Engineering Applications of Artificial Intelligence 22 (2009) 329–342

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

Evolutionary multi-criteria trajectory modeling of industrial robots in the presence of obstacles

R. Saravanan [a], S. Ramabalan [b,*], C. Balamurugan [b]

[a] Department of Mechatronics Engineering, Kumaraguru College of Technology, Coimbatore 641006, Tamilnadu, India
[b] Faculty of CAD/CAM (P.G. Course), J.J. College of Engineering and Technology, Thiruchirapalli 620009, Tamilnadu, India

ARTICLE INFO

Article history:
Received 15 October 2007
Received in revised form
11 April 2008
Accepted 10 June 2008
Available online 28 August 2008

Keywords:
Multi-objective optimal trajectory modeling
Obstacle avoidance
Elitist non-dominated sorting genetic algorithm (NSGA-II)
Multi-objective differential evolution (MODE)

ABSTRACT

Optimal trajectory planning for robot manipulators is always a hot spot in research fields of robotics. This paper presents two new novel general methods for computing optimal motions of an industrial robot manipulator (STANFORD robot) in presence of obstacles. The problem has a multi-criterion character in which three objective functions, a maximum of 72 variables and 103 constraints are considered. The objective functions for optimal trajectory planning are minimum traveling time, minimum mechanical energy of the actuators and minimum penalty for obstacle avoidance. By far, there has been no planning algorithm designed to treat the objective functions simultaneously. When existing optimization algorithms of trajectory planning tackle the complex instances (obstacles environment), they have some notable drawbacks viz.: (1) they may fail to find the optimal path (or spend much time and memory storage to find one) and (2) they have limited capabilities when handling constraints. In order to overcome the above drawbacks, two evolutionary algorithms (Elitist non-dominated sorting genetic algorithm (NSGA-II) and multi-objective differential evolution (MODE) algorithm) are used for the optimization. Two methods (normalized weighting objective functions method and average fitness factor method) are combinedly used to select best optimal solution from Pareto optimal front. Two multi-objective performance measures (solution spread measure and ratio of non-dominated individuals) are used to evaluate strength of the Pareto optimal fronts. Two more multi-objective performance measures namely optimizer overhead and algorithm effort are used to find computational effort of NSGA-II and MODE algorithms. The Pareto optimal fronts and results obtained from various techniques are compared and analyzed.

1. Introduction

The design of engineering systems involves simultaneous consideration of multiple criteria or objectives. Some of these objectives will be in conflict often. Thus, a trade-off exists, which can be investigated by using multi-objective optimization methods. In such a problem, no single optimal solution exists; rather there is a set of equally valid optimal solutions known as the Pareto optimal set. The solutions in this set show the designer what is possible and allow him to make a fully informed choice.

The goal of robot systems is to do tasks at a cost as low as possible. Thus, the minimum-cost trajectory planning in the two-stage realization of manipulators control (i.e., planning first and tracking next) is an important effort to accomplish the goal. A large number of robotic applications involve repetitive processes. This technological characteristic justifies offline trajectory planning.

In order to maximize the speed of operation that affects the productivity in industrial situations, it is necessary to minimize the total traveling time of the robot. More research works have been carried out to get minimum time trajectories (Shiller and Dubowsky, 1991).

Planning the robot trajectory using energy criteria provides several advantages. It yields smooth trajectories easier to track and reduce the stresses of the actuators and of the manipulator structure. Moreover, saving energy may be desirable in several applications, such as those with a limited capacity of the energy source (e.g., robots for spatial or submarine exploration). Examples of energy optimal trajectory planning are provided in some literatures. Both optimal traveling time and the minimum mechanical energy of the actuators are considered together as objective functions in some literatures (Saramago and Steffen, 1998, 1999, 2001 and Chettibi et al., 2004). Saramago and Steffen (2001) used sequential unconstrained minimization techniques (SUMT) to do optimum trajectory planning of an STANFORD

* Corresponding author. Tel.: +91 431 2695608; fax: +91 431 2695607.
E-mail addresses: saradharani@hotmail.com (R. Saravanan), cadsrb@gmail.com (S. Ramabalan).

0952-1976/$ - see front matter © 2008 Elsevier Ltd. All rights reserved.
doi:10.1016/j.engappai.2008.06.002

Nomenclature

J	robot joint number
m	knot number of the robot joint trajectory
K_1	robot joint trajectory in the workspace
K_2, K_3 and K_4	obstacles
z_1	traveling time of the robot
z_2	mechanical energy of the robot actuators
z_3	penalty for obstacle avoidance
$\dot{q}$	generalized robot joint velocity
$\ddot{q}$	generalized robot joint acceleration
D_{ij}	inertial system matrix of the robot
C_{ijk}	coriolis and centripetal forces matrix
G_i	gravity loading vector,
J_i	moments of Inertia of robot
$\bar{r}_i$	center of mass of robot links
g	acceleration due to gravity with respect to the base coordinate system
T	traveling time of robot end effector from initial configuration to final robot configuration
u_i	generalized forces
QC_j	maximum displacement of robot joint j
VC_j	maximum velocity of robot joint j
WC_j	maximum acceleration of robot joint j
JC_j	maximum jerk of robot joint j
UC_j	maximum force/torque of robot joint j
I	set of possibly colliding pairs of parts
$h\varepsilon R^m$	design variables (time intervals) for strategy 1
γ	B-spline coefficients (variables) in strategy 2
$t_1<t_2\cdots<t_{m-1}<t_m$	an ordered time sequence
$h_i=t_{i+1}-t_i$	time interval
$T=\sum h_i$	total time
$\dot{q}_1$, $\dot{q}_m$ and $\ddot{q}_1$, $\ddot{q}_m$	robot joint velocities and accelerations at the initial time t_1 and terminal time t_m
γ_j^i	B-spline coefficients

manipulator. The objective functions used in optimal trajectory planning are minimum traveling time, minimum mechanical energy of the actuators and minimum penalty for obstacle avoidance. The limitations of their work are: (1) They have approached the problem as a single objective optimization problem (all objectives are combined as a single objective function using weightage objective function method). (2) If the user does not know the weightage to be given for each objective, their approach is not applicable. (3) The method used by them cannot be used for treating multi-objectives simultaneously. (4) They got only one optimal solution. But for a real world problem, a Pareto optimal front that offers a good number of optimal solutions for user's choice is most desirable and (5) SUMT is a conventional optimization technique and hence a global solution may not be possible.

The methods that are used in the literatures such as dynamic programming (Shiller and Dubowsky, 1991), SUMT (Saramago and Steffen, 1998, 1999, 2001) and sequential quadratic programming (Chettibi et al., 2004) to tackle the complex instances (obstacles environment) have some notable drawbacks viz.: (1) they may fail to find the optimal path (or spend much time and memory storage to find one) and (2) they have limited capabilities when handling constraints. For example, the robot joint acceleration and deceleration may not be within their maximum limits. To overcome the above drawbacks, the evolutionary algorithms can be used. The advantages of evolutionary techniques are: (1) They are population based search. So a global optimal solution is possible. (2) They do not require any auxiliary information like gradients, derivatives, etc. (3) Complex and multimodal problems can also be solved for global optimality. (4) They are problem independent techniques, i.e., suitable for all types of problems.

In the last 20 years, evolutionary algorithms such as multi-objective genetic algorithm (MOGA) (Fonseca and Fleming, 1995), Elitist non-dominated sorting genetic algorithm (NSGA-II) (Deb et al., 2002), multi-objective differential evolution (MODE) (Babu and Anbarasu, 2005), Niched Pareto genetic algorithm (NPGA) (Horn et al., 1994), among many other variants (Coello and Carlos, 1999) have been applied in a plethora of fields such as control, system identification, robotics, planning and scheduling, image processing, pattern recognition and speech recognition (Bäck et al., 1997). Evolutionary techniques for multi-objective optimization 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 optimization are capable of treating each objective component separately and lead the search in discovering the global Pareto optimal front.

Intelligent optimization algorithms such as NSGA-II and MODE are very much needed for trajectory planner of an intelligent real world robot. Since trajectory planning for a real world robot is a very complex and tedious task. This is due to the following reasons:

1. The planning algorithm has to consider the dynamic model of the robot. The dynamic model is depending on traveling time, payload and robot's task. It 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, the obstacle avoidance checking is a very complex and time-dependent one.
3. The environment around the robot is an ever-changing one. This calls for the planning algorithm to update the details for trajectory planning for each time instant.

In this paper, two evolutionary algorithms namely NSGA-II and MODE are proposed to obtain optimal trajectory planning for an industrial robot (STANFORD Robot). The result of a multi-objective optimization is a Pareto optimal front, which is a set of competing solutions. However, the design implementation for a real-time problem will require a single solution. Two methods (normalized weighting objective functions method and average fitness factor method) are combinedly used to select best optimal solution from Pareto optimal front. Two multi-objective performance measures namely solution spread measure (SSM) and ratio of non-dominated individuals are used to evaluate the strength of Pareto optimal fronts. Two more multi-objective performance measures namely optimizer overhead and algorithm effort are used to find the computational effort of NSGA-II and MODE algorithms. This research work considers all the important decision criteria for the optimal trajectory planning of industrial robot manipulators, including the obstacle avoidance criteria for the obstacles and also incorporates information vagueness (Fuzziness).

The trajectory modeling can be divided into two groups:

(i) To obtain a constrained trajectory considering the condition that trajectory must pass through a given number of points.
(ii) To obtain the constrained trajectory given only the initial and final points.

This paper deals with optimal trajectory modeling of both groups as two strategies. The optimal robot manipulator trajectory is the one that minimizes the objectives without any obstacle collision in the workspace. Following this introduction, the rest of the paper is organized as follows: Section 2 presents the problem statement of this paper. In Section 3, the proposed NSGA-II and MODE techniques to obtain the optimal trajectory planning for STANFORD robot are presented. Section 4 deals with two methods and four performance metrics proposed and used for multi-objective optimization. In Section 5, two numerical examples are presented to illustrate the proposed optimization methodologies. In Section 6, the results obtained from various techniques are presented and compared. The conclusions are presented in Section 7.

2. Problem statement

The industrial robot manipulator STANFORD robot with six degrees of freedom is considered. Let j represents the robot joint number and m represents the knot number of the trajectories. The task is to move the robot along the trajectory K_1 in the workspace avoiding the obstacles K_2, K_3 and K_4, while minimizing the traveling time, the energy consumed in the move and penalty for obstacle avoidance, subject to physical constraints and actuator limits as shown in Fig. 1. The obstacles are considered as objects sharing the same workspace performed 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.

The traveling time (z_1), the mechanical energy of the actuators (z_2) and penalty for obstacle avoidance (z_3) are considered as objective functions and the optimization problem is defined as follows (Saramago and Steffen, 2001):

Minimize:

$$z_1 = T, \tag{1a}$$

$$z_2 = \int_0^T (u_i(t))^2\,dt, \tag{1b}$$

$$z_3 = \sum_{i=1}^{\mathrm{Obs}} \frac{1}{(\min\{d_{lq}\})^2}, \tag{1c}$$

Subject to:

$$|Q_{ji}(t)| \leqslant \mathrm{QC}_j,\quad \text{for } j = 1,2,\ldots,N \quad\text{and}\quad I = 1,2,\ldots,m-1, \tag{2}$$

$$|V_{ji}(t)| \leqslant \mathrm{VC}_j,\quad \text{for } j = 1,2,\ldots,N \quad\text{and}\quad I = 1,2,\ldots,m-1, \tag{3}$$

$$|W_{ji}(t)| \leqslant \mathrm{WC}_j,\quad \text{for } j = 1,2,\ldots,N \quad\text{and}\quad I = 1,2,\ldots,m-1, \tag{4}$$

$$|J_{ji}(t)| \leqslant \mathrm{JC}_j,\quad \text{for } j = 1,2,\ldots,N \quad\text{and}\quad I = 1,2,\ldots,m-1, \tag{5}$$

$$|U_{ji}(t)| \leqslant \mathrm{UC}_j,\quad \text{for } j = 1,2,\ldots,N \quad\text{and}\quad I = 1,2,\ldots,m-1, \tag{6}$$

$$d_{lq}(t) \geqslant 0 \quad \text{for } (l,q) \in I, \tag{7}$$

$$h_i^{(L)} \leqslant h_i \leqslant h_i^{(U)} \quad \text{for } i = 1,2,\ldots,m-1, \tag{8a}$$

$$\gamma_i^{j(L)} \leqslant \gamma_i^j \leqslant \gamma_i^{j(U)} \quad \text{for } i = 1,2,\ldots,m-1 \quad\text{and}\quad j = 1,2, \text{ and } 6. \tag{8b}$$

Eq. (7) gives the obstacle avoidance. Eqs. 8(a) and 8(b) represent the side constraints for strategies 1 and 2, respectively.

2.1. Kinematic and dynamic models

According to Saramago and Steffen (2001), the generalized forces u_i are calculated as:

$$u_i = \sum_{j=1}^{n}\sum_{k=1}^{j} \mathrm{Tr}[U_{jk}J_i(U_{ji})^{\mathrm{T}}]\ddot{q}_k + \sum_{i=1}^{n}\sum_{k=1}^{j}\sum_{m=1}^{j} \mathrm{Tr}[U_{jkm}J_i(U_{ji})^{\mathrm{T}}]\dot{q}_k\dot{q}_m - \sum_{j=1}^{n} m_j g^{\mathrm{T}}(U_{ji}\bar{r}_j). \tag{9}$$

The above equation can be rewritten as:

$$u_i = \sum_{j=1}^{n} D_{ij}\dot{q}_j + \sum_{j=1}^{n}\sum_{k=1}^{j} C_{ijk}\dot{q}_j\dot{q}_k + G_i, \tag{10}$$

$$D_{ij} = \sum_{p=\max(i,j)}^{n} \mathrm{Tr}[U_{pj}J_p(U_{pj})^{\mathrm{T}}], \tag{11}$$

$$C_{ijk} = \sum_{p=\max(i,j,k)}^{n} \mathrm{Tr}[U_{pjk}J_p(U_{pi})^{T}], \tag{12}$$

$$G_i = \sum_{p=1}^{n} -m_p g^{\mathrm{T}}(U_{pi}\bar{r}_p). \tag{13}$$

The generalized forces u_i calculated by Eq. (9) will be used in the optimization problem (1a)–(1c) and in the force/torque constraint equations given by Eq. (6).

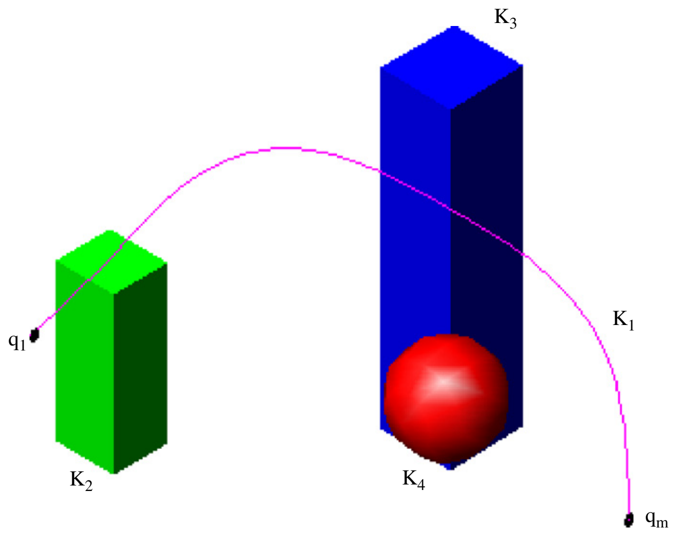

Fig. 1. Optimal path with obstacles.

2.2. Formulation of trajectories

2.2.1. Strategy 1 (trajectory modeling with several given points)

In this strategy, the aim is to obtain a constrained trajectory considering the trajectory must pass through a given number of points (Fig. 2).

A cubic polynomial function $Q_i(t)$ is defined on the time interval. It is assumed that the displacement, velocity and acceleration are continuous on this time interval. The second time derivative $W_i(t)$ can be written as:

$$W_i(t) = \frac{t_{i+1} - t}{h_i} W_i(t_i) + \frac{t - t_i}{h_i} W_i(t_{i+1}) \quad \text{for } i = 1, 2, \ldots, m-1, \tag{14}$$

where, $h_i = t_{i+1} - t_i$.

Integrating Eq. (14) for the given points $Q_i(t_i) = q_i$ and $Q(t_{i+1}) = q_{i+1}$, the following interpolating functions are obtained:

$$V_i(t) = \frac{-W_i}{2h_i}(t_{i+1} - t)^2 + \frac{W_{i+1}}{2h_i}(t - t_i)^2 + \left(\frac{q_{i+1}}{h_i} - \frac{h_i W_{i+1}}{6}\right) - \left(\frac{q_i}{h_i} - \frac{h_i W_i}{6}\right) \tag{15}$$

and

$$Q_i(t) = \frac{W_i(t)}{6h_i}(t_{i+1} - t)^3 + \frac{W_i(t_{i+1})}{6h_i}(t - t_i)^3) + \left(\frac{q_{i+1}}{h_i} - \frac{h_i W_i(t_{i+1})}{6}\right)(t - t_i) + \left(\frac{q_i}{h_i} - \frac{h_i W_i(t_i)}{6}\right)(t_{i+1} - t). \tag{16}$$

Two extra knots q_2 and q_{m-1} are not fixed and are used to add two new equations to the system in such a way that it can be solved. The joint displacements of these two knots are written as:

$$q_2 = q_1 + h_1 v_1 + \frac{h_1^2}{3} a_1 + \frac{h_1^2}{6} W_2(t_2), \tag{17}$$

$$q_{m-1} = q_m - h_{m-1} v_m + \frac{h_{m-1}^2}{3} a_m + \frac{h_{m-1}^2}{6} W_{m-1}(t_{m-1}). \tag{18}$$

The velocity will further be used in the optimization problem (Eqs. (1a)–(1c)).

2.2.2. Strategy 2 (trajectory modeling with initial and final points)

Only the initial and final points to construct the joint trajectories are given (Fig. 3). A uniform cubic B-spline is used to define the trajectory. A third-degree polynomial function f with the following properties defines the B-spline trajectory.

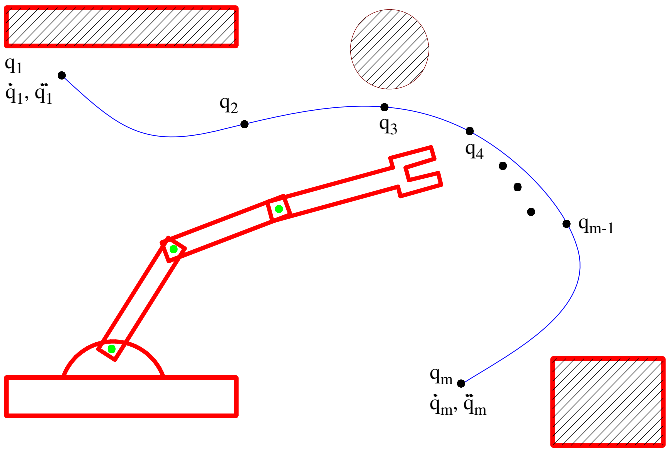

Fig. 2. Trajectory of gripper (strategy 1).

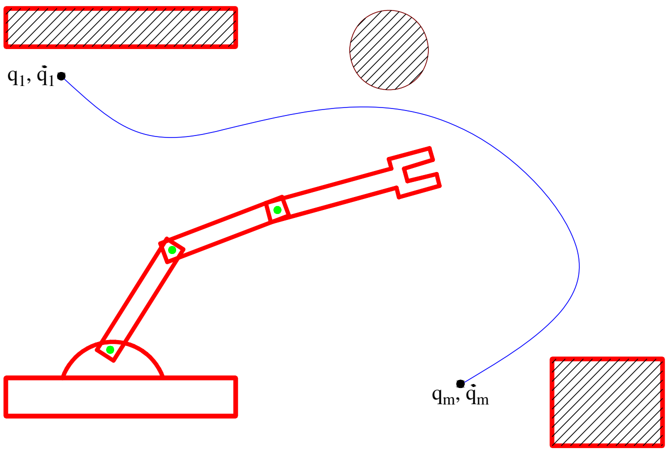

Fig. 3. Trajectory of gripper (strategy 2).

1. The knots are uniformly spaced, i.e., $\delta = t_{j+1} - t_j$ (constant).
2. For each interval $[t_j, t_{j+1}]$, the function f is equal to the cubic polynomial.

$$f_j(t) = q_j^i(t) = \gamma_{j-3}^i b_{-3}(t) + \gamma_{j-2}^i b_{-2}(t) + \gamma_{j-1}^i b_{-1}(t) + \gamma_{j-0}^i b_{-0}(t), \tag{19}$$

$$b_{-0}(t, t_j) = \frac{\mu_j^3(t)}{6}; \quad t \in I_j = [t_j, t_j + \delta]; \quad \mu_j(t) = \frac{t - t_j}{\delta}, \tag{20}$$

$$b_{-1}(t, t_{j+1}) = \frac{1 + 3\mu_{j+1}(t) + 3\mu_{j+1}^2(t) - 3\mu_{j+1}^3(t)}{6}; \quad t \in I_{j+1} = [t_{j+1}, t_{j+1} + \delta]; \ \mu_{j+1}(t) = \frac{t - t_{j+1}}{\delta}, \tag{21}$$

$$b_{-2}(t, t_{j+2}) = \frac{4 - 6\mu_{j+2}^2(t) + 3\mu_{j+2}^3(t)}{6}; \quad t \in I_{j+2} = [t_{j+2}, t_{j+2} + \delta]; \quad \mu_{j+2}(t) = \frac{t - t_{j+2}}{\delta}, \tag{22}$$

$$b_{-3}(t, t_{j+3}) = \frac{1 - 3\mu_{j+3}(t) + 3\mu_{j+2}^2(t) - \mu_{j+2}^3(t)}{6}; \quad t \in I_{j+3} = [t_{j+3}, t_{j+3} + \delta]; \quad \mu_{j+3}(t) = \frac{t - t_{j+3}}{\delta}, \tag{23}$$

where γ_j^i are the coefficients of the B-spline approximation for $q_i(t)$ in the interval 'I_j'.

The derivatives of $q_j^i(t)$ with respect to t are:

$$\frac{d^k q_j^i(t)}{dt^k} = \gamma_{j-3}^i b_{-3}^{(k)}(t) + \gamma_{j-2}^i b_{-2}^{(k)}(t) + \gamma_{j-1}^i b_{-1}^{(k)}(t) + \gamma_{j-0}^i b_{-0}^{(k)}(t), \tag{24}$$

where

$$b_{-j}^{(k)} = \frac{d^k(b_{-j})}{dt^k}.$$

For the problems with free terminal time a new time variable $\tau = t/T$ is introduced. This way the interval $[0, T]$ is replaced by the non-dimensional interval $[0, 1]$. In this case, the trajectories $q_i(\tau)$ are obtained with knots $0 = \tau_1 < \tau_2 \ldots < \tau_{m-1} < \tau_m = T$:

$$q_i(\tau) = \sum_{j=0}^{m-1} q_j^i(\tau), \quad 1, \ldots, n, \tag{25}$$

where $q_j^i(\tau)$ is given by Eq. (19).

In the optimization of the trajectories, the decision variables are the B-spline coefficients γ_j^i. The dimensions of the design variable vector is $n(m+2)$.

2.3. Obstacle avoidance constraints

In this paper, the geometry of STANFORD robot has six rectangular prisms, four cylinders and one cube. Here the assumption is fifth joint and gripper are circumscribed by rectangular prism. Also the obstacles are only sphere (K_4) and rectangular prisms (K_2 and K_3). Hence for finding obstacle avoidance, we have considered 8 points (corner points) for cube and rectangular prism, 8 points (4 quadrant points on circle on each side) for cylinder and 4 quadrant points for sphere. So totally 88 points for STANFORD robot, 4 points for obstacle K_4 and 8 points for obstacles K_2 and K_3. All the points of STANFORD 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 Eq. (26). 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.:

$$d_q(t) = T_l(t)R_l(t)C_l. \tag{26}$$

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

$$\begin{bmatrix} x_l(t_i) \\ y_l(t_i) \\ z_l(t_i) \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & p_x(t) \\ 0 & 1 & 0 & p_y(t) \\ 0 & 0 & 1 & p_z(t) \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} \cos(\theta(t_i)) & -\sin(\theta(t_i)) & 0 & 0 \\ \sin(\theta(t_i)) & \cos(\theta(t_i)) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_l(t_0) \\ y_l(t_0) \\ z_l(t_0) \\ 1 \end{bmatrix} \tag{27}$$

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

Let a point on an obstacle as z_l and a point belongs to the robot as z_q. Let a set $d_l(t)$ defines 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 each time t. Here the obstacles are stationary. So changes in position of points of the robot due to the movement of the robot are found by Eq. (27). Now the software subroutine for obstacle avoidance finds 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. Proposed methods

In this section, two evolutionary optimization techniques (NSGA-II and MODE) used for obtaining the optimal trajectory planning of STANFORD robot with fixed obstacles are described.

3.1. Elitist non-dominated sorting genetic algorithm (NSGA-II)

Deb et al. (2002) proposed the NSGA-II algorithm. Essentially, NSGA-II differs from non-dominated sorting genetic algorithm (NSGA) implementation in a number of ways. First, NSGA-II uses an elite-preserving mechanism, thereby assuring preservation of previously found good solutions. Second, NSGA-II uses a fast non-dominated sorting procedure. Third, NSGA-II does not require any tunable parameter, thereby making the algorithm independent of the user.

Initially, a random parent population P_o created. The population is sorted out based on the non-domination. A special book-keeping procedure is used in order to reduce the computational complexity down to O(N^2). Each solution is assigned a fitness equal to its non-domination level (1 is the best level). Thus, minimization of fitness is assumed. Binary tournament selection, recombination, and mutation operators are used to create a child population Q_o of size N. Thereafter, we use the following algorithm in every generation. First, a combined population $R_i = P_iUQ_i$ is formed. This allows parent solutions to be compared with the child population, thereby ensuring elitism. The population R_i is of size $2N$. Then, the population R_i is sorted according to non-domination. The new parent population P_{i+1} is formed by adding solutions from the first front and continuing to other fronts successively till the size exceeds N. Thereafter, the solutions of the last accepted front are sorted according to a crowded comparison criterion and the first N points are picked. Since the diversity among the solutions is important, we use a partial order relation $\geqslant n$ as follows:

$$i \geqslant_n j \quad \text{if } i_{\text{rank}} < j_{\text{rank}} \quad \text{or} \quad i_{\text{rank}} = j_{\text{rank}} \quad \text{and} \quad i_{\text{fitness}} > j_{\text{fitness}}.$$

That is, between two solutions of differing nondomination ranks we prefer the point with the lower rank. Otherwise, if both the points belong to the same front then we prefer the point,

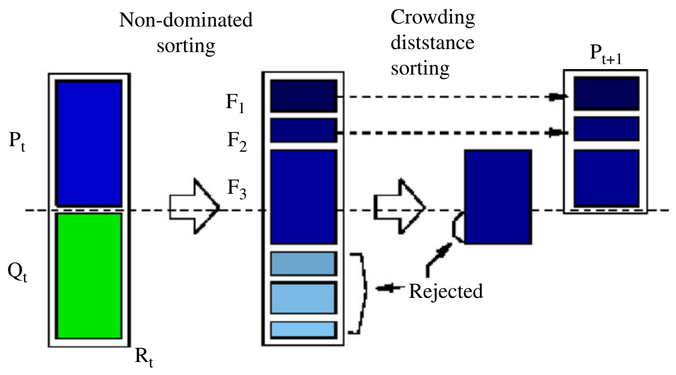

Fig. 4. An iteration of the NSGA-II procedure.

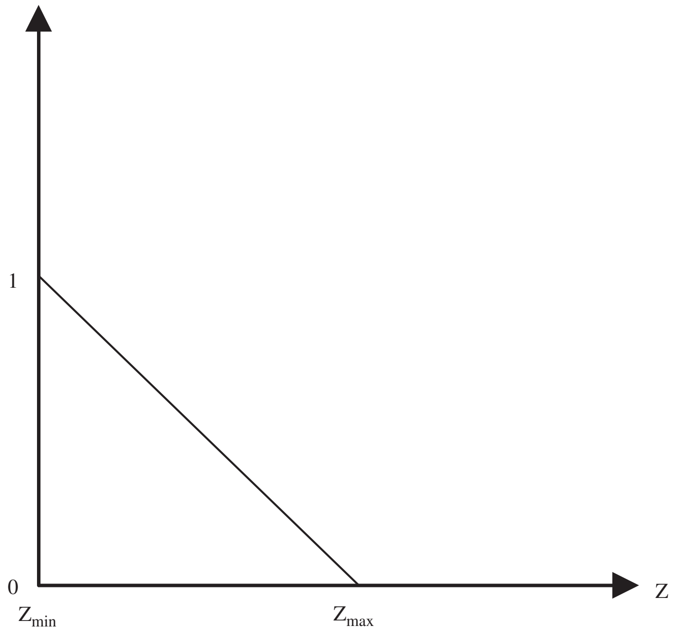

Fig. 5. Representation of fitness factor for minimization of an objective function.

which is located in a region with lesser number of points (or with larger crowded distance). This way, solutions from less dense regions in the search space are given importance in deciding which solutions to choose from R_i. This constructs the population P_{i+1}. This population of size N is now used for selection, crossover and mutation to create a new population Q_{i+1} of size N. We use a binary tournament selection operator but the selection criterion is now based on the crowded comparison operator $\geqslant_n$. The above procedure is continued for a specified number of generations.

It is clear from the above description that NSGA-II uses: (i) a faster non-dominated sorting approach, (ii) an elitist strategy, and (ii) no niching parameter. Diversity is preserved by the use of crowded comparison criterion in the tournament selection and in the phase of population reduction. NSGA-II has been shown to outperform other current elitist multi-objective evolutionary algorithms on a number of difficult test problems. Flowchart in Fig. 4 shows an iteration of the NSGA-II.

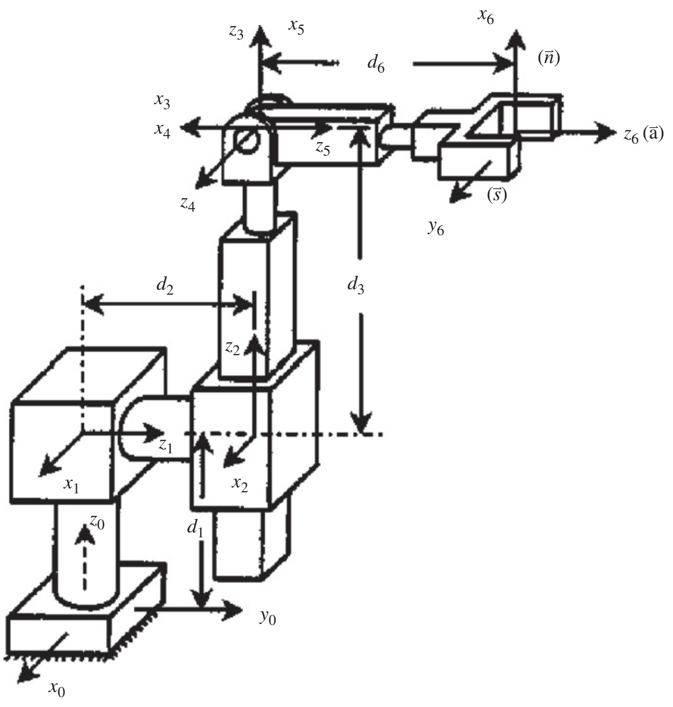

Fig. 6. STANFORD robot.

Table 1
Denavit–Hartenberg parameters for an STANFORD robot (Saramago and Steffen, 2001)

Joint no.	θ_i	α_i	a_i (m)	d_i (m)
1	θ_1	-90°	0	0
2	θ_2	90°	0	1.2
3	0	0	0	d_3
4	θ_4	-90°	0	0
5	θ_5	90°	0	0
6	θ_6	0	0	0

3.2. Multi-objective differential evolution (MODE)

MODE can be categorized into a class of floating-point encoded evolutionary algorithms. The theoretical framework of MODE is very simple and MODE is computationally inexpensive in terms of memory requirements and CPU times. Thus, nowadays MODE has gained much attention and wide application in a variety of fields. The advantages of MODE are its simple structure, ease of use, speed and robustness (Babu and Anbarasu, 2005).

In a multi-objective domain, the goal is to identify the Pareto optimal solution set. In this proposed multi-objective differential evolution (MODE), a Pareto-based approach is introduced to implement the selection of the best individuals. Firstly, a population of size, NP, is generated randomly and the fitness functions are evaluated. At a given generation of the evolutionary search, the population is sorted into several ranks based on dominance concept. Secondly, differential evolution (DE) operations are carried out over the individuals of the population. The fitness functions of the trial vectors, thus formed, are evaluated. One of the major differences between DE and MODE is that the trial vectors are not compared with the corresponding parent vectors. Instead, both the parent vectors and the trial vectors are combined to form a global population of size, 2NP. Then, the ranking of the global population is carried out followed by the crowding distance calculation. The best NP individuals are selected based on its ranking and crowding distance. These act as the parent vectors for the next generation.

The procedure is carried out up to maximum generation number. The Pseudo code for MODE algorithm (Babu and Anbarasu, 2005) is presented later:

The following assumes that we are minimizing all the objective functions, f_q:

(1) Generate box, P, of N_p parent vectors using a random-number code to generate the several real variables. These vectors are given a sequence (position) number as generated.
(2) Classify these vectors into fronts based on nondomination as follows:
   a. Create new (empty) box, P', of size, N_p;
   b. Transfer ith vector from P to P', starting with $i = 1$;
   c. Compare vector I with each member, say each member, say j, already present in P', one at a time;
   d. If i dominates over j (i.e., i is superior to or better than j in terms of all objective functions), remove the jth vector from P' and put it back in its original location in P;
   e. If i dominated over by j, remove i from P' and put it back in its position in P;
   f. If i and j are non-dominating (i.e., there is at least one objective function associated with i that is superior to/better than that of j), keep both i and j in P' (in sequence). Test for all j present in P';

Table 2
Geometric and inertial parameters of STANFORD robot (Saramago and Steffen, 2001)

Joint no.	M (kg)	$\bar{x}$ (m)	$\bar{y}$ (m)	$\bar{z}$ (m)	I_{xx} (kg m^2)	I_{yy} (kg m^2)	I_{zz} (kg m^2)	I_a (kg m^2)
1	9.29	0	1.75	−11.05	0.276	0.255	0.071	0.953
2	5.01	0	−10.54	0.0	0.108	0.018	0.100	2.193
3	4.25	0	0.0	−64.47	2.510	2.510	0.006	0.782
4	1.08	0	0.92	−0.54	0.002	0.001	0.001	0.106
5	0.63	0	0.0	5.66	0.003	0.003	0.0004	0.097
6	0.51	0	0.0	15.54	0.013	0.013	0.0003	0.020

Table 3
Limiting parameters used for STANFORD robot (Saramago and Steffen, 2001)

Constraint	Joint number					
	1	2	3	4	5	6
QC (rad)	3.1	3.1	1.5	3.1	3.1	3.1
VC (rad/s)	2.5	2.5	2.5	2.5	2.5	2.5
WC (rad/s^2)	9.5	9.5	9.5	9.5	9.5	9.5
JC (rad/s^3)	50	50	50	50	50	50
UC (N/m)	50	80	100	10	10	10

Joint 3—QC (m), VC (m/s), WC (m/s^2), JC (m/s^3), UC (N).

g. Repeat for next vector (in the sequence, without going back) in P till all N_p are tested. P' now contains a sub-box (of size $\leqslant N_p$) of nondominated vectors (a subset of P), referred to as the first front or sub-box. Assign it a rank number, I_{rank}, of I; and

h. Create subsequent fronts in (lower) sub-boxes of P', using Step 2b above (with the vectors remaining in P). Compare these members only with the members present in the current sub-box, and not with those in earlier (better) sub-boxes. Assign these $I_{\text{rank}} = 2, 3, \ldots$. Finally, we have all N_p vectors in P', boxed into one or more fronts.

Table 4
Joints displacement for STANFORD robot (Saramago and Steffen, 2001)

Knot	Joint 1 (rad)	Joint 2 (rad)	Joint 3 (m)	Joint 4 (rad)	Joint 5 (rad)	Joint 6 (rad)
1	0.175	0.175	0.80	0.087	0.174	0.105
2	(Extra knot)					
3	−0.056	0.356	1.12	−0.032	0.243	0.314
4	−0.164	0.515	1.44	−0.151	0.312	0.524
5	−0.138	0.679	1.43	−0.272	0.386	0.733
6	−0.256	1.043	1.19	−1.001	0.333	1.525
7	−1.013	0.920	0.30	−0.511	0.539	1.152
8	−1.199	0.908	1.06	−0.522	0.760	0.990
9	−1.329	1.260	0.92	−0.748	0.691	1.571
10	−1.474	1.486	0.80	−0.867	0.763	1.781
11	(Extra knot)					
12	−1.745	1.396	1.2	−0.853	1.078	−1.389

Table 5
Variables limits for strategy 2

Variable	Joint 1		Joint 2		Joint 3	
	Lower bound	Upper bound	Lower bound	Upper bound	Lower bound	Upper bound
γ^i_{-2}	0.18443503	0.38443503	0.11729967	0.31729967	0.64890760	0.84890760
γ^i_{-1}	0.06173106	0.26173106	−0.01246253	0.21246253	0.06569432	0.26569432
γ^i_0	0.07175214	0.27175214	−0.13759169	0.33759169	−0.28421954	0.48421954
γ^i_1	0.15768338	0.35768338	−0.26014745	0.46014745	−0.50092491	0.70092491
γ^i_2	0.28741420	0.48741420	−0.38162133	0.58162133	−0.64194904	0.84194904
γ^i_3	0.44444368	0.64444368	−0.50452678	0.70452678	−0.74197351	0.94197351
γ^i_4	0.32908217	0.82908217	−0.63907577	0.83907577	−0.83039481	1.03039481
γ^i_5	0.69291174	1.09291174	−0.83742785	1.03742785	−0.97459290	1.17459290
γ^i_6	1.00622795	2.03622795	−1.37344709	1.57344709	−1.47956861	1.67956861
γ^i_7	4.14358694	4.64358694	−3.67697051	4.17697051	−3.91004991	4.11004991
γ^i_8	14.6292457	18.6292457	−14.881515	16.8815155	−16.057886	18.0578860
γ^i_9	63.6720063	70.6720063	−69.740602	62.7406020	−67.158242	69.1502426

Variable	Joint 4		Joint 5		Joint 6	
	Lower bound	Upper bound	Lower bound	Upper bound	Lower bound	Upper bound
γ^i_{-2}	0.09184290	0.29184290	0.12409651	0.32409651	0.11797939	0.31797939
γ^i_{-1}	0.03041082	0.23041082	−0.00495111	0.20495111	0.04512642	0.24512642
γ^i_0	0.03456371	0.23456371	−0.11639756	0.31639756	0.06587169	0.26587169
γ^i_1	0.07622646	0.27622646	−0.21784587	0.41784587	0.14015748	0.34015748
γ^i_2	0.13952944	0.33952944	−0.38388727	0.51388727	0.24537981	0.44537981
γ^i_3	0.21630670	0.41630670	−0.40845163	0.60845163	0.37011316	0.57011316
γ^i_4	0.30664648	0.50664648	−0.51068537	0.71068537	0.50569105	0.71569105
γ^i_5	0.43569399	0.63569399	−0.66224534	0.86224534	0.62437125	0.92437125
γ^i_6	0.75002699	0.95002699	−1.07886635	1.27886635	1.03892045	1.43892045
γ^i_7	2.02318380	2.22318380	−2.88520306	3.08520306	3.33867121	3.83867121
γ^i_8	8.11874463	8.61874463	−11.6913083	13.6913083	13.4125361	15.4125361
γ^i_9	33.5252846	35.5252846	−48.5373897	50.5373897	55.4194857	57.4194857

(3) Spreading out: evaluate the crowding distance, $I_{i,\text{dist}}$, for the ith vector in any front, j, of P' using the following procedure:
   a. Rearrange all vectors in front j in ascending order of the values of any one (say, the qth) of their several objective functions (fitness functions). This provides a sequence, and, thus, defines the nearest neighbors of any vector in front j;
   b. Find the largest cuboid (rectangle for two fitness functions) enclosing vector i that just touches its nearest neighbors in the f-space;
   c. $I_{i,\text{dist}} = (1/2) \times$ (sum of all sides of this cuboid);
   d. Assign large values of I_i, dist to solutions at the boundaries (the convergence characteristics would be influenced by this choice).

(4) Perform DE operation over the NP target vectors in P' to generate NP trial vectors and store it in P''.
   a. Create new (empty) box, P'' of size N_p;
   b. Select a target vector, i in P', starting with $i = 1$;
   c. Choose two vectors, r_1 and r_2 at random from the NP vectors in P' and find the weighted difference. This is carried out by the following steps: (1) Generate two random numbers, (2) decide which two population members are to be selected, and (3) find the vector difference between the two vectors. Multiply this difference with F to obtain the weighted difference;
   d. Find the noisy random vector. This is done by (1) generate a random number, (2) choose a third random vector, r_3, from the NP vectors in P', and (3) add this vector to the weighted difference to obtain the noisy random vector;
   e. Perform crossover between the target vector and noisy random vector to find the trial vector and put it in box P''. This is carried out by (1) generate random numbers equal to the dimension of the problem, (2) for each of the dimensions: if random no. $>$CR; copy the value from the target vector, else copy the value from the noisy random vector into the trial vector and put it in box P'';

(5) Elitism: copy all the N_p parent vectors (P') and all the N_p trial vectors (P'') into box PT. Box PT has $2N_p$ vectors
   a. Reclassify these $2N_p$ vectors into fronts (box PT′) using only non-domination (as described in Step 2 above);
   b. Take the best N_p from box PT′ and put into box P'''. The following procedure is adopted to identify the better of the two chromosomes. Chromosome i is better than chromosome j if

$$I_{i,\text{rank}} \neq I_{j,\text{rank}} : I_{i,\text{rank}} < I_{j,\text{rank}},$$

$$I_{i,\text{rank}} = I_{j,\text{rank}} : I_{i,\text{dist}} > I_{j,\text{dist}}.$$

This completes one generation. Stop if appropriate criteria are met, e.g., the generation number$>$maximum number of generations (user specified). Else, Copy P''' into starting box P. Go to Step 2 above.

4. Performance measures and methods for multi-objective optimization

In this section, two methods and four performance metrics are recommended and applied to examine the strengths and

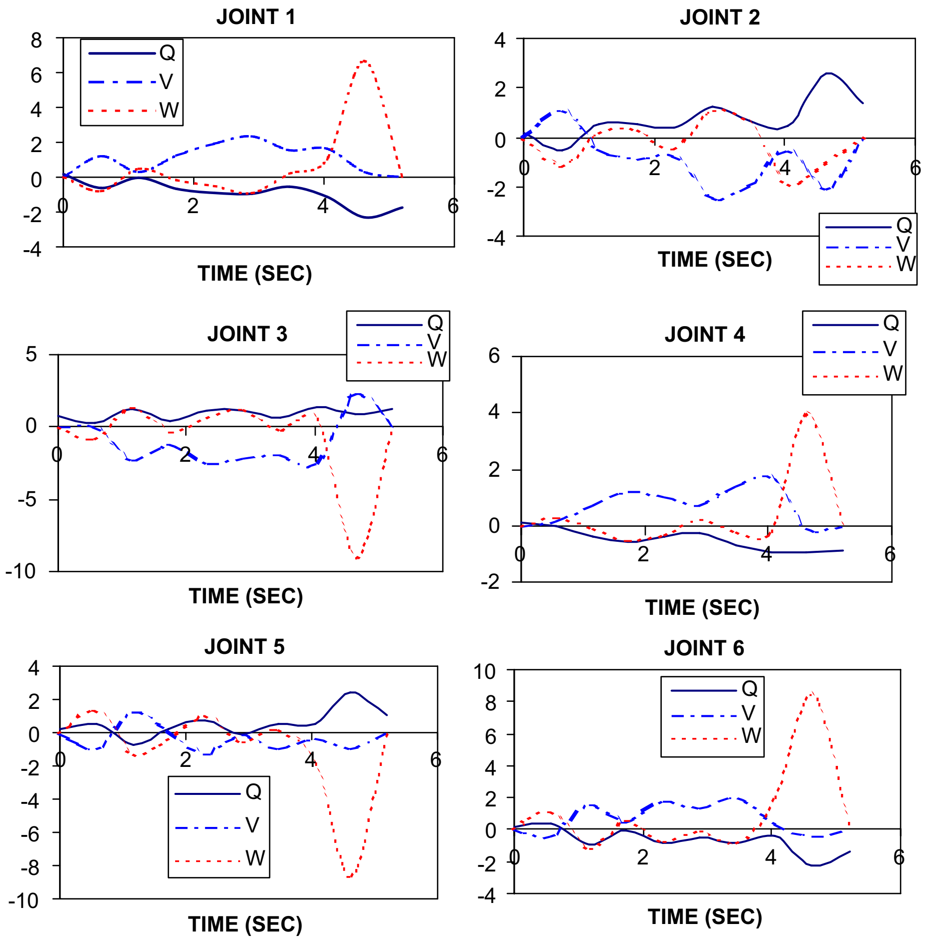

Fig. 7. Optimal motions of the robot joints obtained from NSGA-II for strategy 2.

weaknesses of the proposed multi-objective evolutionary algorithms. Two methods (normalized weighting objective functions method and average fitness factor method) are combinedly used to select best optimal solution from Pareto optimal front. Two multiobjective performance measures namely SSM and ratio of non-dominated individuals are used to evaluate the strength of Pareto optimal fronts. Two more multiobjective performance measures namely optimizer overhead and algorithm effort are used to find computational effort of NSGA-II and MODE algorithms. These methods and metrics are chosen since they have been widely used for performance comparisons in multi-objective optimization (Tan et al., 2002).

4.1. Normalized weighting objective functions method

Multiple objectives are combined into scalar objective via weight vector. Weights may be assigned through: direct assignment, eigenvector method, empty method, minimal information method and randomly determined or adaptively determined. If the objective functions are simply weighted and added to produce a single fitness, the function with largest range would dominate evolution. A poor input value for the objective with the larger range makes the overall value much worse than a poor value for the objective with smaller range. To avoid this, all objective functions are normalized to have a same range. For our problem, the combined objective function (f_c) is defined as follows:

$$\text{Minimize}\, f_c = \frac{\alpha_1 z_1}{N_1} + \frac{\alpha_2 z_2}{N_2} + \frac{\alpha_3 z_3}{N_3} \tag{28}$$

$\alpha_1 = 0.3$, $\alpha_2 = 0.3$ and $\alpha_3 = 0.4$ are weightage values for objective functions in strategies 1 and 2. $N_1 = 10$ and $N_2 = 1000$ and $N_3 = 1$ are normalizing parameters of objective functions in strategies 1 and 2 (average value of individual objective functions).

4.2. Average fitness factor (F_{avg}) method

The deterministic models proposed in the literature suffer from real world optimal trajectory planning limitation due to the fact that a decision maker does not have sufficient information related to the different criteria. So he may not know the weightage to be given to each objective function. In that situation, he may use the average fitness factor method proposed in this paper.

The average fitness factor given in Fig. 5 is a graphical representation of the magnitude of each input. The F value is 1 at Z_{min} and 0 at Z_{max} for minimization of an objective function and it is vice versa for maximizing an objective function.

The proposed fitness factor is as follows:

$$F_i = \frac{Z_{imax} - Z_i}{Z_{imax} - Z_{imin}} \quad \text{for minimization of objective function,}$$

where Z_i is the objective function (i is the objective function number, $i = 1, \ldots, 3$ for this problem); Z_{max} is the maximum objective function value; and Z_{min} is the minimum objective function value.

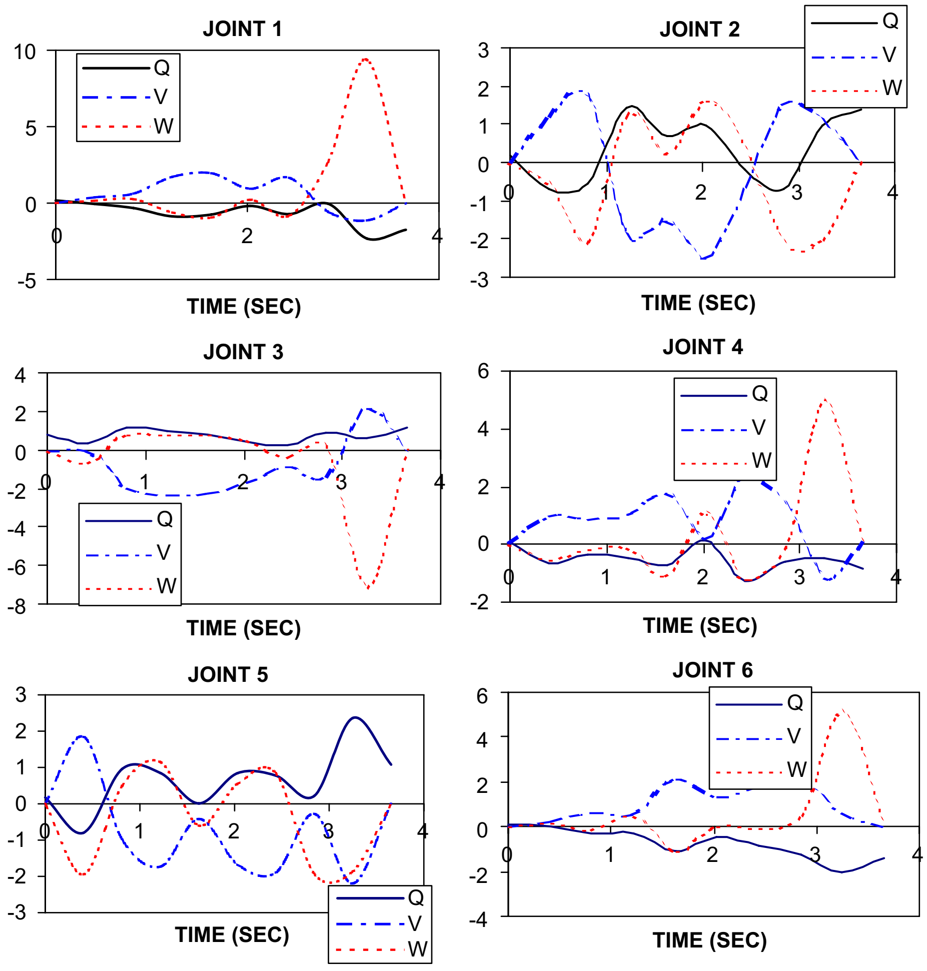

Fig. 8. Optimal motions of the robot joints obtained from MODE for strategy 2.

The solution that has the highest average fitness factor value (F_{avg}) is the best optimal solution, which gives a nondominated solution.

For our problem, the average fitness factor is defined as follows:

$$F_{avg} = \frac{F_1 + F_2 + F_3}{3.0}, \tag{29}$$

where $F_1 = (z_{1max} - z_1)/(z_{1max} - z_{1min})$, $F_2 = (z_{2max} - z_2)/(z_{2max} - z_{2min})$, $F_3 = (z_{3max} - z_3)/(z_{3max} - z_{3min})$.

4.3. Solution spread measure

While it is desirable to find more Pareto-optimal solutions, it is also desirable to find the ones scattered uniformly over the Pareto frontier in order to provide a variety of compromise solutions to the decision maker. SSM represents the distribution of the solutions along the Pareto front:

$$SSM = \frac{d_f + d_l + \sum_{i=1}^{N=1} |d_i = \bar{d}|}{d_f + d_l + (N-1)\bar{d}}, \tag{30}$$

where N is the number of solutions along the Pareto front so there are $(N-1)$ consecutive distances, d_i is the distance (in objective space) between each solution, $\bar{d}$ is the arithmetic mean of all d_i and d_f and d_l are the Euclidean distances between the extreme solutions and the boundary solutions of the obtained non-dominated set. Thus, a low performance measure characterizes an algorithm with good distribution capacity.

4.4. Ratio of non-dominated individuals

This performance metric is defined as the ratio of non-dominated individuals (RNI) for a given population X:

$$RNI(X) = \frac{\text{nondom\_indiv}}{P}, \tag{31}$$

where nondom_indiv is the number of non-dominated individuals in population X and P is the size of population X. Therefore, the value $RNI = 1$ means all the individuals in the population are non-dominated, and $RNI = 0$ represents the situation where none of the individuals in the population are non-dominated. Since a population size of more than zero is often desired, there is always at least one non-dominated individual in the population within the range of $0 < RNI < 1$.

4.5. Optimizer overhead

Total number of evaluations and total CPU time may be used for testing the algorithm. This would be useful in indicating how much time an optimization or simulated evolution process would take in real world and to indicate the amount of program overhead as a result of the optimization manipulations such as those by evolutionary algorithm operators. More quantitatively,

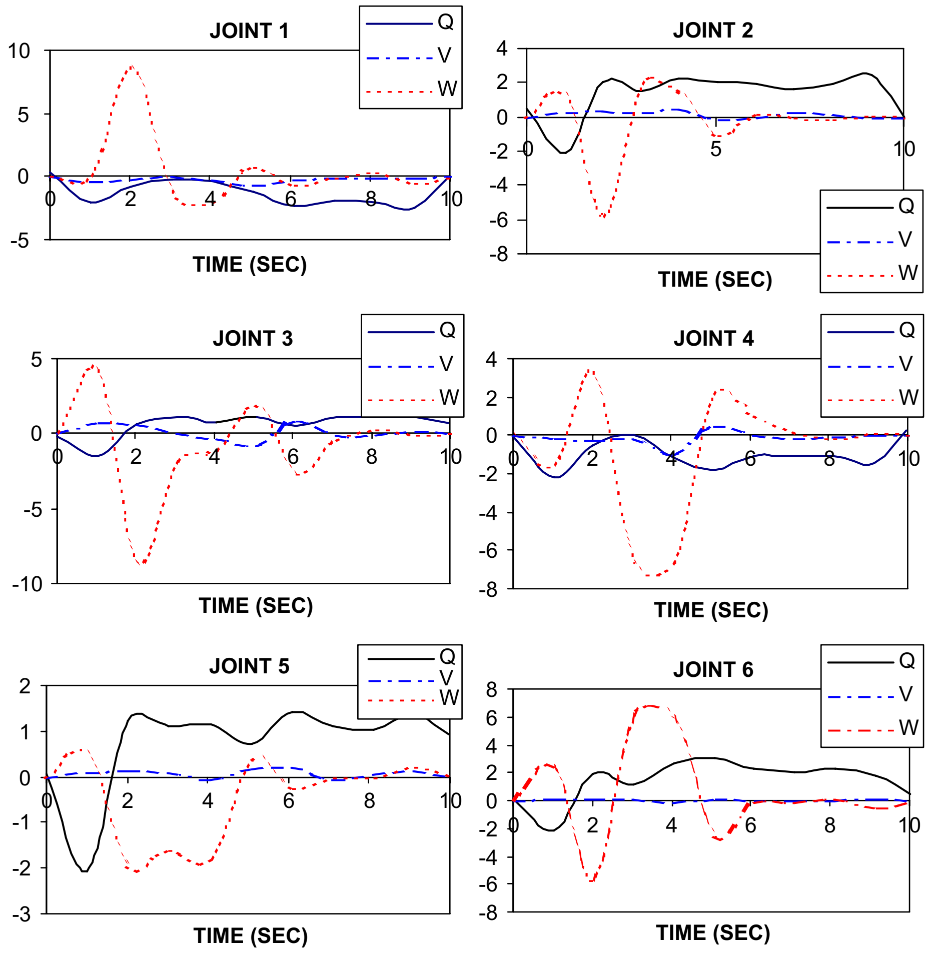

Fig. 9. Optimal motions of the robot joints obtained from NSGA-II for strategy 1.

the optimizer overhead (OO) may be calculated by

$$\text{Optimizer overhead} = \frac{T_{\text{Total}} - T_{\text{PFP}}}{T_{\text{PFP}}}, \tag{32}$$

where T_{Total} is the total time taken and T_{PFP} is the time taken for pure function evaluations. Thus, a value of zero indicates that an algorithm is efficient and does not have any overhead. However, this is an ideal case and is not practically reachable.

4.6. *Algorithm effort*

The performance in multi-objective optimization is often evaluated not only in terms of how the final pareto-front is, but also in terms of the computational effort required in obtaining the optimal solutions. For this purpose, the algorithm effort is defined as the ratio of the total number of functions evolutions N_{eval} over a fixed period of simulation time T_{run}:

$$\text{Algorithm effort} = \frac{T_{\text{run}}}{N_{\text{eval}}}, \quad (T_{\text{run}} > T_{\text{1stgen}}) \cap (T_{\text{eval}} \propto N_{\text{eval}}). \tag{33}$$

As shown in the above equation, for a fixed period of T_{run}, more number of function evolutions being performed indirectly indicates that less computational effort is required by the optimization algorithm and hence resulting in a smaller algorithm effort. The condition of $T_{\text{run}} > T_{\text{1stgen}}$, where T_{1stgen} is the computation time for the first generation, should be hold that T_{run} and N_{eval} are >0. This results algorithm effort is bounded in the range of $(0, \infty)$.

5. Numerical example

The optimal trajectory planning for STANFORD robot with six degree of freedom is dealt in this paper (Fig. 6).

The Denavit–Hartemberg parameters of the STANFORD robot are given in Table 1 (Saramago and Steffen, 2001). The dynamic characteristics of STANFORD robot are given in Table 2 (Saramago and Steffen, 2001). The robot is at rest initially, and comes to a full stop at the end of the trajectory. So, $\dot{q}_1 = \dot{q}_m = \ddot{q}_1 = \ddot{q}_m = 0$ for all joints. The velocity, acceleration, jerk and force constraints are given in Table 3 (Saramago and Steffen, 2001).

5.1. *Strategy 1*

Knots from the Cartesian path of the hand are given in Table 4. The optimum traveling time for the robot corresponds to the minimization of a set of time intervals $h_1, h_2, \ldots, h_{m-1}$. This technique leads to the maximization of the operation speed (Saramago and Steffen, 1998, 1999, 2001). The joints of the robot must be considered simultaneously.

The variables (time interval) limits used for strategy 1 are as follows: $0.25 \leqslant h_1 \leqslant 0.45$, $0.22 \leqslant h_2 \leqslant 0.68$, $0.09 \leqslant h_3 \leqslant 2.10$, $0.52 \leqslant h_4 \leqslant 1.43$, $0.16 \leqslant h_5 \leqslant 1.97$, $0.51 \leqslant h_6 \leqslant 1.52$, $0.88 \leqslant h_7 \leqslant 0.99$, $1.01 \leqslant h_8 \leqslant 2.01$, $0.52 \leqslant h_9 \leqslant 2.12$, $1.03 \leqslant h_{10} \leqslant 2.23$, $0.24 \leqslant h_{11} \leqslant 2.34$.

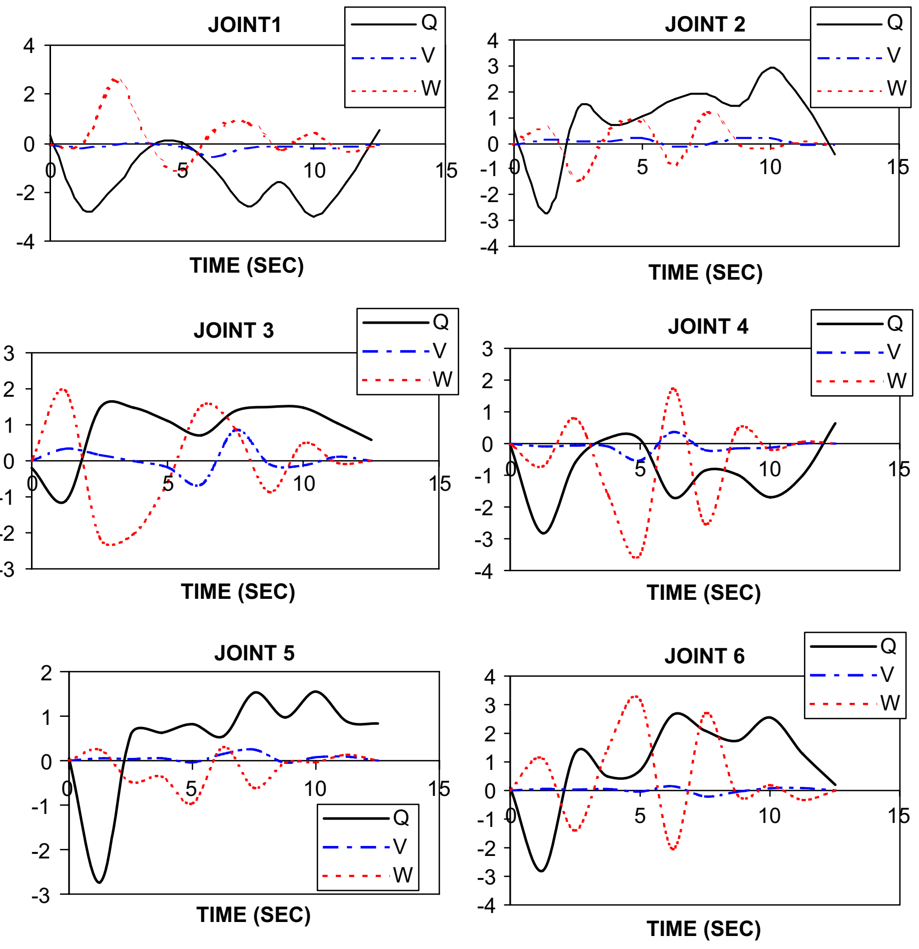

Fig. 10. Optimal motions of the robot joints obtained from MODE for strategy 1.

5.2. Strategy 2

In this application, the aim is to obtain optimal trajectory (ψ_1) of the end-effectors considering obstacles. The initial and final trajectory points of the end-effectors are: $q_1 = [0.30\text{rd}, 0.50\text{rd}, -0.20\text{rd}, -0.05\text{rd}, 0.05\text{rd}, 0.10\text{rd}]$ and $q_m = [0.51\text{rd}, -0.42\text{rd}, 0.58\text{rd}, 0.638\text{rd}, 0.835\text{rd}, 0.20\text{rd}]$. B-spline curve has four-knot points. B-spline coefficients (variables) limits used for the six joints are given in Table 5.

5.3. NSGA-II operators

The values of the parameter that have been used in the NSGA-II technique are: variable type = real variable, population size = 100, crossover probability = 0.7, real-parameter mutation probability = 0.01, real-parameter SBX parameter = 10, real-parameter mutation parameter = 100, and total number of generations = 100.

5.4. Multi-objective differential evolution operators

The values of the parameter that have been used in the proposed MODE technique are strategy = MODE/rand/1/bin, crossover constant CR = 0.9, number of population NP = 500, $F = 0.5$ and total number of generations = 100.

6. Results and discussion

From the proposed NSGA-II and MODE, 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 links are shown by Figs. 7 and 8 for the strategy 2 and Figs. 9 and 10 for the strategy 1. From Figs. 7–10, it is noted that the robot joints displacement, velocity, acceleration are within their limiting values. So the obtained trajectories are smooth. The optimum variables of best solution obtained from various techniques for the strategies 1 and 2 are given in Tables 6–8.

Optimum value of the objective functions of the best solution for strategies 1 and 2 obtained from the SUMT (Saramago and Steffen, 2001), NSGA-II and MODE are given in Table 9. The Pareto optimal fronts obtained from NSGA-II and MODE in strategies 1 and 2 are given in Figs. 11 and 12, respectively.

Table 6
Optimum variables for strategy 1

	h_1	h_2	h_3	h_4	h_5	h_6	h_7	h_8	h_9	h_{10}	h_{11}
MODE	0.41	0.66	1.95	1.35	1.36	1.35	0.89	1.52	0.94	1.78	0.30
NSGA-II	0.36	0.31	0.48	0.73	0.72	1.09	0.94	1.31	1.44	1.18	1.59

Table 7
Optimum variables for strategy 2 from MODE

Variable	Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6
γ^i_{-2}	0.2035135	0.21722654	0.7541260	0.19184290	0.22409651	0.2172539
γ^i_{-1}	0.22394264	0.05348743	0.12947036	0.03043552	0.14339086	0.19090760
γ^i_0	0.18118020	0.25570567	−0.12077296	0.23455965	0.24003441	0.14060400
γ^i_1	0.26785997	0.20331466	−0.49766009	0.27615953	−0.20298155	0.20282928
γ^i_2	0.46221550	−0.36668976	−0.62810945	0.33948423	−0.37835822	0.27990106
γ^i_3	0.50937447	−0.47272424	−0.61495414	0.41629592	−0.28868613	0.52145026
γ^i_4	0.41923362	−0.54733788	−0.52106388	0.30665341	−0.48133283	0.52844817
γ^i_5	0.58946250	−0.28223616	−0.51853862	0.63549420	−0.59882802	0.74546375
γ^i_6	1.05865055	−0.35070459	−1.03590088	0.83661414	−0.83620383	1.35758182
γ^i_7	4.18845965	−2.21417693	−3.03687862	2.04577136	−2.601031	3.63873836
γ^i_8	16.9794412	−10.2712284	−13.1796754	8.42475949	−8.75816068	13.6060684
γ^i_9	67.2458897	−50.4309285	−56.0936922	34.5286948	−35.9135327	56.9424306

Table 8
Optimum variables for strategy 2 from NSGA-II

Variable	Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6
γ^i_{-2}	0.244351	0.217021	0.7021540	0.1118429	0.3140965	0.201279
γ^i_{-1}	0.111439	0.034740	0.089929	0.144738	0.105529	0.232784
γ^i_0	0.259408	0.138924	−0.127737	0.123809	−0.061819	0.091267
γ^i_1	0.216808	−0.112566	−0.500438	0.228547	0.167384	0.334500
γ^i_2	0.396138	−0.2563855	−0.499176	0.327311	−0.015280	0.279952
γ^i_3	0.568161	−0.354225	−0.7135863	0.323714	−0.234120	0.466637
γ^i_4	0.677984	−0.631625	−0.829042	0.330822	−0.174536	0.5065748
γ^i_5	0.750362	−0.7018930	−0.855355	0.525591	−0.383671	0.6894307
γ^i_6	1.421522	−1.091552	−1.4689838	0.905506	−0.822831	1.1352605
γ^i_7	4.152590	−3.451813	−3.81979476	2.189411	−2.710173	3.825449
γ^i_8	15.882278	−12.697895	−16.053774	8.332870	−8.833946	14.830821
γ^i_9	65.752426	−58.779152	−67.155579	34.225063	−22.285528	56.982727

Table 9
Results obtained from SUMT (Saramago et al., 2001), NSGA-II and MODE algorithms

Technique	Time (z_1) (s)	Energy (z_2) (Nm)	Penalty for obstacle avoidance (z_3)	Combined objective function (f_c)
Strategy 1				
SUMT	8.6	22170	0	0.923100
NSGA-II	10.15	127.68	0	0.308330
MODE	12.51	124.7	0	0.379041
Strategy 2				
SUMT	10.5	29200	0	1.191000
NSGA-II	5.2	276.42	0	0.164293
MODE	3.64	284.3	0	0.117729

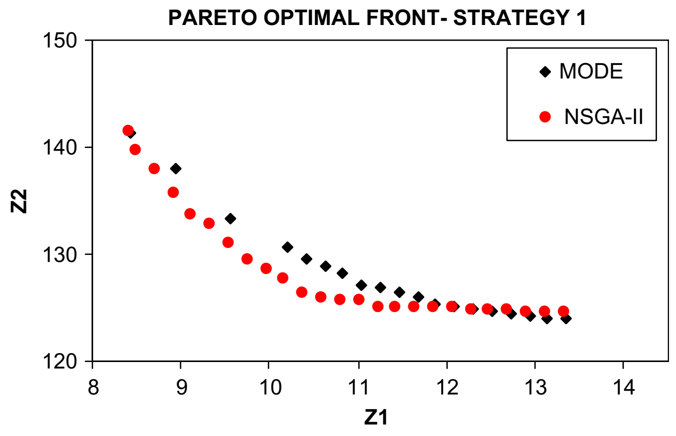

Fig. 11. Pareto optimal fronts obtained from NSGA-II and MODE for strategy 1.

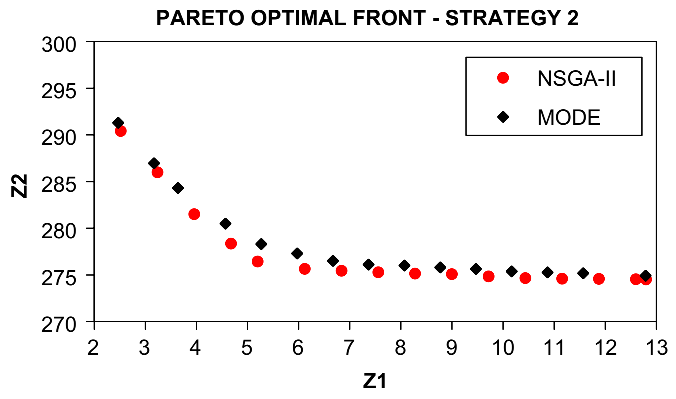

Fig. 12. Pareto optimal fronts obtained from NSGA-II and MODE for strategy 2.

The algorithm that gives minimum combined objective function (f_c), the highest average fitness factor value (F_{avg}), minimum SSM, maximum ratio of non-dominated solutions (RNI), minimum optimizer overhead (OO) and minimum algorithm effort is the best optimization algorithm. The results from NSGA-II and MODE for strategies 1 and 2 are listed in Tables 10–12.

From Tables 9–12, it is observed that the NSGA-II gives maximum ratio of non-dominated solutions (RNI) and minimum SSM than those of the MODE in both strategies. Also NSGA-II gives minimum combined objective function (f_c) and maximum average fitness factor value (F_{avg}) than those of MODE and SUMT (Saramago and Steffen, 2001) in strategy 1. But MODE gives minimum combined objective function (f_c) and maximum average fitness factor value (F_{avg}) than those of NSGA-II and SUMT (Saramago and Steffen, 2001) in strategy 2. Also MODE technique is much better than NSGA-II in terms of minimum optimizer overhead (OO) and minimum algorithm effort. It is observed that MODE technique converges quicker than NSGA-II. Also the computational time to find out Pareto optimal front by using a HP computer (with configuration of 640MB DDR RAM, 40 GB HDD, Pentium 4 Processor) in MODE is 1/3rd of that of NSGA-II. It is given in Table 13. So MODE is faster than NSGA-II. From Figs. 11 and 12, it is observed that NSGA-II gives more number of non-dominated solutions than MODE. Also Pareto optimal front from NSGA-II is better than that of MODE according to distance metric. So NSGA-II technique is best one for this multi-criterion optimization problem, if the user wants more number of solutions for his choice.

A threshold value is needed for SSM metric to determine if "uniformity" has been reached or not. Considerable research is going on in fixing threshold value. Ideal value for SSM metric is zero. In our problem, the value of SSM from NSGA-II and MODE are nearer to

Table 10
Average fitness factor from NSGA-II and MODE algorithms

	Strategy 1	Strategy 2
Z_{1max}	13.42	12.84
Z_{1min}	8.4	2.52
Z_{2max}	22170	29200
z_{2min}	124.4	274.5
F_{avg}		
SUMT	0.320053	0.075581
NSGA-II	0.550415	0.580081
MODE	0.393754	0.630378

Table 11
Algorithm effort obtained from NSGA-II and MODE algorithms

Technique	Simulation time, T_{run} (s)	No. of function evolution, N_{eval}	Algorithm effort
Strategy 1			
NSGA-II	2	91	0.02198
MODE	2	131	0.01527
Strategy 2			
NSGA-II	2	92	0.02174
MODE	2	138	0.01449

Table 12
SSM, RNI and OO obtained from NSGA-II and MODE algorithms

Strategy no.	Technique	SSM	RNI	OO
1	NSGA-II	0.80147	0.25	0.1416
1	MODE	0.98524	0.19	0.0473
2	NSGA-II	0.86238	0.16	0.1354
2	MODE	0.95746	0.15	0.0257

Table 13
Computational time to find Pareto optimal front

Strategy no.	Computational time (s)	
	NSGA-II	MODE
1	18	6
2	19	7

Table 14
Results of Mann–Whitney *U*-test

Strategy	n_1	n_2	R_1	R_2	Case	U	μ_u	σ_u	v
1	19	25	395	458	1	270	237.5	42.20486	0.770054
1	19	25	395	458	2	207	237.5	42.20486	−0.72267
2	15	16	181	231	1	179	120	25.29822	2.33218
2	15	16	181	231	2	129	120	25.29822	0.355756

zero. So we can say that uniformity is reached among the solutions of NSGA-II and MODE. In this paper, Mann–Whitney *U*-test with 95% confidence level is used to find which algorithm is best according to solution spreads. The procedure of the test is as follows:

1. Null hypothesis: Both NSGA-II and MODE are having better solution spreads.
   Alternate hypothesis. Either NSGA-II or MODE is having better solution spreads than other. So there are two cases.
   Case 1. If the calculated value of standard normal variate (v) is less than its tabulated value for R_1, we accept MODE is having best solution spreads than NSGA-II.
   Case 2. If the calculated value of standard normal variate (v) is less than its tabulated value for R_2, we accept NSGA-II is having best solution spreads than MODE.
2. First the distance between adjacent solutions of Pareto optimal fronts gave by NSGA-II and MODE are calculated. Then all solutions are arranged in ascending order.
3. Ranks are given to all solutions by taking both NSGA-II and MODE solutions in combined manner.
4. The values of n_1, n_2, R_1, R_2, U, μ_u, σ_u and v are found out.
   - n_1 = number of optimal solutions in Pareto optimal front given by NSGA-II,
   - n_2 = number of optimal solutions in Pareto optimal front given by MODE,
   - R_1 = sum of ranks of all optimal solutions in Pareto optimal front given by NSGA-II,
   - R_2 = sum of ranks of all optimal solutions in Pareto optimal front given by MODE,
   - $U = U$ statistic $= n_1n_2+n_1(n_1+1)/2-R$, We may use R_1 or R_2 in place of R. We have considered the two options as two cases. In case 1, R_1 is considered for R and in case 2, R_2 is considered for R.
   - μ_u = mean of the sampling data of U statistic $= n_1n_2/2$,
   - σ_u = standard error of the U statistic $= \sqrt{n_1n_2(n_1+n_2+1)/12}$,
   - and v = standard normal variate $= (U-\mu_u)/\sigma_u$.
5. The obtained results are given in Table 14.
6. In strategy 1, for both cases the calculated value of v is less than its tabulated value (critical values for 95% confidence level $(-1.64, 1.64)$). We accept the null hypothesis. So both NSGA-II and MODE are having best solution spreads in strategy 1.
7. In strategy 2, for case 2 the calculated value of v is less than its tabulated value. But for case 1, the calculated value of v is not less than its tabulated value. We don't accept the null hypothesis. We accept alternate hypothesis case 2. So NSGA-II is having best solution spreads than MODE in strategy 2.

7. Conclusions

Two new general strategies using NSGA-II and MODE for the off-line tridimensional optimal trajectory planning of the industrial robot manipulator (STANFORD robot) in the presence of fixed obstacles are presented. When dealing with fixed obstacles, both the objective functions and the constraint functions have to be updated simultaneously at each time instant. Two methods (normalized weighting objective functions method and average fitness factor method) are combinedly used to select best optimal solution from Pareto optimal fronts. Two multi-objective performance measures namely SSM and ratio of non-dominated individuals are used to evaluate the strength of Pareto optimal fronts. Two more multi-objective performance measures namely optimizer overhead and algorithm effort are used to find computational effort of NSGA-II and MODE algorithms. Two numerical examples demonstrated the efficiency of the proposed techniques. Both NSGA-II and MODE techniques are better than SUMT (Saramago and Steffen, 2001). It is observed that MODE technique converges quickly than NSGA-II. Also the computational time to find Pareto optimal front in MODE is one-third of that of the NSGA-II (Table 13). So MODE is faster than NSGA-II. From Figs. 11 and 12, it is observed that NSGA-II gives more number of non-dominated solutions than MODE. Also Pareto optimal front from NSGA-II is better than that of MODE according to distance metric. So NSGA-II technique is best for this multi-criterion optimization problem, if the user wants more number of solutions for his choice. To get more accurate and high flexible trajectory, NURBS function with more number of control points will be used to represent the trajectory in future work. This work opens the door for further investigations on how the evolutionary optimization techniques can be used to solve complex problems.

References

Babu, B.V., Anbarasu, B., 2005. Multi-objective differential evolution (MODE): an evolutionary algorithm for multi-objective optimization problems (MOOPs) 〈http://discovery.bitspilani.ac.in/discipline/chemical/BVb/publications/html〉.
Bäck, T., Hammel, U., Schwefel, H.P., 1997. Evolutionary computation: comments on the history and current state. IEEE Transaction on Evolutionary Computation 1 (1), 3–17.
Chettibi, T., Lehtihet, H.E., Haddad, M., Hanchi, S., 2004. Minimum cost trajectory planning for industrial robots. European Journal of Mechanics A/Solids 23, 703–715.
Coello, C., Carlos, A., 1999. A comprehensive survey of evolutionary-based multi-objective optimization techniques. Knowledge Information Systems 1 (3), 269–308.
Deb, K., Pratap, A., Agarwal, S., Meyarivan, T., 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6 (2), 182–197.
Fonseca, C.M., Fleming, P.J., 1995. An overview of evolutionary algorithms in multi-objective optimization. Evolutionary Computations Journal 3 (1), 1–16.
Horn, J., Nafploitis, N., Goldberg, D., 1994. A niched Pareto genetic algorithm for multi-objective optimization. In: Proceedings of the First IEEE Conference on Evolutionary Computation, pp. 82–87.
Saramago, S.F.P., Steffen Jr., V., 1998. Optimization of the trajectory planning of robot manipulators taking into account the dynamics of the system. Mechanism and Machine Theory 33 (7), 883–894.
Saramago, S.F.P., Steffen Jr., V., 1999. Dynamic optimization for the trajectory planning of robot manipulators in the presence of obstacles. Journal of Brazilian Society of Mechanical Sciences 21 (3), 1–17.
Saramago, S.F.P., Steffen Jr., V., 2001. Trajectory modeling of robot manipulators in the presence of obstacles. Journal of Optimization Theory and Applications 110 (1), 17–34.
Shiller, Z., Dubowsky, S., 1991. The global time-optimal motions of robotic manipulators in the presence of obstacles. IEEE Transaction of Robotics and Automation 7 (6), 785–797.
Tan, K.C., Lee, T.H., Khor, E.F., 2002. Evolutionary algorithms for multi-objective optimization: performance assessments and computations. Artificial Intelligence Review 17, 253–290.