1

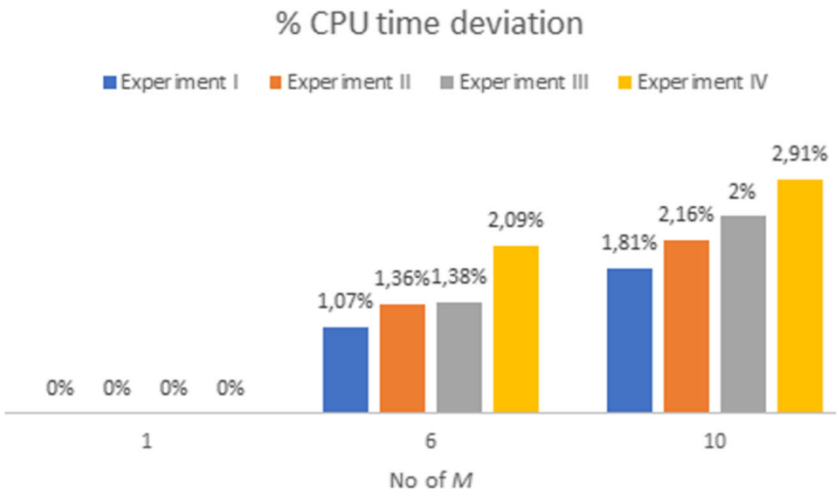

Fig. 25 The influence of M in CPU time

Table 2 Comparison between the proposed approach and the two-step approach

No. experiment		I	II	III (robotic work cell, Fig. 18)	IV
Proposed approach	Proposed sequence of demand points	2 → 1 → 3 → 2	2 → 1 → 3 → 2	2 → 6 → 4 → 7 → 5 → 3 → 1 → 2	1 → 2 → 6 → 4 → 7 → 5 → 9 → 8 → 3 → 1
	Cycle time	10.25	16.92	23.21	29.1
Two-step approach	Proposed sequence of demand points	2 → 3 → 1 → 2	2 → 3 → 1 → 2	2 → 6 → 7 → 4 → 5 → 1 → 3 → 2	9 → 8 → 5 → 4 → 7 → 6 → 1 → 2 → 3 → 9
	Cycle time	10.25	17.72	27.12	36.12

collision avoidance manoeuvre in the robotic work cell. The results of the comparison are presented in Table 2, from which one can see that the proposed approach provides better results than the two-step approach.

5 Conclusions

A new approach for the combined problem of motion planning and task scheduling for 4-link manipulator with rotational joint is presented in this paper. In contrast with other approaches which decouple the above problem into the task scheduling problem and the motion planning problem, the proposed approach integrates the two problems in one and resolves them simultaneously. The objective is the minimization of the cycle time by taking into account both the multiple solutions of the inverse kinematics at each demand point and the geometry of the robotic work cell. The inverse kinematic problem is solved by combining an adaptive neuro-fuzzy system and a genetic approach. The robotic work cell can have non-convex and convex obstacles shape while a set of demand points is positioned on the obstacles' vertices. The solution is searched using a genetic algorithm with special encoding.

A variety of criteria and constraints can be included which vary easily in the objective function which can affect the solution quality, such as (a) the geometry of the manipulator's end-effector tool, (b) the required time to accomplish an operation at each demand point and (c) the desired orientation of the end-effector at each demand point. All our experiments show that the proposed approach is effective and efficient and can determine collision-free (near) optimum routes in complicated robotic work cells. The derived solutions are always conformal to the problem's objectives **A** and **B**.

Furthermore, the proposed approach can solve the IK problem efficient within few seconds because it uses a geometrical approach and an ANFIS system in order to determine the multiple solution of the IK problem. Currently, we assume that manipulator is planar with 4 links and rotational joints which limits the applications of the proposed approach. The future directions are to apply the proposed approach (a) to robotic work cells with dynamic demand points, (b) in 3D robotic work cells, (c) to any non-redundant manipulator and (c) combination of the above.

Funding This research has been financially supported by General Secretariat for Research and Technology (GSRT) and the Hellenic Foundation for Research and Innovation (HFRI) (Code: 1184).

References

1. Dissanayake MWMG, Gal JA (1994) Workstation planning for redundant manipulators. Int J Prod Res 32(5):1105–1118. https://doi.org/10.1080/00207549408956990
2. Tubaileh AS (2015) Layout of robot cells based on kinematic constraints. Int J Comput Integr Manuf 28(11):1142–1154. https://doi.org/10.1080/0951192X.2014.961552
3. Chen C.-H., Chen L.-C., & Hwang W.-S (2017) "Optimization of robotic task sequencing problems by using inheritance-based PSO", 2017 56th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE). https://doi.org/10.23919/sice.2017.8105451
4. Zacharia PT, Aspragathos NA (2005) Optimal robot task scheduling based on genetic algorithms. Robot Comput Integr Manuf 21(1):67–79. https://doi.org/10.1016/j.rcim.2004.04.003
5. Latombe J-C (1991) Introduction and overview. Robot Motion Planning:1–57. https://doi.org/10.1007/978-1-4615-4022-9_1
6. Boysen N, Stephan K (2016) A survey on single crane scheduling in automated storage/retrieval systems. Eur J Oper Res 254(3):691–704. https://doi.org/10.1016/j.ejor.2016.04.008
7. Xidias EK, Zacharia PT, Aspragathos NA (2010) Time-optimal task scheduling for articulated manipulators in environments cluttered with obstacles. Robotica 28(03):427–440. https://doi.org/10.1017/s0263574709005748
8. Azariadis PN, Aspragathos NA (2005) Obstacle representation by bump-surfaces for optimal motion-planning. Robot Auton Syst 51(2–3):129–150. https://doi.org/10.1016/j.robot.2004.11.001
9. Zacharia PT, Xidias EK, Aspragathos NA (2013) Task scheduling and motion planning for an industrial manipulator. Robot Comput Integr Manuf 29(6):449–462. https://doi.org/10.1016/j.rcim.2013.05.002
10. Huang Y, Gueta LB, Chiba R, Arai T, Ueyama T, Ota J (2013) Selection of manipulator system for multiple-goal task by evaluating task completion time and cost with computational time constraints. Adv Robot 27(4):233–245. https://doi.org/10.1080/01691864.2013.755244
11. Lattanzi L, & Cristalli C (2013) An efficient motion planning algorithm for robot multi-goal tasks. 2013 IEEE International Symposium on Industrial Electronics. https://doi.org/10.1109/isie.2013.6563727
12. Baizid K, Yousnadj A, Meddahi A, Chellali R, Iqbal J (2015) Time scheduling and optimization of industrial robotized tasks based on genetic algorithms. Robot Comput Integr Manuf 34:140–150. https://doi.org/10.1016/j.rcim.2014.12.003
13. Alatartsev S, Stellmacher S, Ortmeier F (2015) Robotic task sequencing problem: a survey. J Intell Robot Syst 80(2):279–298. https://doi.org/10.1007/s10846-015-0190-6
14. Jang J.-SR (1993) ANFIS: adaptive-network-based fuzzy inference systems. IEEE Trans. Syst., Man, and Cybernetics, 23(03), 665-685, May 1993
15. Goldberg DE (1989a) Genetic algorithms in search, optimization, and machine learning. Addison−Wesley