Robotica (2010) volume 28, pp. 427–440. © Cambridge University Press 2009
doi:10.1017/S0263574709005748

Time-optimal task scheduling for articulated manipulators in environments cluttered with obstacles

E. K. Xidias, P. Th. Zacharia* and N. A. Aspragathos

Department of Mechanical Engineering and Aeronautics, University of Patras, Greece

(Received in Final Form: April 15, 2009. First published online: August 24, 2009)

SUMMARY

This paper proposes a new approach for solving a generalization of the task scheduling problem for articulated robots (either redundant or non-redundant), where the robot's 2D environment is cluttered with obstacles of arbitrary size, shape and location, while a set of task-points are located in the robot's free-space. The objective is to determine the optimum collision-free robot's tip tour through all task-points passing from each one exactly once and returning to the initial task-point. This scheduling problem combines two computationally NP-hard problems: the optimal scheduling of robot tasks and the collision-free motion planning between the task-points.

The proposed approach employs the bump-surface (B-Surface) concept for the representation of the 2D robot's environment by a B-Spline surface embedded in 3D Euclidean space. The time-optimal task scheduling is being searched on the generated B-Surface using a genetic algorithm (GA) with a special encoding in order to take into consideration the infinite configurations corresponding to each task-point. The result of the GA's searching constitutes the solution to the task scheduling problem and satisfies optimally the task scheduling criteria and objectives. Extensive experimental results show the efficiency and the effectiveness of the proposed method to determine the collision-free motion among obstacles.

KEYWORDS: Articulated robots; Task scheduling; Bump-surface, Genetic algorithms.

1. Introduction

In recent years, redundant articulated robots are of immense interest due to the extra capabilities they perform. Considerable attention has been paid to industrial applications, where the robot is assigned to reach a sequence of task-points with its hand tip in environments cluttered with obstacles. This problem occurs often in practice, e.g. in spot welding, multiple drilling, laser cutting, car painting and inspection tasks. The sequence with which the robot will reach a number of task-points and return to the starting point is not predetermined. On the other hand, the constraint for collision-free movements should also be considered.

The goal of the research for this paper is to determine the optimum sequence of robot's task-points amidst obstacles. This problem is usually divided into two separate computationally hard problems: Optimal Sequencing, closely related to the travelling salesman problem (TSP)[1] and path planning problem.[2] Sequencing arranges the order of the task-points to minimize an objective function, such as time or travelled distance. Path planning determines an efficient and collision-free path between two task-points.

Optimal Sequencing: The TSP is one of the most widely discussed NP-hard problems in combinatorial optimization.[3,4] The salesman has to visit exactly once each one of a finite number of cities and return to the starting city. Given the distances between the cities, the objective of the optimization is to find the shortest path that includes each city exactly once. Adapting the TSP to robotics, the problem becomes harder due to the fact that the objective is to determine the minimum possible cycle time in order that the robot's end-effector (EE) passes through the given task-points (cities) and simultaneously ensure obstacle avoidance. Besides, the multiple solutions of the Inverse Kinematics Problem should also be considered.

Some authors[5–8] have worked on finding the optimum sequence of the robotic task performance (the minimum cycle time of a manipulator visiting a number of task-points). Abdel-Malek *et al.*[5] developed an algorithm for the determination of the minimum cycle time taking into consideration the multiplicity of robot configurations. However, this algorithm was not extended for robots with more than two configurations and there is no indication that can be extended by further elaboration. The elastic net method (ENM)[6] and the Little's branch and bound algorithm[7] have also been used in order to minimize the cycle time of robotic tasks. These algorithms work successfully for low degrees of freedom (DOF) and a small number of task-points. However, in 5% of the tests, the ENM did not converge due to wrong choice of the algorithmic parameters. To alleviate these difficulties, genetic algorithms (GAs) have been successfully employed for non-redundant manipulators operating in a 2D or 3D environments that are requested to visit a large number of task-points.[8]

Most of the aforementioned methods do not consider obstacles in the robot's environment and are limited to low DOF manipulators. Petiot *et al.*[6] presented in a method that takes into account the obstacles, but it is limited to convex obstacles that can be represented in the workspace. The authors admit that it is difficult to generalize the method in more than 3-DOF robots because of the increasing time cost.

* Corresponding author. E-mail: zacharia@mech.upatras.gr

https://doi.org/10.1017/S0263574709005748 Published online by Cambridge University Press

Path Planning: The basic motion planning problem refers to the computation of an optimal collision-free tour for the manipulator's end-effector tip between two task-points. Generally, there are three main approaches for solving the robot motion planning problem: *the pseudo-inverse approach*, *techniques based on the Configuration Space* and *optimization techniques based on the direct kinematics*.

The pseudo-inverse approaches for obstacle avoidance of redundant manipulators are based on the Jacobian and make use of the inverse kinematics. Generally, the primary task is associated with works performed at the end-effector and utilization of the redundancy refers to the secondary (or constraint) tasks. Thus, the technique is employed to optimize a criterion for obstacle avoidance, which is usually the maximization of the distances between links and obstacles.[9,10,11] These methods, which are local approaches, have been successfully applied to simply shaped obstacles and planar manipulators with low redundant DOF. Besides, they pose the limitation of yielding feasible solutions when the range of motion is small.

The second group consists of approaches based on the Configuration Space (C-Space),[12] where each pose (position and orientation) of a robot is represented by a point and the obstacles are expanded appropriately. The motion-planning methods based on the C-Space are classified in three main categories[2]: *cell decomposition, potential fields and roadmaps*. The cell decomposition methods decompose the robot's free space into cells that form the nodes of a non-directed graph, representing the adjacency relation between the cells. However, these approaches are usually too complex to be useful. In the techniques based on the artificial potential fields, obstacles are represented by repulsive surfaces, while the target position is represented as an attractive pole so that the manipulator may reach the goal without colliding with obstacles. The main disadvantage of these techniques is the presence of local minima, where the robot can not escape.[13] In the roadmaps approach, the robot's free space is mapped onto a network of one-dimensional lines. Roadmaps are constructed based on different methods, such as visibility graph, Voronoi diagrams, freeway nets and silhouettes. The above methods are global in nature and can generate a path, but they cannot determine a path satisfying some desired objectives and criteria, such as the shortest path and the smoothness of the path. In addition, they usually include a huge number of nodes causing inefficiencies.[14]

To circumvent the limitations and the drawbacks of the two aforementioned techniques, optimization techniques based on the direct kinematics, such as GAs, have gained significant impetus as an effective alternative to the solution of the path planning problem. GAs[15,16] are global search techniques based on natural selection and the 'survival of the fittest'. They have the advantage over the gradient methods that they do not require the derivative of the function. In addition, they can handle any kind of functions (non-linear, non-continuous and procedural) and any kind of constraints (linear or nonlinear).

Several authors[17–21] have worked on robot motion planning using the aforementioned approaches. Liang *et al.*[17] use the pseudo-inverse with null-space projection method that uses the gradient of the objective function. In their approach, the path points are prespecified and are planned according to the geometry of the constrained workspace. The proper postures are determined using a combination of the potential field method and the elastic net method. According to Conkur,[18] the basic idea for motion planning is to find a smooth path consisting of points close enough to each other using harmonic potential fields and then, to keep the tip of each link on these path points until the manipulator reaches the goal. Conkur[14] assumes that an approximated path is given by B-spline curves and path following is defined by requiring manipulator links to remain approximately tangent to these curves. The positions of the links are checked by a propagation procedure. The path planning approaches[19,20,21] discussed in the following are based on GAs. Nearchou *et al.*[19] formulated the problem as a constrained optimization problem with the main objective of minimizing the end-effector's positional error and a secondary objective of satisfying the constraints related to the collision-free movement of the robot. The collision is detected by checking for interference between the links closely located to the obstacles and the associated obstacles. Tian *et al.*[20] formulated a constraint by keeping the trajectory of the end-effector away from the obstacle. In their work, a floating point representation is used for the GA. To determine the optimal motion planning between two task-points, Azariadis *et al.*[21] used for the first time the novel notion of obstacle representation by B-surfaces.

The objective of the present work is the development of a new task scheduling approach for articulated (redundant and non-redundant) robots in constrained environments considering simultaneously both NP-hard problems: *Optimal sequencing and path planning*. The proposed approach takes into account the information of the entire robot's environment to solve the task scheduling problem and searches for a near-optimum solution using a GA. The B-Surface concept[21] is used to formulate a searching space represented by a single entity, where the optimum sequence of the task-points should be determined and simultaneously motion planning of the robot's tip between the obstacles should be obtained. A global optimization problem is formulated considering simultaneously both problems and is solved using a GA with a special encoding. The main advantage of our approach concerns the determination of the optimum sequence of the task-points ensuring collision avoidance, while simultaneously choosing the appropriate robot's configuration at each task-point. Unlike other methods that are limited to the sequencing problem for non-redundant robots or the path planning problem between only two task-points, our method is generalized to both non-redundant and redundant robots and resolves the global optimal task scheduling problem. In addition, the proposed method maintains the robot in its physical Cartesian space and thus, it does not need the construction of the configuration-space obstacles.

The rest of the paper is organized as follows: in Section 2 the robot scheduling problem is described. Section 3 presents the B-Surface concept and the basic notions for articulated robots, while in Section 4 the task scheduling problem is formulated. Section 5 presents the proposed optimization algorithm and Section 6 demonstrates and

https://doi.org/10.1017/S0263574709005748 Published online by Cambridge University Press

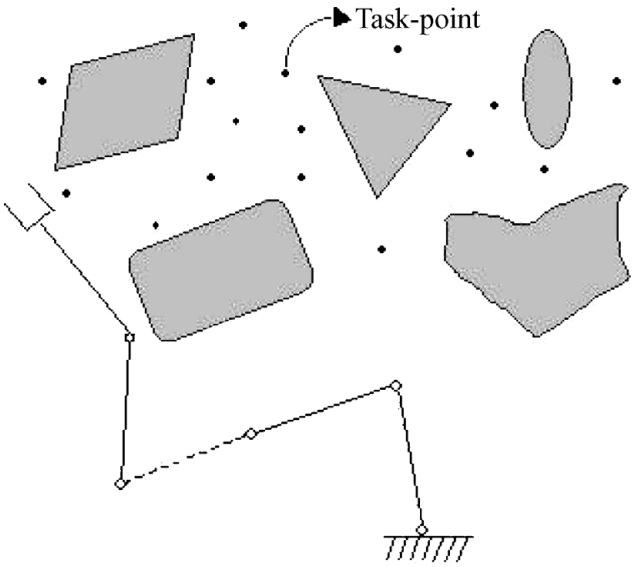


Fig. 1. An n-dof planar articulated robot in a 2D environment cluttered with obstacles and a finite number of task-points.

discusses the efficiency of the proposed method through multiple experiments applied for planar manipulators with four DOF. Finally, Section 7 summarizes the contribution of the paper.

2. Problem Statement

Let a planar articulated robot (either redundant or non-redundant) with n rotational joints. Assume that the robot operates in a 2D environment (2D *task-space*) cluttered with a priori known static obstacles of arbitrary size, shape and location and a set of M task-points to be visited by the robot's end-effector (Fig. 1). Ignoring the dynamic properties of the robot, the robot's motion is determined by its kinematic model and constrained by the obstacles. In addition, the robot stops at each task-point in order to perform an operation for fixed time, which is not considered in the total cycle time optimization, since it depends on the process. The problem addressed in this paper is to find the sequence with which the robot should visit all the task-points exactly once and return to the initial task-point, so that the total cycle time is minimum, while simultaneously ensuring that no collision occurs between the robot links and the obstacles. The determined tour should satisfy the following task scheduling criteria:

(a) The robot's end-effector tip should pass through each one of the task-points exactly once and return to the initial task-point in the minimum cycle time, taking into consideration the multiplicity of robot configurations at the task-points.
(b) The manipulator should not collide with the obstacles along the tour.

The path of the robot's tip is represented by a first-degree B-Spline curve in order to take advantage of the local attributes and the well-known numerical stability of the respective computational implementations.[22]

The manipulator's task scheduling problem combines two NP-hard problems: the TSP[1] and the motion planning problem.[2] The first problem (adapted to robotics) refers to the determination of the optimal sequence between the task-points in terms of time and the second problem deals with the determination of the collision-free paths between the successive goals. The task scheduling problem can be characterized as a combinatorial NP-hard problem. Due to the combinatorial explosion, the extraction of exact optimal solutions for NP-hard problems is computationally impracticable. Thus, the reduction of the search space complexity has a great impact on the final optimum solution. In our approach, this is achieved by using the B-Surface concept to formulate a search space represented by a single mathematical entity. On the other hand, research on combinatorial explosion based on metaheuristics, such as GAs, can lead to approximate solutions in polynomial time instead of exact solutions that would be at intolerably high cost.

3. Preliminaries

3.1. Kinematics of articulated robots

The kinematics equations of a nth DOF planar manipulator with rotational joints operating in a 2D Cartesian Space (Fig. 2) is expressed by Eqs. (1)–(2):

$$x_{i+1} = x_0 + \sum_{k=0}^{i} \ell_k \cos\left(\sum_{j=0}^{k} \theta_j\right), \quad i = 0, 1, \ldots, n-1, \tag{1}$$

$$y_{i+1} = y_0 + \sum_{k=0}^{i} \ell_k \sin\left(\sum_{j=0}^{k} \theta_j\right), \quad i = 0, 1, \ldots, n-1, \tag{2}$$

where $\ell = (\ell_0, \ell_1, \ldots, \ell_{n-1}) \in \Re^n$ are the lengths of the arm's links, $\boldsymbol{\theta} = (\theta_0, \theta_1, \ldots, \theta_{n-1}) \in \Re^n$ are the angle variables of the arm's joints and $\mathbf{p} = [(x_0, y_0), (x_1, y_1), \ldots, (x_k, y_k), \ldots, (x_n, y_n)] \in \Re^{(n+1)\times 2}$ are the Cartesian coordinates of the arm's joints with respect to a fixed coordinate system.

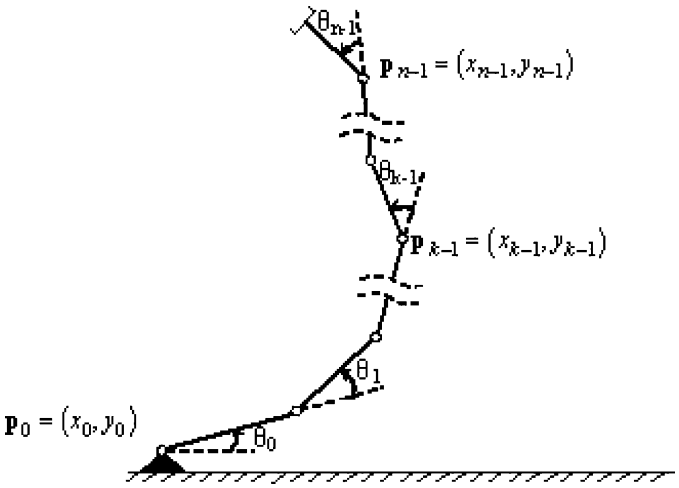


Fig. 2. n-dof manipulator operating in the two dimensional space.

https://doi.org/10.1017/S0263574709005748 Published online by Cambridge University Press

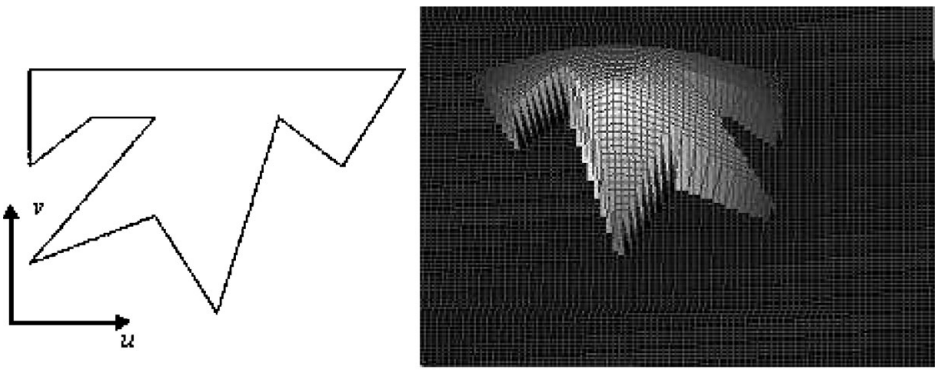


Fig. 3. (a) The initial 2D environment with a non-convex polygonal obstacle. (b) The resulting B-surface.

Inverse kinematics concerns the solution of the above kinematic equations, which are nonlinear and very complex, to obtain the joint variables θ_j, $j = 0, \ldots, n-1$. In the case of redundant manipulators with n-DOF, there is an infinite number of configurations corresponding to a given end-effector position, since the kinematic mapping from Cartesian space to joint space is not one to one.

3.2. The bump-surface concept

The B-surface concept is an effective tool to capture both the free and the forbidden areas of the entire robot's workspace by using a single mathematical entity and can be used to solve complicated motion-planning problems.[23,24] It is a global method that represents the overall robot's workspace by using a B-Spline surface embedded in a higher dimension space. For example, a 2D static environment is represented by a 2D B-Spline surface embedded in 3D Euclidean space. The construction of the B-surface is based on control-points net with variable density depending on the required motion-planning accuracy (higher accuracy is achieved using denser grid of control-points). In addition, the desired accuracy can be achieved by taking advantage of the flexibility of the B-Spline surfaces ability for local and global control, e.g. by changing the position of the control points (global control) or by modifying the knot vectors (local control).

Due to the flexibility of the B-Spline surfaces, regions containing narrow passages are captured very easily on the B-surface. Individual motion planning problems can then be formulated as optimization problems, where a variety of motion planning objectives and constraints are encoded in an objective function. The motion planning solution is searched onto the resulted surface (Bump-Surface = domain space) in such a way that its inverse image on the initial environment satisfies the given motion-planning objectives and constraints. A variety of criteria can be easily included in the objective function taking into account several factors that affect the solution-path quality (e.g. non-holonomic constraints, curvature continuity, robot's dimension, etc). In contrast to the potential field approaches, the B-surface concept represents the entire robot environment by one simple mathematical entity (B-Spline surface), which is constructed in linear time. Another advantage is that the resulted surface is free from dead-ends, where a robot could be trapped, owing to the construction of the B-surface (contains flat areas and bumps).

Given a 2D workspace cluttered with a priori known static obstacles, the construction of the B-Surface is a straightforward application of the $z-$*value algorithm*.[21] This algorithm considers that the robot's environment which is enclosed within a square $[0, 1]^2$ area, which is the actual *normalized space*, is discretized into equally-spaced subintervals along its two orthogonal directions (denoted by u and v, respectively), forming a grid of points $\mathbf{p}_{ij} = (x_{ij}, y_{ij}, z_{ij}) \in [0, 1]^3$, $0 \leq i, j \leq N-1$, where N denotes the grid size. The $z_{ij}-$coordinate of each grid point $\mathbf{p}_{ij}$ takes a value in the interval (0, 1] if the corresponding grid point lies within an obstacle, and the value 0 if the corresponding grid point lies in the free space.

The B-Surface is constructed using the mapping $S : [0, 1]^2 \rightarrow [0, 1]^3$ which is represented by a tensor product B-Spline surface with uniform parameterization:[22]

$$S = \mathbf{S}(u, v) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} N_{i,p}(u) N_{j,q}(v) \mathbf{p}_{ij}, \qquad (3)$$

where $N_{i,p}$ and $N_{j,q}$ are the basis function and p, q are the degrees in the u and v parametric directions of the surface respectively. A visual interpretation of this construction is depicted in Fig. 3. The grid points within the obstacle area shown in Fig. 3(a) are lifted along the $z-$axis of the orthonormal coordinate system (x, y, z) resulting the B-Surface in the Euclidean space shown in Fig. 3(b).

For the sake of simplicity, we consider that the *normalized space* is the actual parametric space of S. The n-DOF articulated robot described by the Eqs. (1) and (2) is considered as a first degree B-Spline curve $C(s) = (u(s), v(s))$ defined in the parametric space as:

$$C = \mathbf{C}(s) = \sum_{k=0}^{n} N_{k,1}(s) \mathbf{p}_k = \sum_{k=0}^{n} N_{k,1}(s)(x_k, y_k), \qquad (4)$$

where $\mathbf{p}_k$ represents the position of the kth joint and $N_{k,1}$ is the basis function. Thus, it is able to represent the robot

https://doi.org/10.1017/S0263574709005748 Published online by Cambridge University Press

by a one-parametric curve $\mathbf{S}(\mathbf{C}(s))$, $s \in [0, 1]$ lying onto S, which provides the necessary motivation to perform a search for an optimal total cycle time on S in order to satisfy the aforementioned task scheduling criteria **a**-**b**.

It has been proved[21] that using the B-Surface for path planning, the searching time does not depend on the complexity of the workspace, namely the number and shape (i.e. the number of vertices) of the obstacles, since the search space is one mathematical entity, called B-Surface. In addition, the motion-planning problem can be easily enriched with more qualitative characteristics like the smoothness of the path followed by the end-effector's tip and the width of the robot links.

4. Task Scheduling Among Obstacles for Articulated Robots

The task scheduling problem is the problem of determining the optimum sequence of the task-points visited by the end-effector, so that no collision occurs during the manipulator motion from task-point to task-point. Although it is easier to decompose the optimal task scheduling problem into two separate subproblems, such an approach would degrade the quality of the final solution. In the following, the B-Surface concept is presented for the obstacle avoidance (Section 4.1) and then, the global task scheduling problem of an articulated robot operating in an environment cluttered with obstacles is addressed (Section 4.2).

4.1. Collision-free motion

The articulated robot is modelled as a first-degree B-Spline curve defined in the 2D normalized space $[0, 1]^2$ (see Section 3.2), where the n-joints are the control points that define the first-degree B-Spline curve. Using the B-surface concept, all feasible configurations of the robot are represented onto the B-Surface S through a family of one-parametric curves lying on S.

Following the results,[21] a continuous collision-free motion for the manipulator that moves its end-effector between pairs of task-points should be searched in the 'flat' areas of the B-Surface. A movement that 'climbs' the bumps of the B-Surface results in a movement of very high cost in $[0, 1]^2$, and thus an 'invalid' movement that penetrates the obstacles. By construction, the arc-length of C approximates the arc-length of its image $\mathbf{S}(\mathbf{C}(s))$ on S as long as C does not penetrate the obstacles. Therefore, it is reasonable to search for a 'flat' image of the manipulator on S in order to satisfy the aforementioned objective **b**.

The 'flatness' of the image of the curve C (manipulator) on S is measured by:

$$Flat = \int_0^1 \mathbf{S}_z(\mathbf{C}(s))\,\mathrm{d}t, \tag{5}$$

where $\mathbf{S}_z(\mathbf{C}(s)) \geq 0$ denotes the $z-$coordinate of the image C on S. The minimization of *Flat* with respect to the joint variables $\boldsymbol{\theta} \in \Re^n$ satisfies the requirement for collision-free robot's configurations.

4.2. Formulation of the objective function for optimum sequence of the task-points ensuring collision avoidance

Let an n-DOF planar manipulator with rotational joints operating in a 2D Cartesian space that is assigned to visit M task-points, the locations of which are known (Fig. 1), aiming at the objectives discussed in Section 2.

The objective **a** concerns the determination of the optimum travel time for visiting all the task-points exactly once and return to the initial task-point. The optimum travel time is determined by the multiplicity of the robot configurations corresponding to each one task-point, as well as the presence of obstacles in the environment. Consequently, the configuration choice depends on the optimum travel time criterion and the shape and location of the obstacles, since the configuration choices are limited due to the presence of the obstacles.

The time t_i spent by the manipulator to travel from one configuration corresponding to the point $(i-1)$ to one configuration corresponding to the point i can be written as:

$$t_i = \max_j \left(\frac{\left|\boldsymbol{\theta}_{ji} - \boldsymbol{\theta}_{j(i-1)}\right|}{\dot{\theta}_j} \right) \qquad \begin{matrix} i = 1, 2, \ldots, \mathrm{M} \\ j = 1, 2, \ldots, n \end{matrix}, \tag{6}$$

where $\boldsymbol{\theta}_{ji}$ is the jth-joint displacement for the ith end effector location and $\dot{\theta}_j$ is the average velocity of the jth-joint that is assumed to be constant. This approximation is reasonable on condition that the time corresponding to acceleration and deceleration is very small. Equation (6) denotes the fact that the travel time between two positions is determined by the slowest manipulator's joint. Due to the multiplicity of solutions of the inverse kinematics problem, the choice of the manipulator's configuration affects significantly the travel time. Thus, the total travel time t_c needed to visit all the task-points and return to the initial task-point is given by

$$t_c = \sum_{i=2}^{\mathrm{M}\cdot(1+\mathrm{r})} t_i + \max_j \left(\frac{\left|\boldsymbol{\theta}_{j1} - \boldsymbol{\theta}_{j\{\mathrm{M}\cdot(1+\mathrm{r})\}}\right|}{\dot{\theta}_j} \right), \tag{7}$$

where r is the number of the intermediate configurations between two successive task-points. It should be noticed that the trajectory of an articulated robot consists of r intermediate configurations between two successive task-points. The number of the intermediate configurations r depends on the difficulty and complexity of the 2D environment. Using a large number for r, a tour with higher accuracy is derived but the computational time becomes too long. Therefore, the selection of r depends on the complexity of the problem and serves as a trade-off between computational cost and tour accuracy.

The objective **b** imposes the requirement that the robot's end-effector should move among the task-points ensuring collision avoidance. In order to avoid infeasible configurations, i.e. configurations colliding with obstacles, the 'flatness' of the image of curve C on S has to be considered.

The requirement for the robot's end-effector to reach the position of the task-point is defined by a function, expressing the position error between the current position of the end-effector's tip $\mathbf{p}_n$ and the position of the ith task-point $\mathbf{P}_i$, and

https://doi.org/10.1017/S0263574709005748 Published online by Cambridge University Press

is formulated as:

$$D = \sum_{i=1}^{M} \|\mathbf{p}_n - \mathbf{P}_i\|^2, \tag{8}$$

where $\mathbf{P}_i$ is defined in the parametric space (see Section 3.2).

Therefore, the combined function is

$$\mathrm{E} = w_1 t_c + w_2 Flat + w_3 D, \tag{9}$$

where w_1, w_2, w_3 are weight factors, $w_1 + w_2 + w_3 = 1, 0 \leq w_1, w_2, w_3 \leq 1$ expresses the total cycle time obtained taking into account the configurations at the M task-points and the r intermediate configurations and simultaneously ensures that no obstacle collision-free motion between these configurations. Besides, the motion between two successive configurations is designated considering that the joint variables change linearly with the time. The minimization of the combined function E,

$$\min_{\theta} \mathrm{E} = \min_{\theta} \left(w_1 t_c + w_2 Flat + w_3 D\right), \tag{10}$$

where $\boldsymbol{\theta} = (\theta_0, \theta_1, \ldots, \theta_{n-1}) \in \Re^n$ and $\theta_j \in [0, 2\pi)$, $j = 0, \ldots, n-1$ provides the required solution to the aforementioned task scheduling problem.

5. The Optimization Algorithm

A key point in solving the task scheduling problem is the selection of the optimization algorithm. GAs, introduced by Holland[25], have the advantage over the gradient optimization methods that they do not require the derivative of the objective function. Unlike the traditional optimization techniques that require one starting point, GAs perform a search in a population of points and are based on probabilistic transition rules (represented by the genetic operators) instead of deterministic rules. As a result, they are more likely to escape local optima and converge to the near global solution. In addition, GAs can handle any kind of constraints (i.e. linear or nonlinear defined on discrete, continuous or mixed search space). Although GAs are not as fast as traditional methods, they are perfectly suited for difficult problems, where there is no derivative information available. Bearing in mind the fact that our objective function is procedural, non-continuous, non-linear and multimodal, as well as the fact that the search space is complex, GAs have been chosen as a powerful tool for optimization.

5.1. The representation mechanism

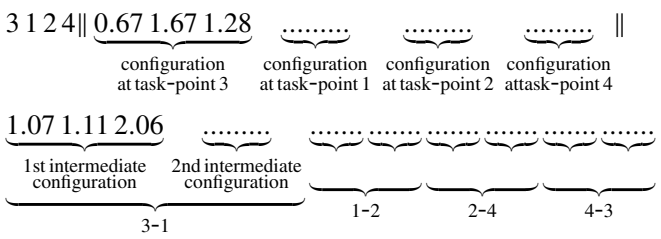
The first step in applying the GA is the choice of an appropriate representation to encode the optimization variables of the current optimization problem. The solution is represented as a string over a specific alphabet. Binary representation of solutions has dominated GA applications, since the binary alphabet offers the maximum number of schemata per bit of information of any encoding.[15] However, the binary encoding exhibits worse performance than floating encoding when applied to multidimensional, high precision numerical problems. The fact that a large number of variables postulates a great increase of the length of the chromosome results in large computational costs. In addition, given a fixed binary length, binary representation must sacrifice precision in case of an increase in domain size.[16]

On the contrary, the floating point representation is capable of representing quite large domains without increasing the computational complexity, which results in saving computer memory and computation time. Michalewicz[16] has showed experimentally that floating point representation is faster, more consistent from run to run and provides a higher precision, especially for large domains where binary coding would require prohibitively long representation.

As far as the TSP is concerned, the binary representation of tours is not well suited, since a change of a single bit may result in an illegal tour and thus, the binary code of the cities would require special repair algorithms. There is an agreement in the GA community that integer representation of tours is suited best for the TSP.[16]

For the case of an n-DOF manipulator that has to visit M task-points in the two-dimensional space with an infinite number of configurations corresponding to each task-point, each chromosome consists of $M + M \times n + M \times r \times n$ genes, where r is the number of intermediate configurations of the path between two successive task-points. The first part of the string, composed of M integer numbers, represents the sequence with which the manipulator reaches the M task-points. The second part of the string is composed of M n floating numbers. Each gene stands for a joint angle variable and each set of n genes represents one of the infinite possible configurations corresponding to the task-points defined in the first part of the chromosome. The third part of the chromosome is composed of $M \cdot r \cdot n$ floating numbers, where each gene stands for a joint angle variable. Each set of $r \cdot n$ genes represents r of the infinite possible intermediate configurations between two successive task-points, as defined in the first part of the chromosome.

For example, for a 3-DOF manipulator that has to visit 4 points in the 2D space through a path defined by $r = 2$ intermediate configurations, a possible chromosome is formed as:



implying that the manipulator reaches task-point 3 with the configuration expressed by the genes $\{0.67, 1.67, 1.28\}$, then reaches task-point 1 with the configuration represented by the next three genes, etc. The path between two successive task-points is determined by two intermediate configurations and, in the above example, the manipulator moves from task-point 3 to task-point 1 on a path determined by the two intermediate configurations $\mathrm{c}_1 = \{1.07, 1.11, 2.06\}$ and $\mathrm{c}_2 = \{\ldots, \ldots, \ldots\}$, etc.

https://doi.org/10.1017/S0263574709005748 Published online by Cambridge University Press

5.2. The evaluation mechanism

The fitness function, derived from a proper combination of objective functions, is the evaluation mechanism to assess the quality (i.e. the fitness) of each chromosome of the population. The fitness function of the problem at hand is expressed by:

$$fitness = \frac{1}{\mathrm{E}}, \tag{11}$$

where $\mathrm{E} > 0$, since it is a combination of time and distance that can not be zero.

5.3. Genetic operators

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. *Crossover* is a recombination operator and follows the reproduction. For the part of the chromosome, consisted of integers, the Order Crossover (OX)[16] is used and for the part of floating numbers, the one-point crossover is used. *Mutation* is applied in order to inject new genetic material into the population and thereby avoid premature convergence to local minima. The mutation operator used for the part of the chromosome consisting of integers is the inversion.[16] Inversion is simply applied to a chromosome and guarantees that the resulting offspring represents a 'legal' tour. For the part of the chromosome, consisting of floating numbers, the mutation operator is applied changing a random gene (i.e. a floating number) to another one lying in the searching space.

5.4. Control parameters

The most important control parameters are the following:

Initial population: Usually, the initial population is randomly selected in order to uniformly distribute the selected chromosomes (solutions over the search space). In some cases, solutions obtained from another optimization algorithm are used to seed the initial population.[26] Although this bears the risk of misguiding the optimization process toward local optima, it has been proved that the seeding approach is very powerful in some cases. In our approach, the algorithm 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.[27] The seeding percentage is set to be 10% of the initial population.

Population size: In this work, the proposed global algorithm was tested for various population sizes. Finally, the selected population size is equal to 150 for the case of the 3-DOF manipulator operating in an environment with only one obstacle and equal to 300 for the cases of a 4-DOF manipulator cluttered with 3 and 5 polygonal obstacles.

Crossover/ Mutation rate: The crossover rate is selected equal to 0.75 and the mutation rate is selected equal to 0.035 after a considerable number of trials.

Termination criterion: In most cases, a maximum number of iterations (generations) is defined in advance. However, it is difficult to determine beforehand the number of generations needed to find near-optimum solutions. Thus, an assessment of the quality level of the GA should be made on-line. According to another termination condition, if the same chromosome constantly appears as the optimum one for an a priori defined number of generations, then the run stops. The maximum number of iterations should be large

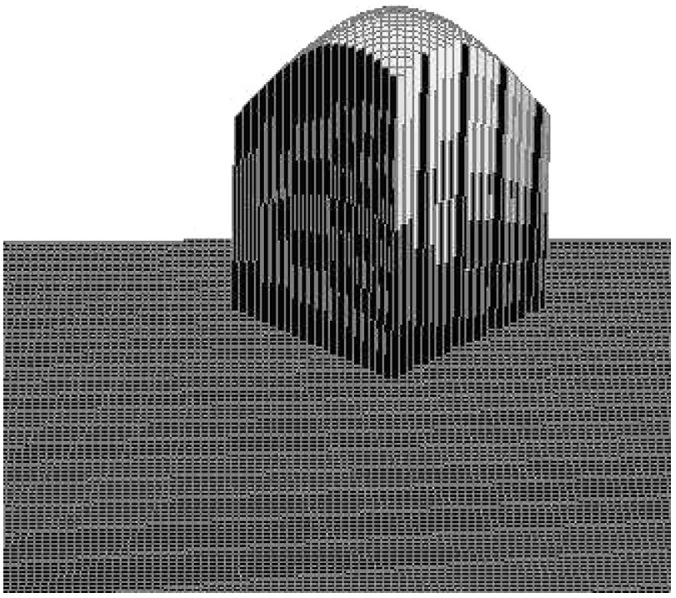


Fig. 4. The B-surface for the workspace of Scene I.

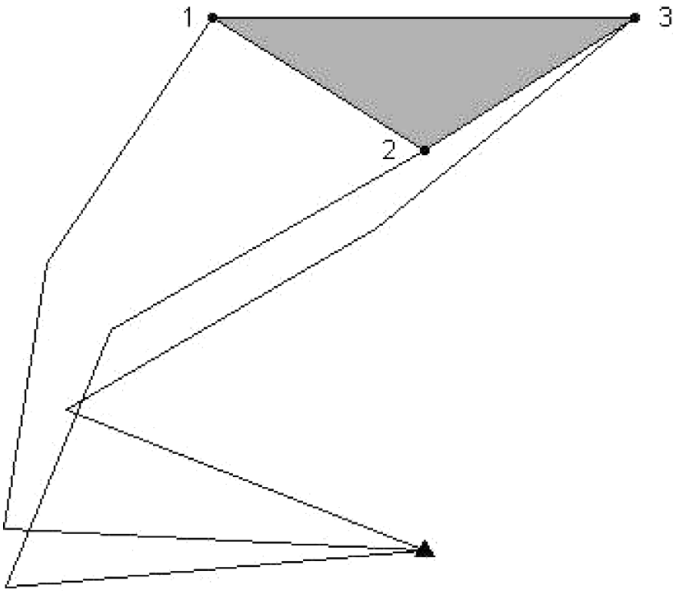


Fig. 5. The robot configurations reaching the three task-points in an environment with one obstacle.

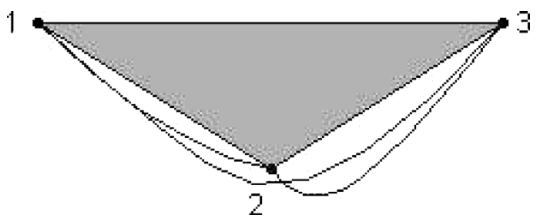


Fig. 6. The robot end-effector's tour 1–2-3–1.

https://doi.org/10.1017/S0263574709005748 Published online by Cambridge University Press

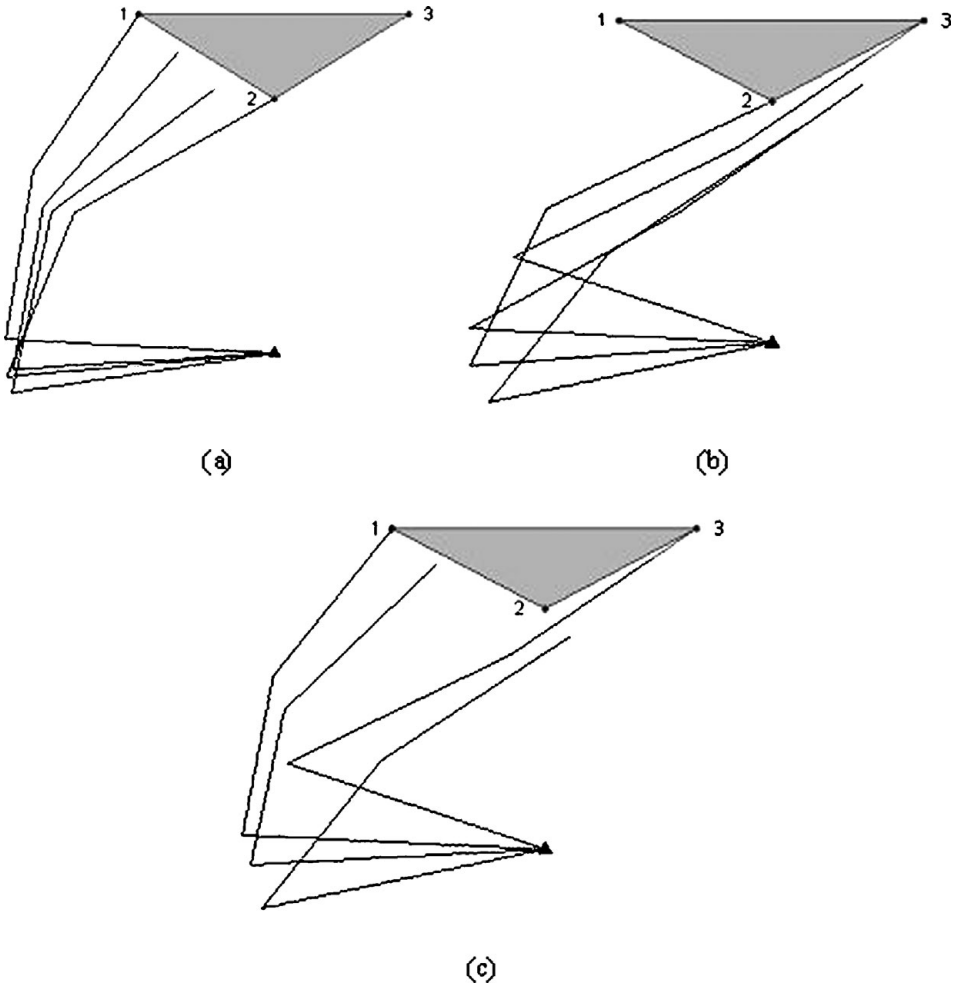


Fig. 7. Two intermediate configurations of the 3-dof manipulator between two successive task-points of the determined tour (a) 1–2 (b) 2–3 (c) 3–1.

enough; otherwise, a misleading result may arise. The proposed algorithm terminates either when the maximum number of generations is achieved or when the same best chromosome appears for a maximum number of generations.

6. Implementation and Results

The proposed method has been implemented on a Pentium IV 3.2 GHz PC using Matlab v7.1. The following examples concern the determination of the optimum sequence of task-points visited by the tip of the end-effector

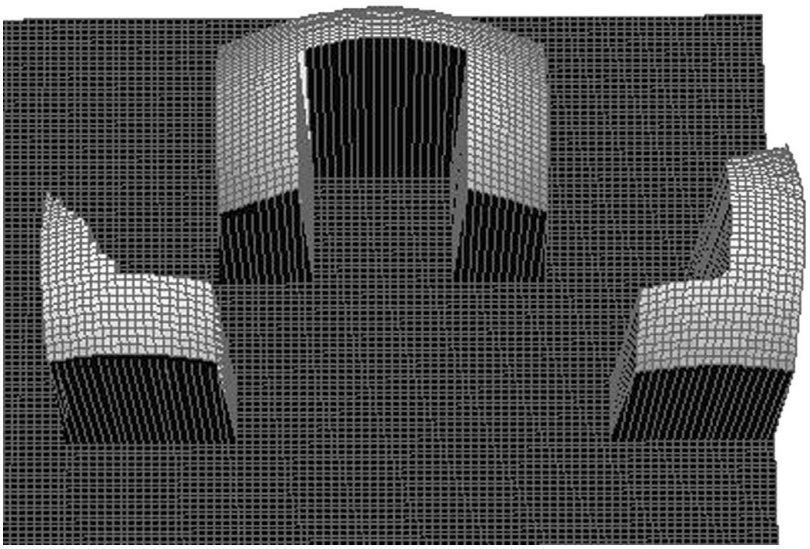


Fig. 8. The B-surface for the workspace of Scene II.

https://doi.org/10.1017/S0263574709005748 Published online by Cambridge University Press

of an articulated robot in 2D complex environments cluttered with static obstacles which have arbitrary size, shape and location. The kinematics of the articulated robots is described in detail Section 3.1. Considering that the layout for the robotic workcell is given in real applications, the robot bases are fixed and known.

Despite the fact that the method is general and thereby applicable to both non-redundant and redundant robots, the experiments are targeted on the solution of the task scheduling problem for redundant robots. The reason for this choice is that the solution for non-redundant robots is easier compared to that for redundant robots, since non-redundant robots have a finite number of configurations at each point and thus, the search space is comparatively small.[8] On the other hand, redundant robots are of immense interest due to the fact that the redundancy in their structure increases the flexibility, the dexterity and the versatility of the robot working in complex environments cluttered with obstacles. Consequently, finding an efficient solution to the problem concerning redundant robots is of greater importance.

6.1. Task scheduling results

In the following, three examples are displayed to validate the feasibility of the proposed algorithm. The experiments are conducted in three different scenes, beginning with a rather simple scene and ending with a much more complex one in the context of more obstacles and a robot of more DOFs. In all cases tested, two intermediate configurations are illustrated between each two successive task-points, so that the results are luminous and distinct. For all tested cases, the robots have the same dimensionless kinematic parameters: link length equal to 2.5 and joint velocity equal to 0.6. It should also be noticed that various combinations of the weight factors have been tried, since the choice of the weight factors have a great influence on the final result.

Scene I: A 3-DOF manipulator has to visit 3 task-points in a 2D workspace, which are the vertices of an obstacle with triangular shape. The robot has links of equal length, and the joint angles lie in the range [0,360) degrees and the joints have the same velocity. Fig. 4 shows the corresponding (2,2)-degree B-Surface. Figure 5 illustrates the solution to the task scheduling problem for the 3-DOF redundant manipulator. As shown, the robot reaches the task-points with high accuracy without colliding with the obstacle. The optimum sequence with which the robot visits the task-points is 1–2-3–1 and the tour is illustrated in Fig. 6. The intermediate configurations between each two successive task-points that ensure free motion for the 3-DOF redundant manipulator are illustrated in Fig. 7.

Scene II: A 4-DOF manipulator has to visit 4 task-points in a 2D workspace cluttered with 3 polygonal obstacles. As before, the robot has links of equal length and the joint angles lie in the range [0,360) degrees. The corresponding B-Surface is illustrated in Fig. 8. Figure 9 illustrates the solution to the task scheduling problem for the 4-DOF redundant manipulator. It is clear that the robot's end-effector touches the exact position of the task-points without colliding with none of the obstacles. The optimum sequence with which the robot visits the task-points is 3–2-4–1-3 and the tour is illustrated in Fig. 10. Figure 11 illustrates the intermediate configurations between each two successive task-points that ensure free motion for the 4-DOF redundant manipulator.

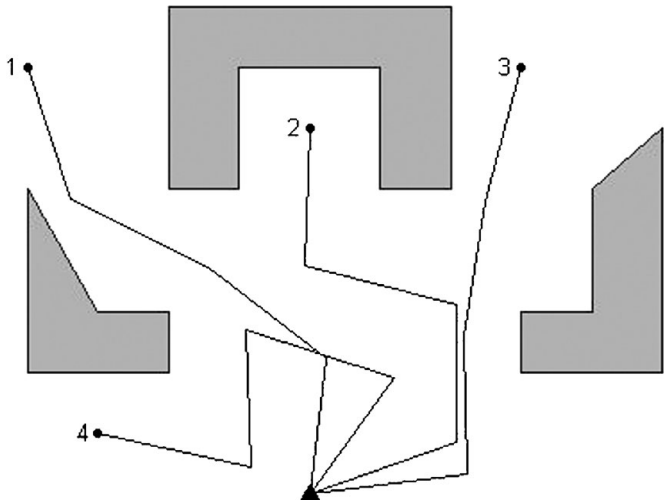


Fig. 9. The robot configurations reaching the four task-points operating in an environment with three obstacles.

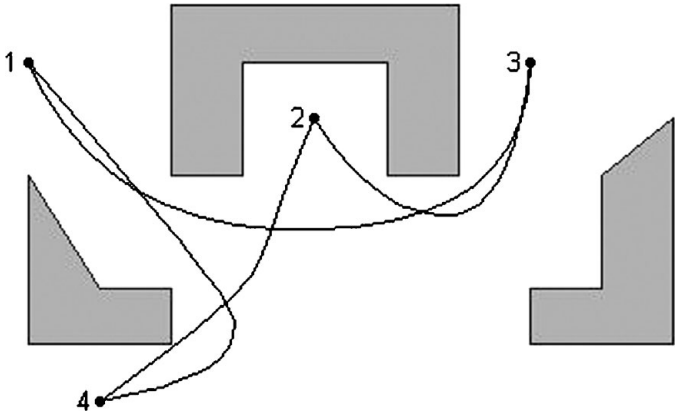


Fig. 10. The robot end-effector's tour 3–2-4–1-3.

One should bear in mind that the optimisation is in terms of time determined in the joint-space and not the distance. Therefore, intuition may delude, since optimisation in time does not necessarily conform to optimisation in distance. Comparing the cycle time corresponding to the resulted optimum sequence 3–2-4–1-3, which is 6.984 s, with the cycle time corresponding to the 'optimum' sequence by intuition 3–2-1–4-3, which is 7.346 s, it is concluded that the cycle time derived from the algorithm is the minimum.

Scene III: A 4-DOF manipulator has to visit 7 task-points in a 2D workspace cluttered with 5 polygonal obstacles. The robot has links of equal length and the joint angles lie in the range [0,360) degrees. Figure 12 shows the corresponding B-Surface. Figure 13 shows the robot configurations corresponding to the task-points, where the robot avoids the obstacles and places its end-effector in acceptable locations between the task-points. The near-optimum

https://doi.org/10.1017/S0263574709005748 Published online by Cambridge University Press

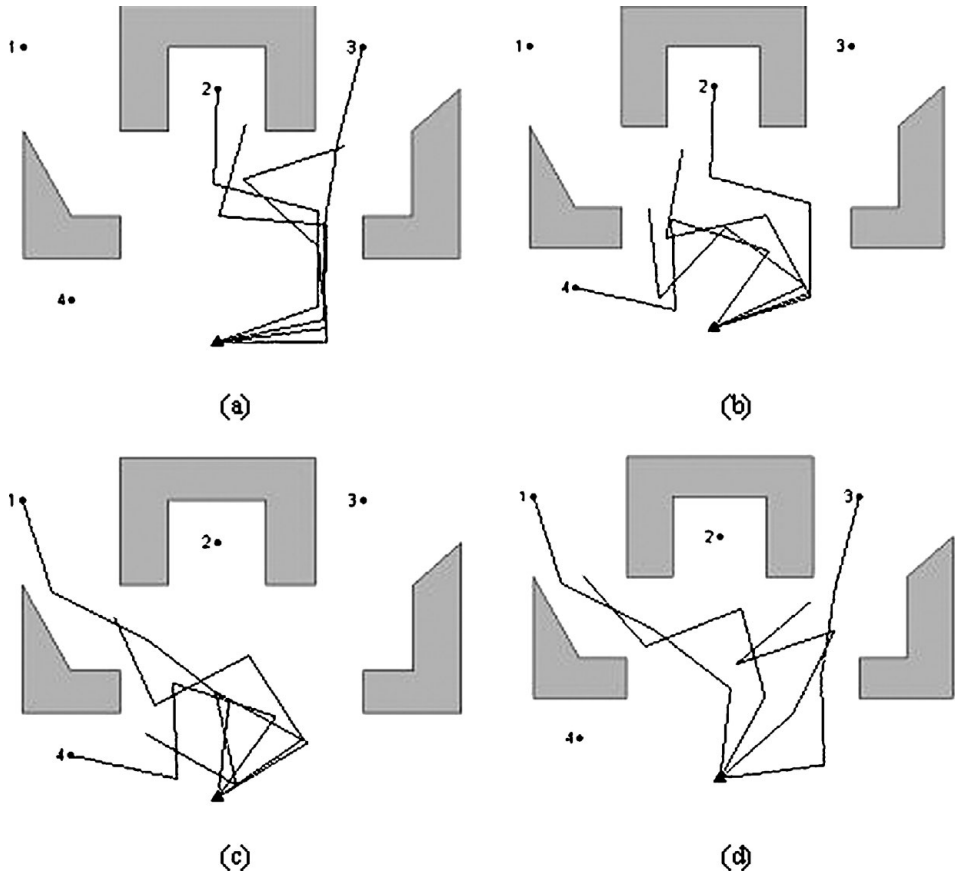


Fig. 11. Two intermediate configurations of the 4-dof manipulator between two successive task-points of the determined tour (a) 3-2 (b) 2–4 (c) 4–1 (c) 1–3.

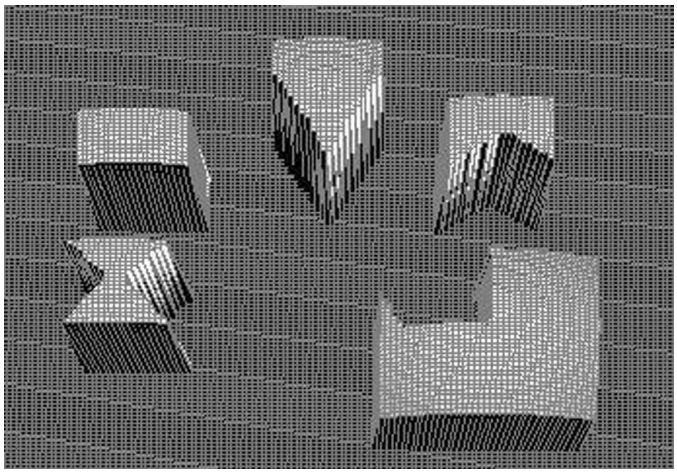


Fig. 12. The B-Surface for the workspace of Scene III.

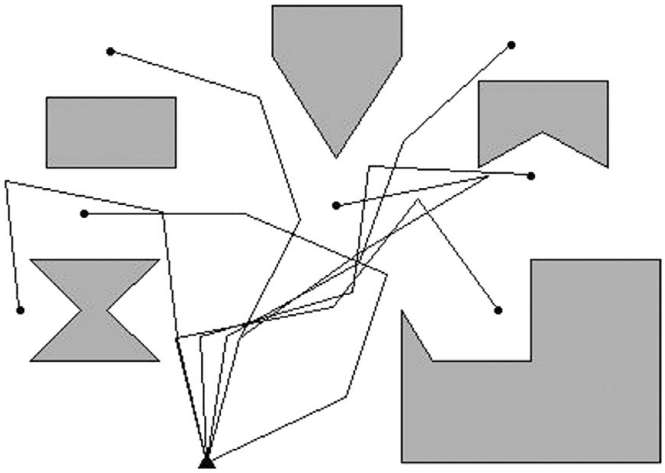


Fig. 13. The robot configurations reaching the seven task-points operating in an environment with five obstacles.

sequence with which the robot visits the task-points is 3–2-1–7-6–5-4–3 and the tour is illustrated in Fig. 14. In contrast to Scene II, the resulted optimum sequence conforms to the 'optimum' sequence by intuition; therefore, there is no point in comparing the resulted sequence with another one.

In the following, the intermediate configurations between task-points 3 and 2 (Fig. 15a), and task-points 2 and 1 (Fig. 15(b)), task-points 1 and 7 (Fig. 15(c)), and task-points 6 and 5 (Fig. 15(d)) are indicatively illustrated. It is obvious from these Figures that the robot moves from task-point to task-point without colliding with the obstacles scattered in the environment.

Scene III is much more complicated compared to Scenes I and II due to the presence of more task-points and obstacles scattered in the environment. The task scheduling problem is undoubtedly hard to be solved for Scene III due to

https://doi.org/10.1017/S0263574709005748 Published online by Cambridge University Press

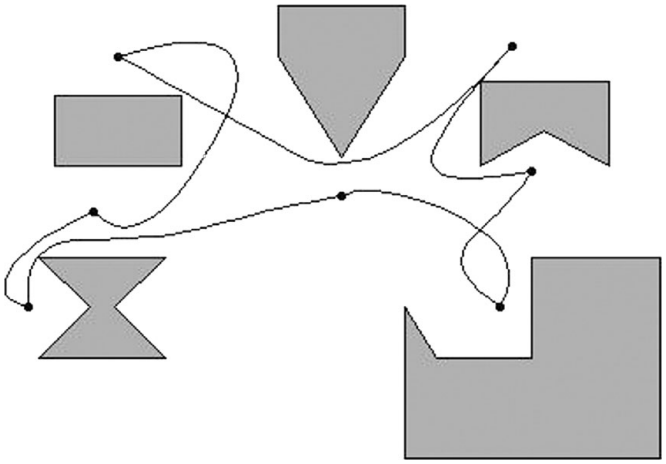


Fig. 14. The robot end-effector's tour 3–2-1–7-6–5-4–3.

the complexity of the scene. The computational time has increased and the robot's end-effector reaches the task-points with a non-zero maximum position error, though within an acceptable tolerance. It is obvious that a higher number of the task-points and the robot's joints (i.e. the DOF of the manipulator) increases the computational time needed for yielding a near-optimum solution.

6.2. Computational time results

Despite the fact that the problem under consideration is off-line, the dependence of the computational time on both the number of task-points and the DOF is of immense interest. Besides, the variation of CPU time is indicative of the problem complexity. An interesting indication of the computational time performance is the variation of CPU time versus the number of the task-points, which is investigated experimentally. In these experiments, a 3-DOF manipulator operates in an environment with three obstacles (see scene II) and is assigned to reach $\{3, 4, 5, 6, \ldots, 12, 13\}$ task-points ensuring the minimum cycle time without colliding with the obstacles. Figure 16 depicts the computational time versus the number of task-points, where it seems that it increases almost linearly with the increase in the number of task-points.

To evaluate the influence of the increase in the number of the DOFs of the robot over the computational time, several experiments were carried out. Firstly, a 3-DOF manipulator operates in an environment with three obstacles and is assigned to reach four task-points (see scene II) ensuring the minimum cycle time without colliding with the obstacles. Then, a 4-DOF manipulator is assigned to execute the same task in the same environment. The experiment is iterated for a $\{4, 5, 6, 7, 8, 9\}-$DOF manipulator. Figure 17 depicts the computational time versus the number of DOF, where it seems that the CPU time does not increase at a great rate versus the number of DOF.

Experiments are conducted for the manipulator operating in the environment described in Scene III and the results are depicted in Fig. 18 and Fig. 19. Although the CPU time is increased compared to that for Scene III, the increase versus number of task-points and the number of DOF is almost linear. The conclusions established from the CPU time results for both Scene II and III confirm that the use of the B-surface for obstacle avoidance reduces the search space complexity.

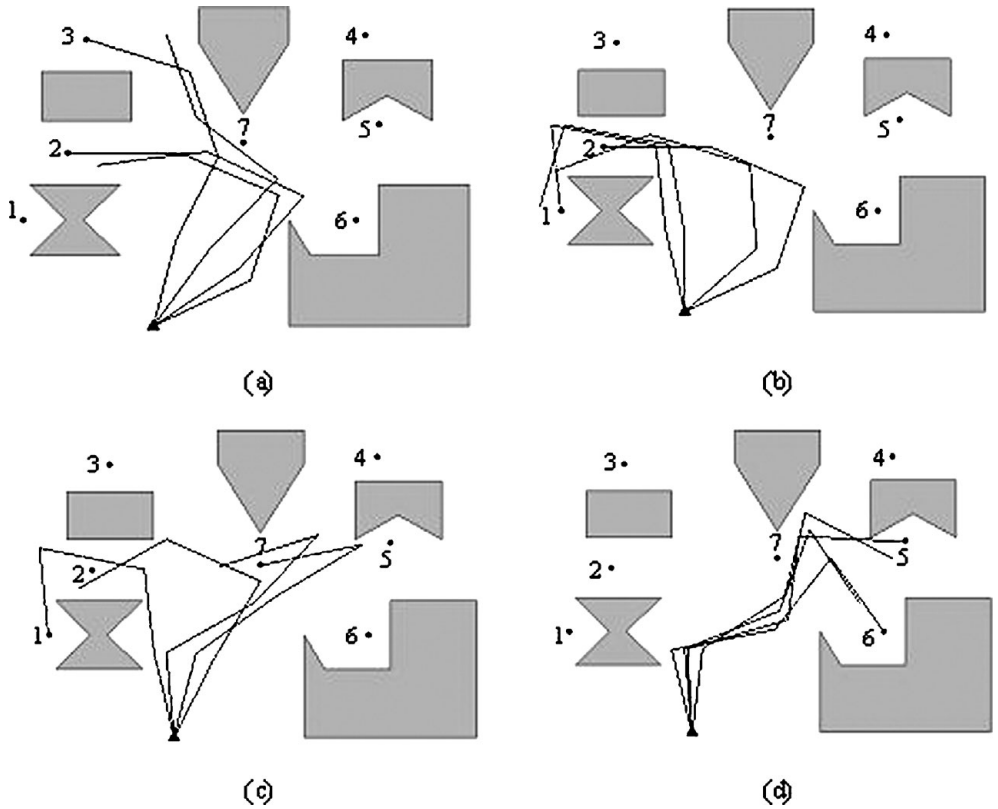


Fig. 15. The two intermediate configurations of the 4-dof manipulator between the task-point (a) 3–2 (b) 2–1 (c) 1–7 (d) 6–5.

https://doi.org/10.1017/S0263574709005748 Published online by Cambridge University Press

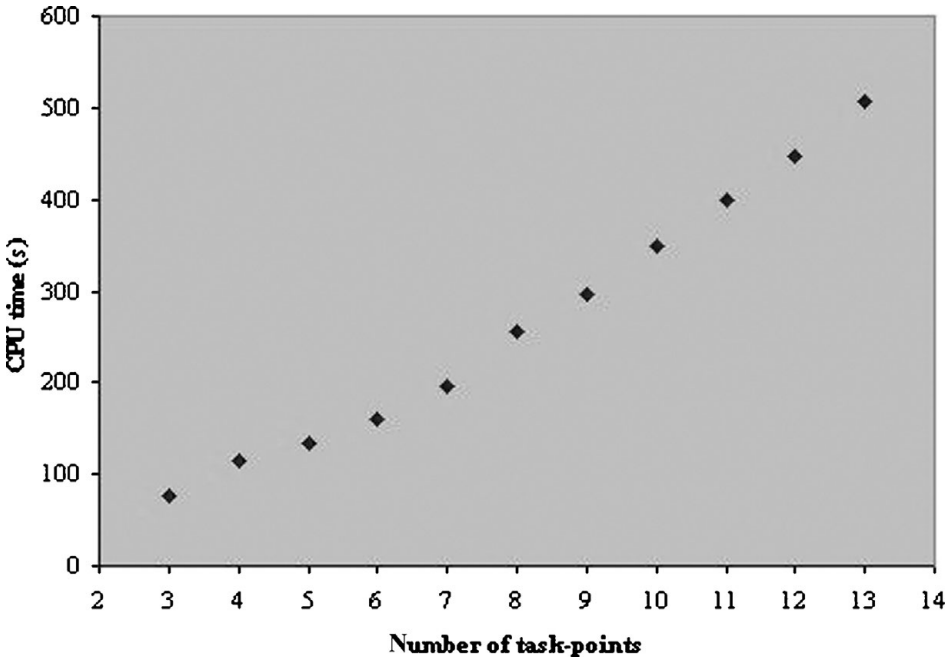


Fig. 16. The computational time versus the number of task-points for Scene II.

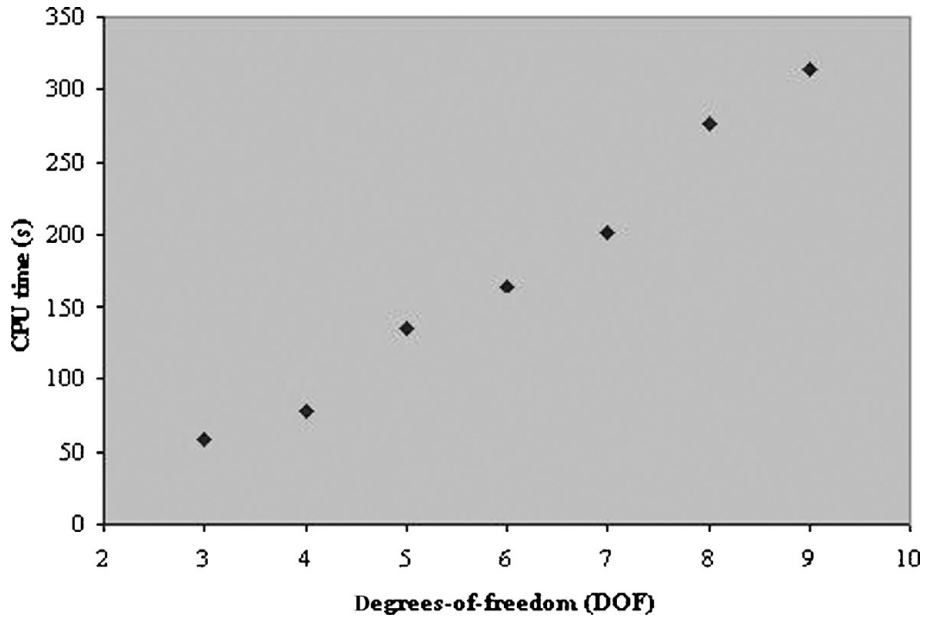


Fig. 17. The computational time versus the DOF for Scene II.

7. Conclusions

A method based on the B-surface concept is introduced to solve the task scheduling problem for articulated robots in environments cluttered with static obstacles, where a set of task-points is located in the free workspace. The main advantage of this method is that it can be applied to any articulated (redundant or non-redundant) robot operating in an environment cluttered with obstacles, despite the fact that the joint redundancy complicates the problem. In addition, using the B-Surface concept, the complexity of the problem does not depend on the complexity of the environment, e.g. the shape and the location of the obstacles.

The adopted GA approach is an effective alternative to the solution of the Inverse Kinematics Problem, which is rather complicated especially for the case of redundant robots. In addition, a variety of criteria and constraints can be included in the objective function taking into consideration several factors that affect the solution quality, like the smoothness of the path followed by the end-effector's tip and the width of the robot links.

The experimental results show that the method is effective and efficient for determining a near-optimum solution in complicated environments cluttered with obstacles of arbitrary shape, size and location. The convergence of

https://doi.org/10.1017/S0263574709005748 Published online by Cambridge University Press

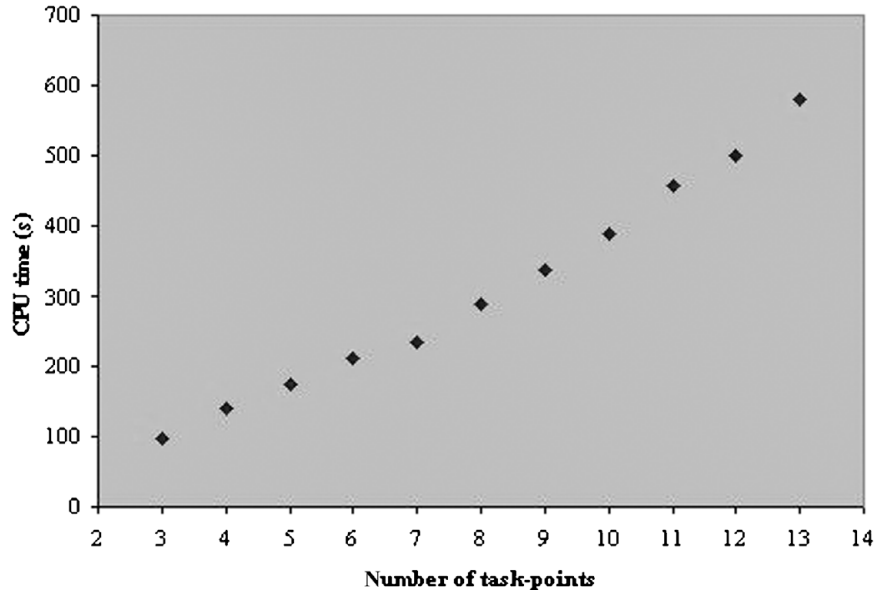


Fig. 18. The computational time versus the number of task-points for Scene III.

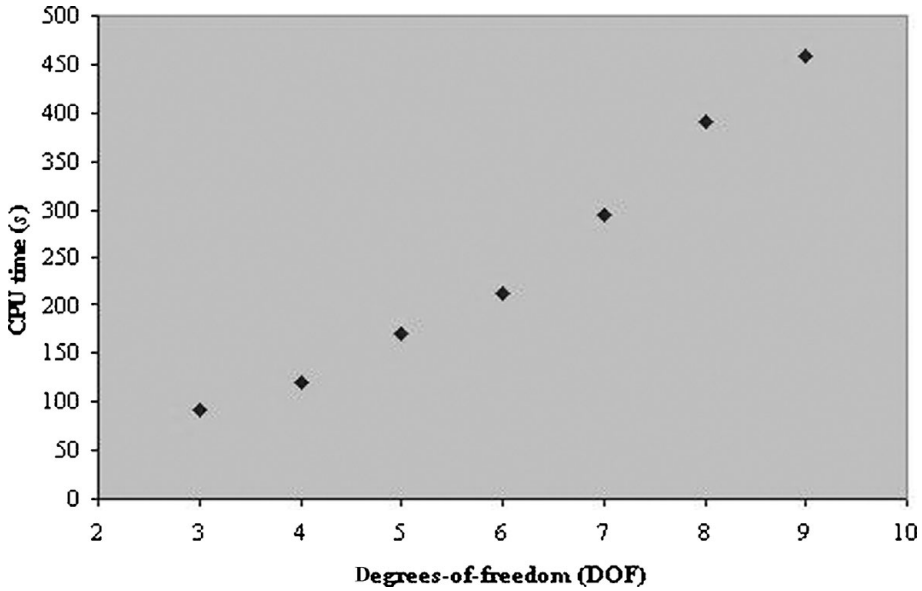


Fig. 19. The computational time versus the DOF for Scene III.

the algorithm depends on the number of task-points, the redundancy in robot's joints (i.e. the redundant DOFs) and the operators of the GA. For all the tested cases, the final solution is proved to be conformal to the conditions (a) and (b). Furthermore, the proposed method proved to be efficient in finding a near-optimum solution within an affordable time.

Considering the future research work, the method could be applied to 3D environments using 3D robots considering their links as solid bodies. It could also be extended to environments cluttered with both static and moving obstacles as well as dynamic task-points.

Acknowledgements

This work is carried out under the project 'XPOMA-Handling of non-rigid materials with robots: application in robotic sewing' PENED 01 funded by the General Secretariat for Research and Technology of Greek Government and it is integrated in the IPROMS Network of Excellence.

References

1. E. Lawer, J. Lenstra, A. R. Kan and Shmoys D., *The Travelling Salesman Problem* (Wiley, Chichester, UK, 1985).
2. J. C. Latombe, *Robot Motion Planning* (Kluwer Academic Publishers, Boston, 1991).
3. M. R. Garey, R. L. Graham and D. S. Johnson. "Some NP-Complete Geometric Problems," *Proceedings of ACM Symposium on Theory of Computing*, Hershey, Pennsylvania, United States (1976) pp. 10–22.
4. C. H. Papadimitriou, "Euclidean TSP is NP-complete," *TCS* **4**, 237–244 (1977).
5. L. Abdel-Malek and Z. Li, "The application of inverse kinematics in the optimum sequencing of robot tasks," *Int. J. Prod. Res.* **28**(1), 75–90 (1990).
6. J. Petiot, P. Chedmail and J.-Y. Haschoët, "Contribution to the scheduling of trajectories in robotics," *Robot. Cim-Int. Manuf.* **14**(3), 237–251 (1998).

https://doi.org/10.1017/S0263574709005748 Published online by Cambridge University Press

7. S. Dubowsky and T. Blubaugh, "Planning time-optimal robotic manipulator motions and work places for point-to-point tasks," *IEEE Trans. Robot Autom.* **5**(3), 377–381 (1989).
8. P. T. Zacharia and N. A. Aspragathos, "Optimal robot task scheduling based on genetic algorithms," *Robot. Cim-Int. Manuf.* **21**(1), 67–79 (2005).
9. N. Rahmanian-Shahri and I. Torch, "Collision avoidance control for redundant articulated robots," *Robotica* **13**(2), 159–168 (1995).
10. R. V. Mayorga, F. Jambi Sharifi and A. K. C. Wong, "A fast approach for the robust trajectory planning of redundant robot manipulators," *J. Robot. Sys.* **12**(2), 147–161 (1995).
11. J.-L. Chen, J.-S. Liu, W.-C. Lee and T.-C. Liang, "On-line multi-criteria based on collision-free posture generation of redundant manipulator in constraint workspace," *Robotica* **20**(6), 625–636 (2002).
12. T. Lozano–Perez, "Spatial planning: A configuration space approach," *IEEE T. Comput.* **C-32**(2), 108–120 (1983).
13. J. Barraquand, B. Langlois and J.-C. Latombe, "Numerical potential field techniques for robot path planning," *IEEE T. Syst. Man Cy.* **22**(2), 224–241 (1992).
14. E. S. Conkur, "Path following algorithm for highly redundant manipulators," *Robot Auton. Syst.* **45**(1), 1–22 (2003).
15. D. E. Goldberg, *Genetic Algorithm in Search, Optimization and Machine Learning* (Addison Wesley, Reading, MA, 1989).
16. Z. Michalewitz, *Genetic Algorithms + Data Structures = Evolution Programs*, 3rd ed. (Springer-Verlag, New York, 1996).
17. T.-C. Liang and J.-S. Liu, "An improved trajectory planner for redundant manipulators in constrained workspace," *J. Robot. Syst.* **16**(6), 339–351 (1999).
18. E. S. Conkur, "Path planning using potential fields for highly redundant manipulators," *Robot Auton. Syst.* **52**(2–3), 209–228 (2005).
19. A. C. Nearchou and N. A. Aspragathos, "A genetic path planning algorithm for redundant articulated robots," *Robotica*. **15**(2), 213–224 (1997).
20. L. Tian and C. Collins, "Motion planning for redundant manipulators using a floating point genetic algorithm," *J. Intell. Robot. Syst.* **38**(3–4), 297–312 (2003).
21. P. Azariadis and N. Aspragathos, "Obstacle representation by bump-surfaces for optimal motion-planning," *Robot Auton. Syst.* **51**(2–3), 129–150 (2005).
22. L. Piegl and W. Tiller, *The NURBS Book* (Springer-Verlag, Berlin, Germany, 1997).
23. E. K. Xidias, P. N. Azariadis and N. A. Aspragathos, "Energy-minimizing motion design for nonholonomic robots amidst moving obstacles," *Comput. Aided Des. Appl.* **3**(1–4), 165–174 (2006).
24. E. K. Xidias, A. C. Nearchou and Nikos A. Aspragathos, "Routing and Scheduling for a Car-Like Robot in 2D Manufacturing Environment with Time Windows," *Ind. Robot.* **36**(2), 176–183 (2009).
25. J. H. Holland, *Adaptation in Natural and Artificial Systems* (The University of Michigan Press, Ann Arbor, MI; 1975).
26. R. Thomsen, G. Fogel and T. Krink, "A Clustal Alignment Improver using Evolutionary Algorithms," *Proceedings of the 2002 Congress on Evolutionary Computation—CEC'02*, Honolulu, HI, USA (May 12–May 17 2002) pp. 309–314.
27. L. Davis, "Applying Adaptive Algorithms to Epistatic Domains," *Proceedings of the International Joint Conference on Artificial Intelligence*, Los Angeles, Calif (August 1985) pp. 162–164.

https://doi.org/10.1017/S0263574709005748 Published online by Cambridge University Press