International Journal of Automation and Computing 7(2), May 2010, 190-198
DOI: 10.1007/s11633-010-0190-8

Evolutionary Trajectory Planning for an Industrial Robot

R. Saravanan[1] S. Ramabalan[2] C. Balamurugan[3] A. Subash[2]

[1]Department of Mechanical Engineering, Bannariamman Institute of Technology, Sathiamangalam, Tamil Nadu 638401, India
[2]Department of Mechanical Engineering, J. J. College of Engineering and Technology, Thiruchirapalli, Tamilnadu 620009, India
[3]Department of Production Engineering, J. J. College of Engineering and Technology, Thiruchirapalli, Tamilnadu 620009, India

Abstract: This paper presents a novel general method for computing optimal motions of an industrial robot manipulator (AdeptOne XL robot) in the presence of fixed and oscillating obstacles. The optimization model considers the nonlinear manipulator dynamics, actuator constraints, joint limits, and obstacle avoidance. The problem has 6 objective functions, 88 variables, and 21 constraints. Two evolutionary algorithms, namely, elitist non-dominated sorting genetic algorithm (NSGA-II) and multi-objective differential evolution (MODE), have been used for the optimization. Two methods (normalized weighting objective functions and average fitness factor) are used to select the best solution tradeoffs. Two multi-objective performance measures, namely solution spread measure and ratio of non-dominated individuals, are used to evaluate the Pareto optimal fronts. Two multi-objective performance measures, namely, optimizer overhead and algorithm effort, are used to find the computational effort of the optimization algorithm. The trajectories are defined by B-spline functions. The results obtained from NSGA-II and MODE are compared and analyzed.

Keywords: Multi-objective optimal trajectory planning, oscillating obstacles, elitist non-dominated sorting genetic algorithm (NSGA-II), multi-objective differential evolution (MODE), multi-objective performance metrics.

1 Introduction

Many real-world design tasks involve complex multi-objective optimization problems of various competing design specifications and constraints. For such problems, it is highly improbable that all the conflicting criteria would be extremized by a single design, and hence, a trade-off among the conflicting design objectives is often inevitable. In mathematics, multi-objective optimisation seeks to optimise a vector of non-commensurable and often competing objectives, cost functions or performance functions within a feasible decision variable space. Intelligent robot system design is one of the complex design problems.

The ultimate requirement of robotics is to create intelligent robotic systems that can operate autonomously. At present, robots are used to perform programmed, repetitious tasks or tasks where a human operator has to constantly specify motions. In case of autonomous robots, the robot is provided with only descriptions of tasks on an abstract level and will carry out those tasks without human intervention or explicit teaching. In order to reach that goal, more development should take place in technologies of perception, which involves automated reasoning, planning, manipulation, and learning. One of the main planning problems is the trajectory planning, where the autonomous robot has to plan its own motions, and, by virtue of this motion, only the robot accomplishes its tasks. The classic trajectory-planning problem is described as follows: given an initial configuration and a final configuration of the robot, it has to find a path connecting both configurations that avoids collision with obstacles. Assumptions are that the geometry and the position of obstacles are known in advance, and obstacles are stationary.

In order to maximize speed of operation, which affects the productivity in industrial situations, it is necessary to minimize the total travelling time of the robot. Therefore, more research works have been carried out to get minimum time trajectories[1−4].

The robot trajectory planning using energetic criteria provides several advantages. It yields smooth trajectories for easier tracking and reduces the stresses to the actuators and to the manipulator structure. Moreover, saving energy may be desirable in several applications, such as those with a limited capacity of energy source (e.g., robots for space or underwater exploration). Examples of energy optimal trajectory planning are provided in some literatures. Both optimal travelling time and minimum mechanical energy of the actuators are considered together as objective functions in some literatures[5,6].

Fields of research such as computer graphics, geometric design, and robotics (motion planning) prefer smooth trajectories, which are achieved by minimizing the joints jerk[7, 8] and joints acceleration[9]. Gasparetto et al.[7, 8] considered only kinematic constraints. They did not consider dynamic constraints such as joint torques. The conventional method (numerical iterative procedure) used by Elnager and Hussein[9] cannot be used for multi-objective problems.

To obtain a practical trajectory (such that the robot does not loose any degree of freedom at any stage), the manipulability measure can be used as the decision criteria for robot trajectory planning[10,11]. So in order to get all the above benefits, all the objective functions have to be considered in a combined manner to do optimal trajectory planning. But none of literatures considered all these objective functions in a combined manner.

Many authors have treated the problem of trajectory planning of robot manipulators in the presence of fixed obstacles[1,12]. When moving obstacles share the same workspace occupied by the robot manipulator, the optimisation of the trajectory defined by the end-effector is complex[1,5,6]. This complexity is associated with the large number of constraints to be taken into account by the op-

Manuscript received June 17, 2008; revised July 10, 2009

timiser. These constraints are time dependent in this case. A design methodology using sequential unconstrained minimisation techniques is proposed by Saramago and Junior[5] to obtain the optimal off-line trajectory planning of robot manipulators, when oscillating obstacles have to be avoided by the end-effector. Their problem of optimal trajectory planning concerns with the determination of the end-effector robot motion in a minimum time and minimum mechanical energy between two given points, while satisfying the limits of the actuator efforts and avoiding collision with oscillating and fixed obstacles.

The methods that are used in the literatures[1−3,5,6,9−14] to tackle the complex instances (oscillating obstacles environment) have some notable drawbacks: 1) they may fail to find the optimal path (or spend a lot of time and memory storage); and 2) they have limited capabilities when handling cases where the constraints of maximum acceleration and maximum deceleration along the solution curve are no longer met, or where singular points or critical points of robot configuration exist. To overcome the above drawbacks, evolutionary algorithms can be used. The advantages of evolutionary techniques are follows: 1) they are population-based search algorithms, so global optimal solution is possible; 2) they do not need any auxiliary information like gradients, derivatives, etc; 3) they can solve complex and multimodal problems for global optimality; and 4) they are problem independent, i.e., suitable for all types of problems. Evolutionary techniques for multi-objective optimisation are currently gaining significant attention from researchers in various fields due to their effectiveness and robustness in searching for a set of trade-off solutions. Unlike conventional methods that aggregate multiple attributes to form a composite scalar objective function, evolutionary algorithms with modified reproduction schemes for multi-objective optimisation are capable of treating each objective component separately and lead the search in discovering the best trade-off solutions.

The motivations for using population-based search techniques are as follows:

1) The final solution obtained from a conventional mathematical optimization technique is always dependent on the input (i.e., initial solution). The wrong selection of initial solution may lead to local optimal solution. But in case of population-based techniques, the final solution will not depend on the initial solution. So, the final solution may be a global optimal solution.

2) Population-based search techniques (e.g., evolutionary algorithms (EAs)) give multi directional search. They deal simultaneously with a set of possible solutions (the so-called population). This allows us to find several possible solutions in a single run of the algorithm, instead of performing a series of separate runs as in the case of traditional mathematical programming techniques.

3) Optimizing all the objectives simultaneously and generating a set of alternative solutions offers more flexibility to decision makers. The simultaneous optimization can fit nicely with population-based approaches such as EAs because they generate multiple solutions in a single run.

4) When compared to conventional and mathematical techniques available in literature[1−12], evolutionary algorithms converge quickly and give more number of Pareto optimal solutions.

5) Computational efficiency of evolutionary algorithms is better than those of the conventional and mathematical techniques available in literature[1−12].

6) Significant computational speed-up could be achieved, because their running time is shorter than those of conventional and mathematical techniques available in literature[1−12].

Intelligent optimization algorithms such as non-dominated sorting genetic algorithm (NSGA-II) and multi-objective differential evolution (MODE) are very much desirable for trajectory planning of an intelligent real world robot. Trajectory planning for a real world robot is a very complex and tedious task, due to the following reasons:

1) The planning algorithm has to consider the dynamic model of the robot, which is depending on travelling time, payload, and robot′s task. So the planning algorithm is a time-dependent one.

2) In robot′s workspace, all types of obstacles (fixed, moving, and oscillating obstacles) may be present. This calls for the planning algorithm to consider all types of obstacles for obstacle avoidance. Further, the information about the obstacles may be partially or fully unknown. Therefore, checking for the presence of obstacles collision with robot is a very complex and time dependent task.

3) The environment around the robot is an ever-changing one. This calls for planning algorithm to update the details for trajectory planning for each time instant.

This paper considers all the decision criteria for the optimal trajectory planning of industrial robot manipulators and the obstacle avoidance criteria for oscillating obstacles. In this paper, two evolutionary algorithms, namely, NSGA-II and MODE, are proposed to obtain optimal trajectory planning for an industrial robot (AdeptOne XL robot). Two methods, namely normalized weighting objective functions and average fitness factor, are used to select the best solution tradeoffs. Two multi-objective performance measures, namely solution spread measure and ratio of non-dominated individuals, are used to evaluate the Pareto optimal fronts. Two multi-objective performance measures, namely optimizer overhead and algorithm effort, are used to find the computational effort of the optimisation algorithm. These methods and metric are chosen since they have been widely used for performance comparisons in multi-objective optimisation[15].

A numerical application related to an industrial robot manipulator (AdeptOne XL) is presented to illustrate the methodology developed in this paper. All the decision criteria of the optimal trajectory planning of industrial robot manipulators, namely, minimal time, minimum mechanical energy of the actuators, collision-free motion, maximization of manipulability measure, minimum accelerations and minimum Jerks, are considered together in this work. The obstacles are considered as objects sharing the same workspace occupied by the robot. The obstacle avoidance is expressed in terms of the distances between potentially colliding parts, and the motion is represented using translation and rotational matrices. The dynamic model of the robot is derived using Euler-Lagrange′s equations and Lagrange′s energy function. The inertia terms of the actuators and friction forces are included in the equations of motion. The joint trajectory is formulated using uniform cubic B-spline function, given only the initial and final configurations. When obstacles are found in the three-dimensional workspace, it is necessary to add penalty functions to the multi-objective problem to guarantee free-collision motion. The obstacles are protected by spherical or hyper-spherical security zones, which would never be penetrated by the end-effector.

This paper is organized as follows. Section 2 presents the problem statement of this paper. In Section 3, a numerical example is presented to illustrate the proposed optimisation methodology. Implementation of NSGA-II and MODE algorithms is given in Section 4. In Section 5, the results obtained from NSGA-II and MODE are presented and compared. The conclusions are presented in Section 6.

Notations.

z_1: Travelling time of the robot.

z_2: Quadratic average of actuator torques.

$z_3 = f_{\rm dis}$: Penalty parameter for free-collision motion.

z_4: Manipulability measure.

z_5: Integral of squared robot joint jerks.

z_6: Integral of squared robot joint accelerations.

i: Robot joint number.

u_i: The generalized forces.

J: Jacobian matrix of the robot.

$Q = q_j^i(t) = q_{ji}(t)$: Displacement of robot joint i at time t.

$V = \frac{{\rm d}(q_j^i(t))}{{\rm d}t} = \dot{q}_{ji}(t) = \dot{q}$: Velocity of robot joint i at time t.

$W = \frac{{\rm d}^2(q_j^i(t))}{{\rm d}t^2} = \ddot{q}_{ji}(t) = \ddot{q}$: Acceleration of robot joint i at time t.

$\dddot{q}_{ji}(t) = \dddot{q}$: Jerk of robot joint i at time t.

J: Jacobian matrix of the robot.

QC_j: Maximum displacement of robot joint i.

VC_j: Maximum velocity of robot joint i.

WC_j: Maximum acceleration of robot joint i.

JC_j: Maximum jerk of robot joint i.

UC_j: Maximum force/torque of robot joint i.

n: Maximum number of the robot joints.

m: Maximum number of the knots used to construct the trajectories.

$d_{lq}(t)$: Distance between the robot $d_l(t)$ and the obstacle $d_q(t)$ at time t.

I_d: The set of possibly colliding pairs of parts.

D_{ij}: The inertial system matrix.

C_{ijk}: The coriolis and centripetal forces matrix.

G_i: The gravity-loading vector.

ff_c: The coulomb force coefficient.

f_d: The viscous damping coefficient.

γ_j^i: The coefficients of the B-spline approximation for $q_{ji}(t)$ in the interval "I_j".

(X_0, Y_0, Z_0): The centre of an obstacle.

r_0: The radius of the sphere that circumscribes this obstacle.

r_t: The distance between the centre of the obstacle and a trajectory point.

$nobs$: The total number of obstacles in the workspace.

r_e: The eccentricity.

f_c: Combined objective function.

$\dot{q}_1$: Velocity of robot joints at starting point.

$\dot{q_m}$: Velocity of robot joints at final point.

$\ddot{q}_1$: Acceleration of robot joints at starting point.

$\ddot{q_m}$: Acceleration of robot joints at final point.

2 Problem statement

An industrial robot manipulator (AdeptOne XL robot) with 4 degrees of freedom is considered. The target is to move the robot in a workspace avoiding the fixed and oscillating obstacles, while minimizing travelling time, mechanical energy of the actuators, joint jerks, joint accelerations, penalty function to guarantee collision-free motion, and maximizing manipulability measure of the robot taking into account the physical constraints, actuator limits, and obstacle avoidance. This problem has 6 objective functions, 21 constraints, and 88 variables (polynomial coefficients γ_j^i of B-spline functions that represent the trajectories).

The singularity avoidance is chosen in the form of the manipulability measure. Maximizing the manipulability measure will force the manipulator away from the singularity. The multicriterion optimisation problem is defined as follows:

Minimize total travelling time between initial and final configurations $= z_1$. (1)

Minimize quadratic average of actuator torques:

$$z_2 = \int_0^T \sum_{i=1}^n (u_i(t))^2 {\rm d}t. \tag{2}$$

Minimize penalty parameter for collision-free motion:

$$z_3 = f_{\rm dis}. \tag{3}$$

Maximize manipulability measure:

$$z_4 = |\det(J)|. \tag{4}$$

Minimize integral of squared joint jerks:

$$z_5 = \int_0^T \sum_{i=1}^n (\dddot{q}_i^2) {\rm d}t. \tag{5}$$

Minimize integral of squared joint accelerations:

$$z_6 = \int_0^T \sum_{i=1}^n (\ddot{q}_i^2) {\rm d}t. \tag{6}$$

Subject to

1) Displacement constraint:

$$\max |q_{ji}(t)| \leqslant QC_j. \tag{7}$$

2) Velocity constraint:

$$\max |\dot{q}_{ji}(t)| \leqslant VC_j. \tag{8}$$

3) Acceleration constraint:

$$\max |\ddot{q}_{ji}(t)| \leqslant WC_j. \tag{9}$$

4) Jerk constraint:

$$\max |J_{ji}(t)| \leqslant JC_j. \tag{10}$$

5) Force/torque constraint:

$$\max |u_{ij}(t)| \leqslant UC_j \quad \text{for } j = 1, 2, \cdots, n \text{ and } i = 1, 2, \cdots, m-1. \tag{11}$$

6) Obstacle avoidance constraint:

$$d_{lq}(t) > 0 \quad \text{for } (l, q) \in I_d. \tag{12}$$

2.1 Obstacle avoidance

The distance between potentially colliding parts is expressed as measure of obstacle avoidance. Further, the motion is represented using translation and rotational matrices. When obstacles are found in the workspace, it is necessary to add a penalty function in the multicriterion problem to guarantee collision-free motion. The idea is to circumscribe each obstacle into a specific sphere (see Fig. 1). The trajectory points (X, Y, Z), which are located outside the sphere, are accepted according to the following equation:

$$r_t = \sqrt{(x-x_0)^2 + (y-y_0)^2 + (z-z_0)^2} > r_0. \tag{13}$$

If (13) is verified, the trajectory is out of the sphere, and the penalty function (f_{dis}) is zero. If the trajectory is tangent or crossing the sphere, a penalty will be added to the multicriterion problem.

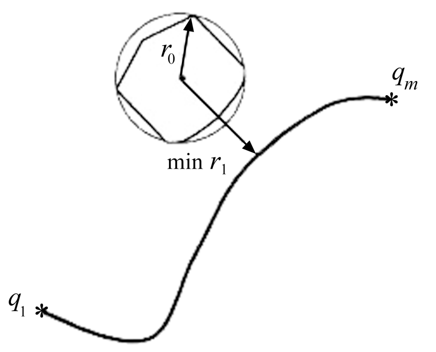


Fig. 1 Obstacle circumscribed by a sphere

$$f_{\text{dis}} = \begin{cases} 0, & \text{if } r_t > r_0 \\ \sum\limits_{i=1}^{nobs} \dfrac{1}{(\min r_i)^2}, & \text{if } r_t \leqslant r_0. \end{cases} \tag{14}$$

There are situations that need to consider the topology of the obstacle. So it is more likely to circumscribe the obstacle by an ellipsoid as shown in Fig. 2.

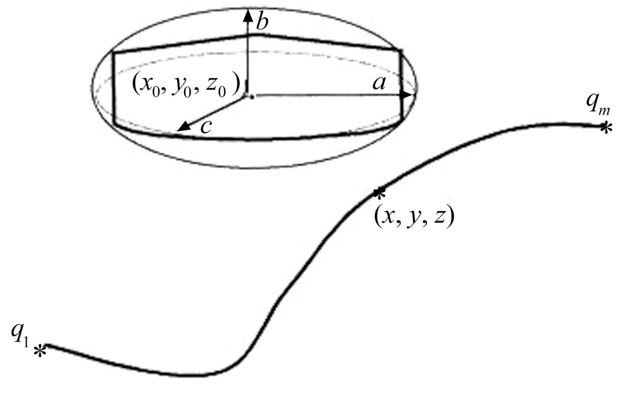


Fig. 2 Obstacle circumscribed by an ellipsoid

Let a, b, and c be the semi-axes of the circumscribing ellipsoids and applying the same principle used for the circumscribing spheres. The trajectory points, which are located outside the ellipsoid, are accepted according to the equation:

$$r_e = \frac{(x-x_0)^2}{a^2} + \frac{(y-y_0)^2}{b^2} + \frac{(z-z_0)^2}{c^2}. \tag{15}$$

Penalization used in this case is below:

$$f_{\text{dis}} = \begin{cases} 0, & \text{if } r_e > 1 \\ \sum\limits_{i=1}^{nobs} \dfrac{1}{(\min r_e)^2}, & \text{if } r_e \leqslant 1. \end{cases} \tag{16}$$

In this way, the optimal control problem is to bring the penalty function given by (14) or (16) to zero by taking into account the kinematic, dynamic, and obstacle avoidance constraints. Besides, constraints that describe minimal acceptable distance between potentially colliding parts are also included in the general non-linear optimization problem. The obstacles are protected by spherical or hyperspherical security zones, which would never be penetrated by the end-effector.

Checking obstacle avoidance criterion is a time-consuming one. But in our case, it is not so due to the following reasons:

1) The evolutionary algorithms give multi directional search. They deal simultaneously with a set of possible solutions (the so-called population). This allows us to find several possible solutions in a single run of the algorithm, instead of having to perform a series of separate runs as in the case of the traditional mathematical programming techniques.

2) The method used for checking obstacle avoidance in this paper is a better one, and it could considerably tackle any complex problem (e.g., industrial environment with moving obstacles).

3) At present, due to the availability of very good computational facilities such as high-speed computing, user-friendly environment, virtual reality software, and efficient programs of evolutionary algorithms, the checking of the presence of obstacles can be done in a few seconds when compared to conventional optimization techniques.

The geometry of AdeptOne XL robot has 4 cylinders and 3 rectangular prisms. Here, the assumption is that the shoulder joint and gripper are circumscribed by rectangular prisms. Also, the obstacles are rod (cylinder) with sphere (oscillating obstacle) and 2 cubes (stationary obstacles). In our paper, we considered the whole structure of the robot manipulator along with all the obstacles. We have considered 8 points (corner points) for cube and rectangular prism, 8 points for cylinder (4 quadrant points on circular face on each side) and 4 quadrant points for sphere. So 56 points for AdeptOne XL robot, 12 points for oscillating obstacle and 16 points for stationary obstacles are considered. All the points of AdeptOne XL robot and obstacles are stored in separate arrays of software subroutine for obstacle avoidance.

The sets $d_q(t)$ describe the points of the parts of the robot as given by (1)–(6). The sets C_l characterize the shape of the rigid bodies (parts of the robot), while $T_l(t)$ and $R_l(t)$

describe the translation and rotation of the bodies, respectively, i.e.,

$$d_q(t) = T_l(t)R_l(t)C_l. \tag{17}$$

Each point in $d_q(t)$ can be calculated as

$$\begin{bmatrix} x_l(t_i) \\ y_l(t_i) \\ z_l(t_i) \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & p_x(t) \\ 0 & 1 & 0 & p_y(t) \\ 0 & 0 & 1 & p_z(t) \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos\theta(t_i) & -\sin\theta(t_i) & 0 & 0 \\ \sin\theta(t_i) & \cos\theta(t_i) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_l(t_0) \\ y_l(t_0) \\ z_l(t_0) \\ 1 \end{bmatrix} \tag{18}$$

where P_x, P_y, and P_z represent the translation, and $\theta(t)$ represents the rotation of the points x_l, y_l, and z_l.

Let a point on an obstacle be z_l and a point that belongs to the robot be z_q. Let a set $d_l(t)$ define all points of the obstacles. A set $d_{lq}(t)$ defines the distances between the points in the sets $d_l(t)$ and $d_q(t)$. The distance between the sets $d_{lq}(t)$ must be recalculated for all points at each time instant t. Here, the obstacles are stationary, so changes in position of points of the robot due to the movement of robot are found by (7). Now, the software subroutine for obstacle avoidance can find the distance between the robot points and obstacle points. If $d_{lq}(t) > 0$, then the corresponding trajectory will be accepted by the software subroutine for obstacle avoidance; otherwise, it will be rejected. In this work, we have checked that there is no duplication of set of points of the robot and obstacles. So the collision results are reliable.

3 Numerical application

AdeptOne XL robot manipulator having four degrees of freedom is considered here (see Fig. 3). The geometrical and limiting parameters of AdeptOne XL robot are presented in Tables 1 and 2, respectively[16]. It is considered that the robot is initially at rest and comes to a full stop at the end of the trajectory. Therefore, $\dot{q}_1 = \dot{q}_m = \ddot{q}_1 = \ddot{q}_m = 0$ for all robot joints. In this application, the end-effector trajectory (ψ_1) of the AdeptOne XL manipulator has to avoid two fixed obstacles (ψ_3 and ψ_4) and an oscillating obstacle – pendulum (ψ_2) as shown in Fig. 4. The goal is to obtain the optimal trajectory under the constraints given in Table 2[16]. In this problem, twenty-knot ($m = 20$) points have been considered for the trajectory. Cubic B-spline coefficients of the joint trajectories are considered as variables, and the total number of variables is 88. A pendulum is considered as an oscillating obstacle, and two cubes are considered as stationary obstacles. The geometrical details of the obstacles are given below:

Cube 1. $x_1 = 0.7$ and $x_2 = 1.1$; $y_1 = 0.6$ and $y_2 = 0.8$; $z_1 = 0.0$ and $z_2 = 0.30$.

Cube 2. $x_1 = 0.7$ and $x_2 = 1.1$; $y_1 = 1.27$ and $y_2 = 1.47$; $z_1 = 0.0$ and $z_2 = 0.21$.

Pendulum. length $l = 1\,\text{m}$, $x = l\cos(\alpha)$; $z = h$; $y = $ constant, $\alpha = \arccos(1 - (h/l))$, $h = (v_o^2 - v)/(2g)$; $v_o = 2\,\text{m/s}$; $g = 9.81\,\text{m/s}^2$, $v = v_o \sin(3.14t)$.

Fig. 3 AdeptOne XL robot manipulator

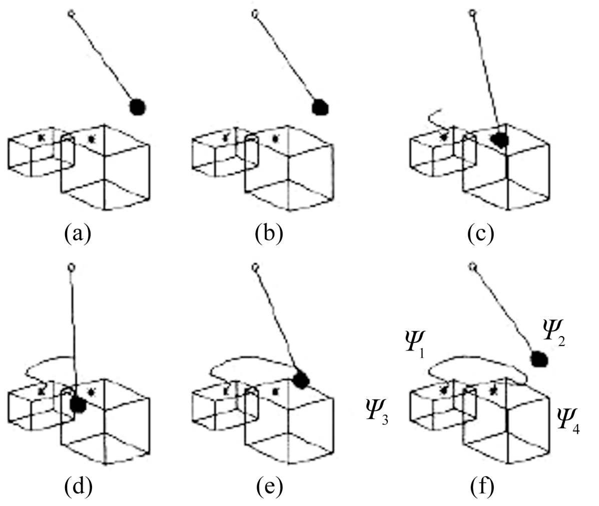


Fig. 4 End-effector tridimensional optimal trajectory. (a)–(f): Motions of end-effector through tridimensional optimal trajectory

Table 1 Joint parameters for AdeptOne XL robot[16]

Joints	a_i (mm)	α_i	d_i (mm)	θ_i
1	425	180^0	876	θ_1
2	375	0	0	θ_2
3	0	0	d_3	0
4	0	0	204	θ_4

Table 2 Limiting values for AdeptOne XL robot[16]*

Constraint	1	2	3	4
QC (rad)	2.6180	2.4435	203	4.7124
VC (rad/s)	11.3446	16.057	1200	57.5959
WC (rad/s^2)	9.5	9.5	9.5	9.5
JC (rad/s^3)	50	50	50	50
UC (N·m)	400	280	100	100

*Joint 3 – QC (mm), VC (mm/s), WC (mm/s^2), JC (mm/s^3), UC (N·m).

4 Implementation of NSGA-II and MODE algorithms

The steps for running the algorithms are summarized below:

Step 1. The following are the inputs to NSGA-II[17, 18] and MODE[19] algorithms:

1) Details of AdeptOne XL robot: geometrical parameters (see Table 1), and displacement, velocity, acceleration, jerk, and torque limits of each robot joint (see Table 2).

2) Details of cubic B-spline curve that defines the trajectory: total number of knot points considered = 20, the initial and final configurations of the end-effectors are $q_1 =$ [0.1682 rad, 1.3849 rad, 1.1362 m, −0.7555 rad, −0.4702 rad, 0.1472 rad] and $q_m =$ [−0.7610 rad, 0.2450 rad, 1.4366 m, 1.0095 rad, −1.0146 rad], and obstacle avoidance checking (13).

3) The formulae to find objective functions (z_1 to z_6), robot joints displacement (q), robot joints velocity ($\dot{q}$), robot joints acceleration ($\ddot{q}$), robot joints jerk ($\dddot{q}$), robot joints torque (u_i) at each time instant (1)–(18).

4) The formulae to find multi-objective performance measures, namely, solution spread measure, ratio of non-dominated individuals, optimiser overhead, and algorithm effort.

5) The formulae to find combined objective function (f_c) and average fitness factor value (F_{avg}). Based on these values, the best optimal solution from Pareto optimal front is selected.

6) The variables bounds.

7) The values of the parameter that have been used in the NSGA-II technique to get the best optimal solution are variable type = real variable, population size = 100, crossover probability = 0.7, real-parameter mutation probability = 0.01, real-parameter simulated binary crossover (SBX) parameter = 10, real-parameter mutation parameter = 100, total number of generations = 100. The values of the parameter that have been used in the proposed MODE technique for getting the best optimal solution are Strategy = MODE/rand/bin, crossover constant $CR = 0.9$, number of population $NP = 500$, $F = 0.5$, and total number of generations = 100.

Step 2. The software programs of NSGA-II and MODE algorithms find the optimal variables in such a way that

1) Travelling time, mechanical energy of the actuators, joint jerks, joint accelerations, and penalty function to guarantee collision-free motion are minimum, and manipulability measure of the robot is maximum.

2) All constraints (joint limits, actuator limits, and obstacle avoidance) are satisfied.

Step 3. Step 2 is repeated up to the maximum number of iterations.

Step 4. The following are the outputs from NSGA-II and MODE algorithms:

1) Pareto optimal fronts obtained from NSGA-II and MODE. A Pareto optimal front gives the number of trade-off solutions. Each solution has optimal objective function value, optimal variable value and the constraint value. All constraints will be satisfied by any solution in the Pareto optimal front.

2) The optimal displacement Q (rad or m), velocity V (rad/s or m/s) and acceleration W (rad/s^2 or m/s^2) of all the robot joints.

3) The best optimal solution trade-off from Pareto optimal fronts is selected by using the methods, namely, normalized weighting objective functions and average fitness factor in a combined manner. The best optimal solution gives a safer, faster, economic, and smoother optimal trajectory.

4) The strength of Pareto optimal fronts is evaluated by the two multi-objective performance measures, namely, solution spread measure and ratio of non-dominated individuals.

5) The computational effort of NSGA-II and MODE algorithms is calculated by the two multi-objective performance measures, namely, optimiser overhead and algorithm effort.

5 Results and discussion

The optimal solution trade-offs obtained from NSGA-II and MODE are given in Figs. 5 and 6, respectively. From these figures, it is observed that NSGA-II gives optimal solution trade-offs with more number of non-dominated solutions for user′s choice than MODE. So NSGA-II technique is the best one for this multicriterion optimization problem, if the user wants more number of solution trade-offs for his choice.

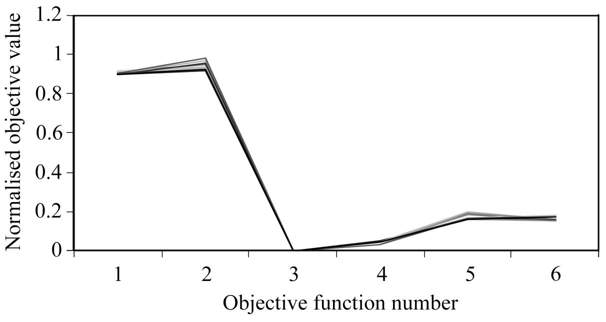


Fig. 5 Optimal solution trade-offs obtained from NSGA-II

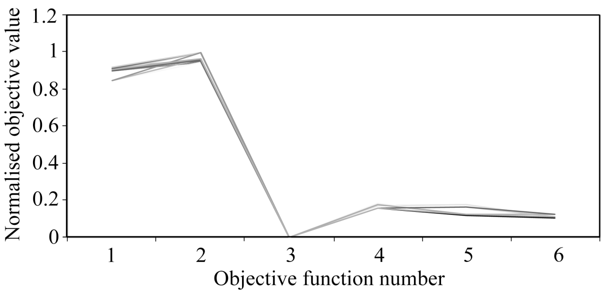


Fig. 6 Optimal solution trade-offs obtained from MODE

The optimization algorithm that gives minimum combined objective function (f_c), maximum average fitness factor value (F_{avg}), minimum solution spread measure (SSM), maximum ratio of non-dominated individuals (RNI), minimum optimiser overhead (OO), and minimum algorithm effort is the best one.

The results from NSGA-II and MODE are listed in Tables 3–5. From Tables 3–5, it is observed that MODE gives minimum combined objective function (f_c), maximum average fitness factor (F_{avg}), minimum optimiser overhead (OO), and minimum algorithm effort than NSGA-II. But NSGA-II technique gives minimum solution spread measure (SSM), and maximum ratio of non-dominated individuals (RNI) than MODE.

Table 3 Results obtained from NSGA-II and MODE algorithms

	z_1	z_2	z_3	z_4	z_5	z_6	$F_{\rm avg}$
$Z_{\max}$	1.0	9506597.528661	0.00005	0.157367	279.932092	269.784010	-
$Z_{\min}$	0.848	9178597.575731	0.0	0.048277	119.843157	107.837977	-
NSGA-II	0.898	9178597.575731	0.0	0.048277	163.237956	156.112759	0.683650
MODE	0.848	9468839.987070	0.0	0.157367	119.843157	107.837977	0.852519

Table 4 Combined objective function and algorithm effort obtained from NSGA-II and MODE algorithms

Proposed algorithm	Combined objective function (f_c)	Simulation time $T_{\rm run}$(s)	No. of function evolution ($N_{\rm eval}$)	Algorithm effort
NSGA-II	0.497040	2	87	0.02299
MODE	0.467136	2	124	0.01575

Table 5 SSM, RNI, and OO obtained from NSGA-II and MODE algorithms

Technique	SSM	RNI	OO
NSGA-II	0.01023	0.32	0.0769
MODE	0.02356	0.24	0.0324

In a combined manner, the two methods (normalized weighting objective functions and average fitness factor methods) are used to select the best optimal solution trade-offs from the optimal solution trade-offs obtained from NSGA-II and MODE. The best optimal solution tradeoffs obtained from NSGA-II and MODE for the sample case $w_1 = w_2 = 0.25$, $w_3 = w_4 = 0.1$, $w_5 = 0.15$, and $w_6 = 0.15$ are shown in Fig. 7. From Fig. 7, it is noted that MODE gives best results for five objective functions (minimum values for z_1, z_3, z_5, and z_6, and maximum value for z_4). It is noted that MODE technique converges quickly than NSGA-II. Also, the computational time to find optimum solution tradeoffs in MODE is 0.33 of that of NSGA-II. So, MODE is faster than NSGA-II. Hence, MODE technique is the best one for this multicriterion optimization problem, if the user wants a best optimal solution trade-off very quickly. Figs. 8 and 9 show the optimal displacement ($Q = q_j^i(t)$ (rad or m)), velocity ($V = {\rm d}(q_j^i(t))/{\rm d}t$ (rad/s or m/s)), and acceleration ($W = {\rm d}^2(q_j^i(t))/{\rm d}t^2$ (rad/s^2 or m/s^2)) of all the robot joints obtained from MODE and NSGA-II. From Figs. 8 and 9, it is noted that the robot joints displacement, velocity, and acceleration are within their limiting values.

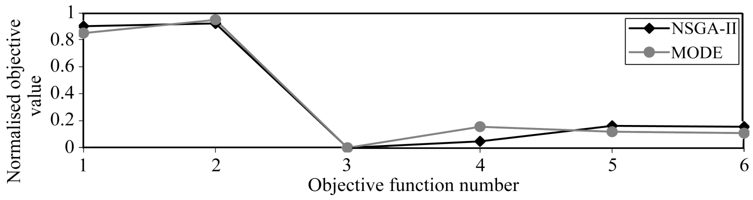


Fig. 7 Best optimal solution tradeoffs obtained from NSGA-II and MODE

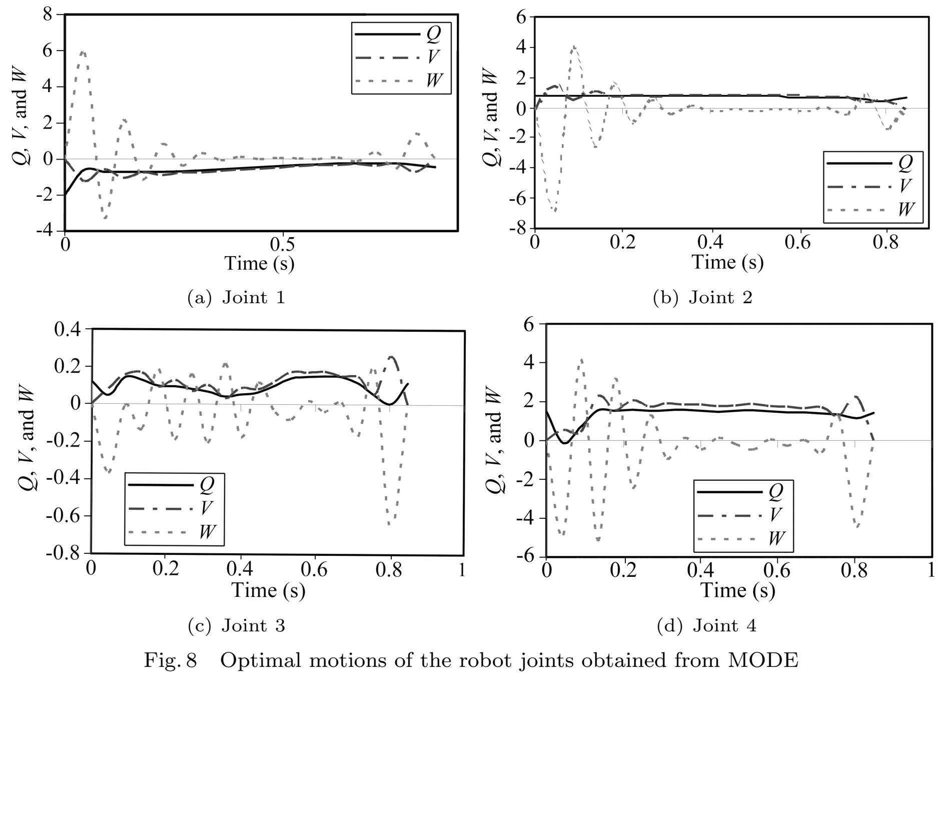
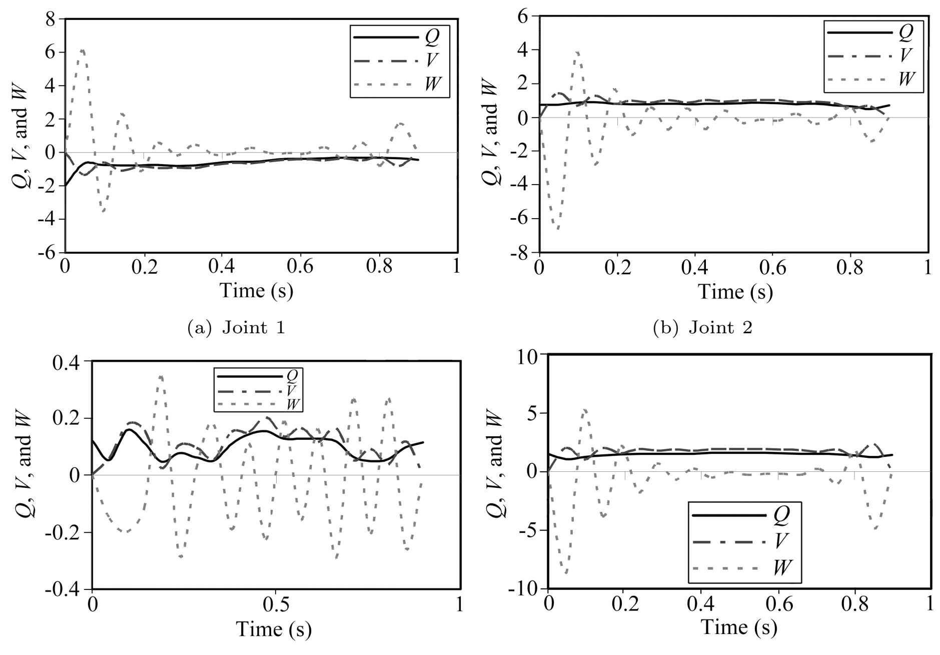


Fig. 8 Optimal motions of the robot joints obtained from MODE



(a) Joint 1 (b) Joint 2

(c) Joint 3 (d) Joint 4

Fig. 9 Optimal motions of the robot joints obtained from NSGA-II

6 Conclusions

A new and general methodology for off-line tridimensional optimal trajectory planning of the industrial robot manipulator (AdeptOne XL) in the presence of fixed and oscillating obstacles using NSGA-II and MODE is presented. Obstacle avoidance is obtained by adding penalty functions to the multicriterion problem. When dealing with fixed and oscillating obstacles, all the objective functions and the constraint functions have to be updated simultaneously at each time instant. Two methods, namely, normalized weighting objective functions and average fitness factor are combinedly used to select the best solution tradeoffs. Two multi-objective performance measures, namely, solution spread measure and ratio of non-dominated individuals, are used to evaluate the Pareto optimal fronts. Two multi-objective performance measures, namely, optimizer overhead and algorithm effort, are used to find the computational effort of an optimization algorithm. A numerical application demonstrated the efficiency of the proposed algorithms. From the results, it is concluded that MODE technique is the best one for this multicriterion optimization problem, if the user wants a best optimal solution trade-off very quickly. Also, NSGA-II technique is the best for this multicriterion optimization problem, if the user wants more number of solution trade-offs for his choice. In the future, instead of B-spline function, non uniform rational B-spline (NURBS) function will be used to represent the trajectory for better accuracy. A general-purpose comprehensive user-friendly software package has been developed for MODE algorithm using VC++ to obtain the optimal trajectory planning. This software can be used for the design optimisation problems in any field.

This work opens the door for further investigations on how the evolutionary optimization techniques can be used to solve complex problems.

Acknowledgement

We thank the support provided by Adept Technology Inc., Canada for providing the details of AdeptOne XL robot used in this study.

References

[1] Z. Shiller, S. Dubowsky. The global time-optimal motions of robotic manipulators in the presence of obstacles. *IEEE Transactions on Robotics and Automation*, vol. 7, no. 6, pp. 785–797, 1991.

[2] C. Santos, M. Ferreira. Timed trajectory generation using dynamical systems: Application to a puma arm. *Robotics and Autonomous Systems*, vol. 57, no. 2, pp. 182–193, 2009.

[3] H. F. Wang, Y. Z. Chen. Time-optimal trajectories for a car-like robot. *Acta Automatica Sinica*, vol. 34, no. 4, pp. 445–452, 2008.

[4] A. Khoukhi, L. Baron, M. Balazinski, K. Demirli. A hierarchical neuro-fuzzy system to near optimal-time trajectory planning of redundant manipulators. *Engineering Applications of Artificial Intelligence*, vol. 21, no. 7, pp. 974–984, 2008.

[5] S. F. P. Saramago, V. J. Steffen. Optimization of the trajectory planning of robot manipulators taking into account the dynamics of the system. *Mechanism and Machine Theory*, vol. 33, no. 7, pp. 883–894, 1998.

[6] S. F. P. Saramago, V. J. Steffen. Optimal trajectory planning of robot manipulators in the presence of moving obstacles. *Mechanism and Machine Theory*, vol. 35, no. 8, pp. 1079–1094, 2000.

[7] A. Gasparetto, V. Zanotto. A new method for smooth trajectory planning of robot manipulators. *Mechanism and Machine Theory*, vol. 42, no. 4, pp. 455–471, 2007.

[8] A. Gasparetto, V. Zanotto. A technique for time-jerk optimal planning of robot trajectories. *Robotics and Computer Integrated Manufacturing*, vol. 24, no. 3, pp. 415–426, 2008.

[9] A. Elnagar, A. Hussein. On optimal constrained trajectory planning in 3D environments. *Robotics and Autonomous Systems*, vol. 33, no. 4, pp. 195–206, 2000.

[10] C. J. Lin. Motion planning of redundant robots by perturbation method. *Mechatronics*, vol. 14, no. 3, pp. 281–297, 2004.

[11] Maria da Graça Marcos, J. A. Tenreiro Machado, T. P. Azevedo-Perdicoúlis. Trajectory planning of redundant manipulators using genetic algorithms. *Communications in Nonlinear Science and Numerical Simulation*, vol. 14, no. 7, pp. 2858–2869, 2009.

[12] E. G. Gillbert, D. W. Johnson, S. S. Keerthi. A fast procedure for computing the distance between complex objects in three-dimensional space. *IEEE Journal of Robotics and Automation*, vol. 4, no. 2, pp. 193–203, 1988.

[13] T. Meyarivan, K. Deb, A. Pratap, S. Agarwal. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[14] K. Deb. *Multi-objective Optimization Using Evolutionary Algorithms*, 3rd Print, Chichester, UK: Wiley, 2001.

[15] B. V. Babu, B. Anbarasu. Multi-objective differential evolution (MODE): An evolutionary algorithm for multi-objective optimization problems (MOOPs). [Online], Available: http://discovery.bits-pilani.ac.in/˜bvbabu/CIRAS_2005_MODE_Final.pdf, 2005.

[16] K. C. Tan, T. H. Lee, E. F. Khor. Evolutionary algorithms for multi-objective optimization: Performance assessments and computations. *Artificial Intelligence Review*, vol. 17, no. 4, pp. 251–290, 2002.

[17] Adept Technology Inc. *AdeptOne Robot with Smart-Controller User's Guide (sEJI)*, [Online], Available: http://www1.adept.com/main/KE/DATA/Robot/A1_SC_sEJI/A1_SC_UG.pdf, December 2005.

[18] C. D. Wu, Y. Zhang, M. X. Li, Y. Yue. A rough set GA-based hybrid method for robot path planning. *International Journal of Automation and Computing*, vol. 3, no. 1, pp. 29–34, 2006.

[19] D. Xu, A. Carlos, Acosta Calderon, J. Q. Gan, H. Hu, M. Tan. An analysis of the inverse kinematics for a 5-DOF manipulator. *International Journal of Automation and Computing*, vol. 2, no. 2, pp. 114–124, 2005.

R. Saravanan received the B. Eng. degree in mechanical and production engineering in 1985 and M. Eng. degree in production engineering in 1992 from Annamalai University, Chithambaram, India. He received the Ph. D. degree in computer-aided manufacturing (CAM) in 2001 from National Institute of Technology, Trichy, India. He has 15 years of teaching experience in various engineering colleges in Tamil Nadu, India. Now, he is a professor and head of Department of Mechanical Engineering, Bannariamman Institute of Technology, Sathiyamangalam, Tamil Nadu, India.

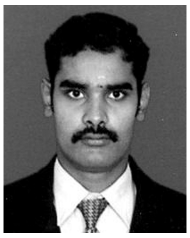
His research interests include robotics, CAM, and non-conventional optimization algorithms.

E-mail: saradharani@hotmail.com

S. Ramabalan received the B. Eng. degree in mechanical engineering in 1997 from Madras University and the M. Eng. degree in CAD/CAM in 2003 from Madurai Kamaraj University, Tamil nadu, India. He received the Ph. D. degree in robotics in Anna University, Chennai, India. He has 12 years of teaching experience. Now, he is a professor in the Department of Mechanical Engineering, J. J. College of Engineering and Technology, Trichy, Tamil Nadu, India.

His research interests include robotics and non-conventional optimization algorithms.

E-mail: cadsrb@gmail.com (Corresponding author)



C. Balamurugan received the B. Eng. degree in mechanical engineering in 2002 from Bharathidasan University and M. Eng. degree in CAD/CAM in 2006 from Anna University, Tamil Nadu, India. Currently, he is a Ph. D. candidate in tolerance analysis in Anna University, Chennai, India, and working as a lecturer in the Department of Production Engineering, J. J. College of Engineering and Technology, Trichy, Tamilnadu, India.

His research interests include tolerance analysis and non-conventional optimization algorithms.

E-mail: rlc_bal@yahoo.co.in

A. Subash received the B. Eng. degree in mechanical engineering in 2005 from Bharathidasan University and M. Eng. degree in CAD/CAM in 2007 from Anna University, Tamil Nadu, India. Now, he is working as a lecturer in the Department of Mechanical Engineering., J. J. College of Engineering and Technology, Trichy, Tamil Nadu, India.

His research interests include robotics, CAD/CAM, and intelligent optimization techniques.

E-mail: subash_devi@yahoo.co.in