Intelligent Service Robotics (2022) 15:627–648
https://doi.org/10.1007/s11370-022-00440-8

ORIGINAL RESEARCH PAPER

Optimal trajectory generation in joint space for 6R industrial serial robots using cuckoo search algorithm

Oğuzhan Karahan[1] · Hasan Karci[1] · Ali Tangel[1]

Received: 17 March 2022 / Accepted: 7 August 2022 / Published online: 6 September 2022

Abstract
In this paper, an optimal trajectory generation approach is proposed based on optimal time, optimal jerk and optimal time-jerk by utilizing the interpolation spline methods. The methods including cubic spline, trigonometric spline and a combination of cubic spline and 7th-order polynomial are used for generating the trajectory in joint space for robot manipulators. Cuckoo search (CS) optimization algorithm is chosen to optimize the joint trajectories based on three objectives, namely, minimizing total travelling time, minimizing mean jerk and minimizing a weighted sum of the travelling time and the mean jerk along the whole trajectory. The spline methods have been applied on PUMA-robot for optimizing the joint trajectories with the CS algorithm based on each objective. Moreover, results from the proposed algorithm have been compared with that of the algorithms suggested by earlier studies in terms of three objectives. With the trajectory planning methods, the joint velocities, accelerations and jerks along the whole trajectory optimized by CS meet the requirements of the kinematic constraints in case of each objective. Simulation results validated that the used trajectory planning methods based on the proposed algorithm are very effective in comparison with the same methods based on the algorithms proposed by earlier authors.

Keywords Trajectory generation · Interpolation spline methods · Industrial robots · CS

1 Introduction

Use of industrial robots in production systems with the development of automation technology has grown tremendously over the past few decades. In particular, the robotic manipulators in industrial assembly lines and production systems are widely used today due to their ability to achieve particular tasks with improved speed, reliability and quality. In these industrial activities, trajectory planning for industrial robots working in industrial environments is very important issue. Accordingly, the trajectory planning problem in industrial robotic applications has recently attracted great attention of many researchers.

The capability of trajectory planning algorithms is restricted by the time needed for carrying out the trajectory and also by physical limitations of the robot [1]. Most of the trajectory planning algorithms have proposed an objective function consisting of execution time, actuator effort and absolute value of the jerk for performing certain criterion such as minimum execution time, minimum jerk and minimum kinetic energy consumed by the robot [2]. The concept of minimum-time trajectory was firstly introduced by Bobrow et al. [3], Shin et al. [4] and Chen et al. [5]. The alternative approach instead of the execution time introduces jerk optimization. With the help of the jerk optimization, the position errors during movement, wear of the mechanical parts of the robot, the stresses to the robot actuators and the resonant vibrations in the robot's structure can be reduced while performing trajectory tracking [6]. Thus, in case of requiring soft and continuous movement, jerk constrained trajectory planning should be performed. The trajectory planning problem based on seeking jerk optimal trajectories was firstly introduced by Kyriakopoulos et al. [7]. In order to enhance the efficiency and smoothness of the robot's movement and reduce the excitation of resonance frequency, the combination of minimum time with minimum jerk for the trajectory planning has been recently focused by various researchers. For a more recent example of this approach, a study presented

✉ Hasan Karci
hsnkarci06@gmail.com

Oğuzhan Karahan
oguzhan.karahan@kocaeli.edu.tr

Ali Tangel
atangel@kocaeli.edu.tr

1 Department of Electronics and Communication Engineering, Kocaeli University, Kocaeli University, Kocaeli, Turkey

by Mattmüller and Gisler [8] can be found. The authors examine the velocity and acceleration as well as jerk. For solving the trajectory planning problem, another different approach has been performed on minimization of the torque and the energy consumed instead of the execution time or the jerk. This approach provides a better way to make the profile of the trajectory smooth with smaller stresses on the actuators and mechanical structure of the robot. The early examples related to the minimum energy trajectory planning for the robot can be found in [9, 10].

In order to enhance the manufacturing productivity and the movement stability, the trajectory planning with short execution time and smooth profile is required. Spline functions have a major role for improving smooth trajectory to ensure velocity continuity, acceleration continuity or jerk continuity in the motion of industrial robots. Lin et al. [11] constructed joint trajectories for industrial robots using cubic splines based on minimum time trajectory through polyhedron search method. Aribowo and Terashima [12] proposed a combination of input shaping with cubic spline optimization for generating the minimum time trajectory of the robot arm. Sequential Quadratic Programming (SQP) was used for solving the constrained trajectory planning optimization problem outlined in that paper. Gasparetto and Zanotto [13] suggested a technique based on cubic B-spline for realizing the jerk continuity in the joint trajectory planning by minimizing of both time and jerk during optimization with the SQP. In [14], the same authors analyzed the results obtained from running the SQP algorithm based on an objective function comprising a weighted sum of time and jerk using cubic splines and fifth-order B-splines. Liu et al. [15] presented a smooth trajectory planning approach using a combination of the planning with multi-degree splines in Cartesian space and multi-degree B-splines in joint space for robot manipulators with kinematic constraints. In that paper, for solving the minimum execution time problem during optimal trajectory planning, the SQP method was used. Chen and Chen [16] used the B-spline curve by fitting this curve through an approximation method, and then in order to plan the motion trajectory, the intermediate points were interpolated by using S-curve feed-rate profile for the trajectories of the tool tip and tool axis of a 6-DOF robot manipulator. Simon and Isik [17] suggested an interpolation of a robot joint trajectory is realized using trigonometric splines. This paper introduces a trajectory interpolation algorithm, discusses a Quadratic Programming (QP) method for path optimization and includes examples. On the other hand, for the 6-DOF robot manipulator, Perumaal and Jawahar [18] presented an automated trajectory planner in order to obtain a smooth and minimum-time trajectory based on the synchronized trigonometric S-curve trajectory technique with jerk constraints. Moreover, in that paper, suitable examples and comparisons with the cubic spline-based trajectory were demonstrated. With fifth-order B-spline or quantic polynomial spline proposed in [19, 20], continuous motion in the position, velocity, acceleration and jerk is provided. But, for obtaining the smooth trajectory, the least six coefficients must be solved. Accordingly, computation complexity may arise. To overcome this problem, Boryga and Graboś [21] designed a trajectory planning mode with high-degree polynomial for serial link manipulators. In that paper, the acceleration profile of the tool tip was planned by means of the determination of only one polynomial coefficient using the property of the root multiplicity.

Recently, interpolation functions have been utilized for generating a trajectory under kinematical or dynamical constraints in joint space for industrial robots. In the literature on the trajectory planning problem, the optimal motion planning for robotic manipulators can be considered as an optimization problem applied to the nature inspired optimization algorithms using an objective function for minimizing execution (travelling) time, maximum jerk or both of them and energy, subject to kinematic or dynamic constraints of all joints. Several natures inspired optimization algorithms developed by researchers have been successfully applied to the trajectory optimization for industrial robots.

Machmudah et al. [22] employed the GA and Particle Swarm Optimization (PSO) algorithms by minimizing the total travelling time and the torque under kinematic and dynamic limitations in order to find the feasible joint trajectory with the high-degree polynomial curve for robotic arm in the obstacle environment. Huang et al. [23] proposed optimal time-jerk trajectory planning in joint space using 5th-order B-spline interpolation technique for robotic manipulators. In that paper, for solving the time-jerk optimal trajectory planning problem, NSGA-II was employed by minimizing an objective function consisting of travelling time and mean jerk under the kinematic limits of velocity, acceleration and jerk. For a space robotic system composing of the 6-DOF space craft and the 7-DOF redundant manipulator, Wang et al. [24] employed PSO to find the optimal solutions using the execution time criterion to construct fifth-order Bèzier curve in joint space under the velocity and acceleration boundaries. Kucuk [25] developed a minimum time smooth motion trajectory in joint space by combining cubic spline with the 7th-order polynomial for serial and parallel manipulators, and in that paper PSO is chosen as an optimization algorithm because of its easy implementation and successful optimization performance. Lu et al. [26] realized time-jerk optimal trajectory under kinematic constraints by adopting cubic spline in joint space for a 7 DOF robot. In that paper, the augmented Lagrange constrained particle swarm optimization (ALCPSO) algorithm was proposed to obtain minimum travelling time and minimum jerk. Rout et al. [27] described an approach for optimal trajectory planning in joint space via cubic spline by using TLBO algorithm based on minimization of total travel time and squared jerk for welding robot. Savsani et al. [28] employed the TLBO and ABC algorithms by minimizing travelling time, travelling distance and total joint Cartesian lengths for planning point to point trajectory in joint space with high-order polynomial spline for a 3-DOF robotic arm. In [29], the same authors presented a comparative study of the proposed heuristic solution approaches for planning the trajectories in joint space based on high-order polynomial spline by minimizing three different objective functions for a 3-DOF robotic arm. Wang et al. [30] presented an optimal joint trajectory planning method using DE algorithm with specific objectives under kinematic limitations through Bèzier curve for a 7-DOF kinematically redundant manipulator. Bureeat et al. [31] proposed the robot optimum trajectory planning in joint space via the five-order polynomial function for a 6-DOF robot. In that paper, for the robot optimum trajectory planning, the hybrid real code population based on incremental learning and DE was used based on the two objective functions as minimizing trajectory time and jerk under kinematic constraints of all joints.

Motivated by the above-mentioned studies, the central objective of this work is to present a comparative study of the S-curve profiles based on interpolation splines, i.e. cubic polynomial model, high-order polynomial model and trigonometric curve model for point-to-point movements. In this study, by using each polynomial model and trigonometric model, firstly the travelling time, secondly mean jerk and thirdly both of them are considered for planning the time optimal trajectory, jerk optimal trajectory and optimal time-jerk trajectory for 6-DOF industrial robots. For obtaining the optimal trajectory from initial to intermediate positions and from intermediate to final position in joint space under kinematic constraints, the CS-based optimization algorithm is employed by minimizing three objective functions, namely, travelling time objective function, mean jerk objective function and the time-jerk objective function. Considering the concept of planning the trajectory optimization addressed in aforementioned works, the main contributions of this study are: (I) to provide independent comparison of the used interpolation methods for solving optimal joint trajectory planning problem, (II) to present the values of maximum joint velocities, accelerations, jerks generated by the used methods for each joint trajectory, (III) to demonstrate the maximal values of all kinematic variables of each joint along the path segments according to the combination of cubic spline and 7th-order polynomial, (IV) to compare the obtained results with that of the other optimization algorithms proposed in the literature based on the same interpolation methods for each joint along the path from initial to the final point, (V) to compare the resulting snaps based on the trigonometric spline and the combination of cubic spline and 7th-order polynomial at the first and last segment for optimal time, optimal jerk and optimal time-jerk trajectory planning.

The rest of the paper is organized as follows: Sect. 2 describes the problem statements, Sect. 3 introduces trajectory planning approaches, Sect. 4 explains the used optimization algorithms. In Sect. 5, implementation of methodology is presented. Some design examples for the proposed theory and simulation results are given in Sects. 6, and 7 comprises of the discussion and conclusion.

2 Problem statements

Trajectory planning for industrial robots can be specified either in Cartesian space or joint space. The trajectory of end-effector in Cartesian space can be constructed based on a set of consecutive intermediate points between the start and end points. Trajectory via-points are defined as the start and end points and the intermediate points formed between them. In view of the kinematic and dynamic constraints imposed to the robot joints, complex modelling and heavy mathematical computations appear in Cartesian space, but trajectory planning in joint space is not the case. Therefore, planning a trajectory in joint space is preferred rather than trajectory planning in Cartesian space. By solving the inverse kinematic problem related to via-points determined in the Cartesian space, the corresponding counterparts in joint space can be found. Thus, a suitable trajectory is generated between two consecutive points under kinematic constraints such as displacement, velocity, acceleration and jerk.

In joint trajectory planning between the start position and the final position, the motion constraints specified in datasheet of the robot manipulator must be considered for each joint. Kinematic constraints for each joint in trajectory planning are defined as:

$$\begin{cases} \left|\theta_j(t)\right| \leq \theta_j^{\max}, \; j = 1, \ldots, N \\ \left|\dot{\theta}_j(t)\right| \leq \dot{\theta}_j^{\max}, \; j = 1, \ldots, N \\ \left|\ddot{\theta}_j(t)\right| \leq \ddot{\theta}_j^{\max}, \; j = 1, \ldots, N \\ \left|\dddot{\theta}_j(t)\right| \leq \dddot{\theta}_j^{\max}, \; j = 1, \ldots, N \end{cases} \tag{1}$$

The optimization formulas used in this work are presented in Table 1.

In the trajectory planning, it is expected that smooth motion trajectory is generated by minimizing mechanical vibration and travelling time for achieving greater productivity with applying simultaneously kinematic limitations. Accordingly, considering the time optimality and jerk limitation at the same time, the constrained time-jerk optimal trajectory can be planned for robotic manipulators. In this paper, for obtaining the time optimal and/or minimum jerk trajectory, the most significant criteria found in the literature are used for the optimal trajectory planning. The optimization

Table 1 Meaning of symbols in the optimization formulations

Symbol	Meaning	Symbol	Meaning
N	Number of robot joints	$\dot{\theta}_j(t)$	Velocity of the jth joint
n	Number of segments	$\ddot{\theta}_j(t)$	Acceleration of the jth joint
h_i	Time interval between the ith via-point and $(i+1)$th via-point	$\dddot{\theta}_j(t)$	Jerk of the jth joint
t_f	Total travelling time of the trajectory	$\theta_j^{\max}$	Displacement constraint for the jth joint
K_t	Weight of the term proportional to the travelling time	$\dot{\theta}_j^{\max}$	Velocity constraint for the jth joint
K_j	Weight of the term proportional to the jerk	$\ddot{\theta}_j^{\max}$	Acceleration constraint for the jth joint
$\theta_j(t)$	Displacement of the jth joint	$\dddot{\theta}_j^{\max}$	Jerk constraint for the jth joint

problem is solved under kinematic constraints given above by minimizing the equations defined as follows:

$$f_1 = \sum_{i=1}^{n} h_i \tag{2}$$

$$f_2 = \sum_{j=1}^{N} \int_0^{t_f} \left(\dddot{\theta}_j(t)\right)^2 \mathrm{d}t \tag{3}$$

$$f_3 = K_t N \sum_{i=1}^{n} h_i + K_j \sum_{j=1}^{N} \int_0^{t_f} \left(\dddot{\theta}_j(t)\right)^2 \mathrm{d}t \tag{4}$$

where $t_f = h_1 + h_2 + \cdots + h_n$ in Eqs. 3 and 4 represents the total trajectory execution time, and also the meanings of the other symbols used in the optimization formulas given above are given in Table 1. As can be observed from the objective functions given Eqs. 2, 3 and 4, the total duration of the trajectory is aimed to be minimized by function f_1. However, shorter travelling time results in higher jerks. On the other hand, from the motion profile based on the function f_2, the jerk can be obtained as close to zero. From the results obtained with the function f_3 that provides a trade-off between speed and smoothness, minimization of the former and second term can produce a smooth motion trajectory with the maximum values of velocity, acceleration and jerk.

In this work, the trajectory planning is applied between any pair of consecutive intermediate points based on minimization of each objective function given above. Also, in order to apply the optimization algorithm, spline functions such as cubic spline, high-order polynomial and trigonometric spline for the interpolation are chosen.

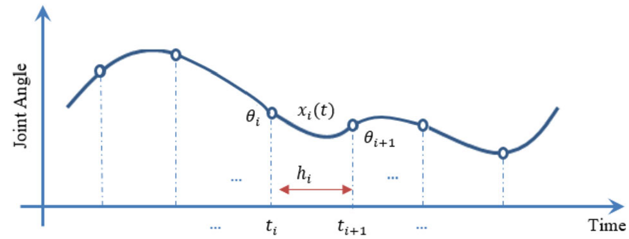

Fig. 1 Cubic spline interpolation of the via-points in the joint trajectory

Fig. 2 PUMA 560 industrial robot manipulator [42]

3 Trajectory planning approach

The trajectory planning of the robot manipulator is usually based on the movement of the end-effector in Cartesian space. Accordingly, from the points specified in Cartesian space, the displacements in Joint space can be calculated by using the inverse kinematics. Consequently, the joint displacement calculated by the inverse kinematics constitutes the trajectory of the motion between given points within the determined time. For obtaining smooth and continuous movements between these via-points in Joint space of the robot manipulator, the appropriate interpolation splines are used. For this study, the spline functions widely used in trajectory planning are outlined in the following sections.

3.1 Cubic splines

For trajectory interpolation, the cubic spline is frequently used in the trajectory planning methods due to its easy and fast mathematical calculation. Also, it can produce continuous velocity and acceleration at every via-point [32]. In Cartesian space, intermediate points are kinematically defined between start and end points. By means of inverse kinemat-

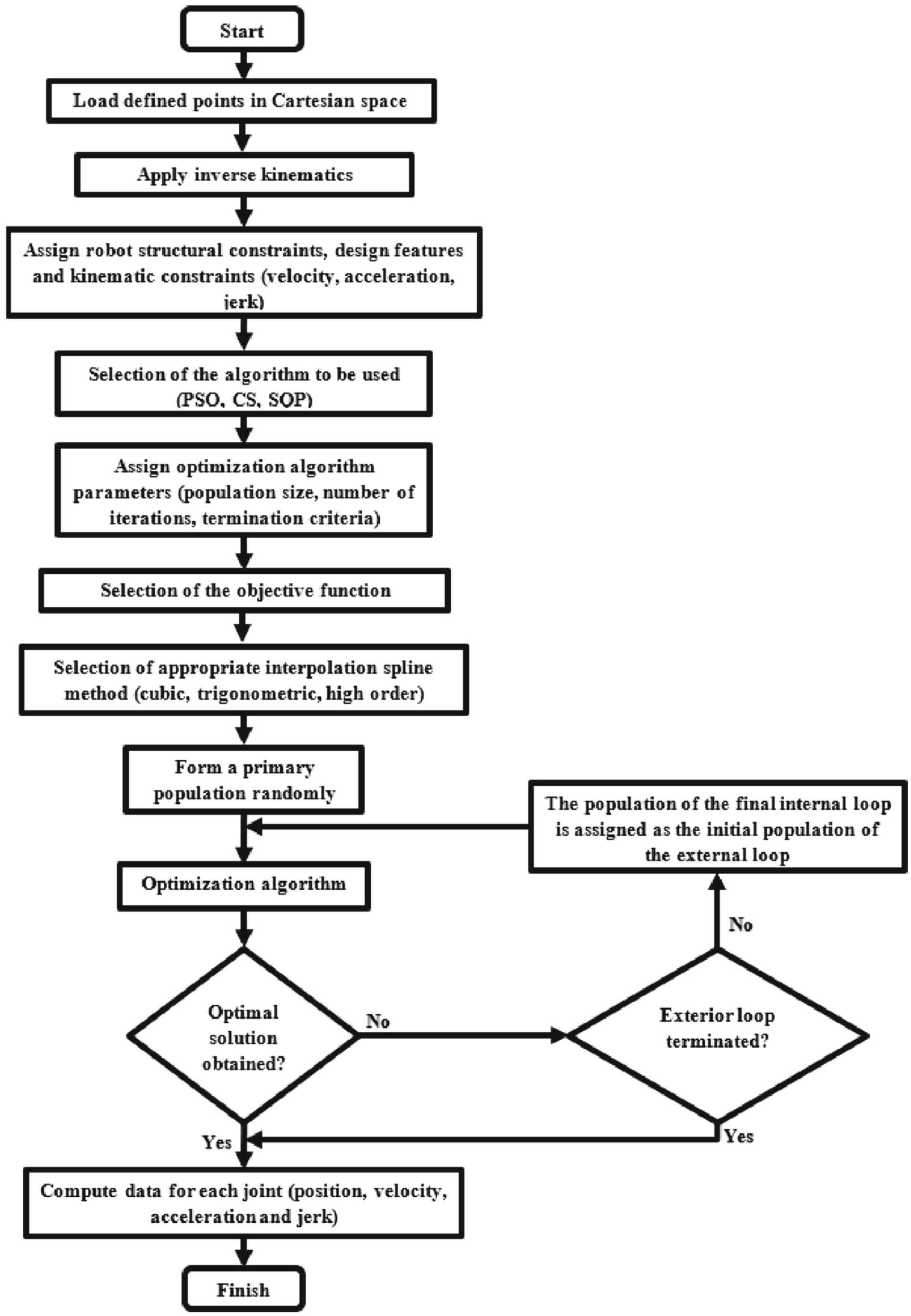

Fig. 3 Flowchart of the optimal trajectory planning for a robotic arm

ics, these nodes defined in Cartesian space are converted into Joint space. The motion trajectory consisting of multiple segments is planned with the cubic spline for each joint.

The cubic spline piecewise polynomials can be given as follows:

$$x_i(t) = a_i(t - t_i)^3 + b_i(t - t_i)^2 + c_i(t - t_i) + d_i \quad i = 1, 2, \ldots, n \tag{5}$$

where the i^{th} cubic polynomial function is shown by $x_i(t)$ with the time interval from t_i to t_{i+1}. The coefficients of the cubic polynomial function are a_i, b_i, c_i and d_i. By taking the first and the second derivatives of the cubic polynomial function, the velocity and acceleration of each via-point can be obtained.

The trajectory in joint space with n cubic segments has $n + 1$ prespecified joint angles [33]. Figure 1 presents such a joint trajectory, in which each segment is represented by a cubic polynomial function of time. The generic component $\theta_i(i = 1, \ldots, n + 1)$ represents the angle value of the i^{th} via-point, and h_i denotes the length of time interval $t_{i+1} - t_i$.

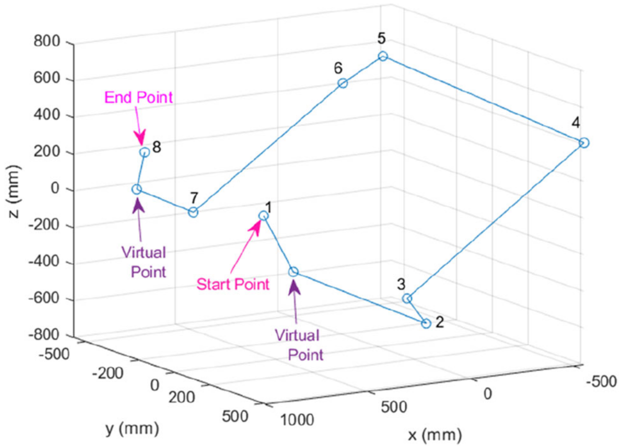

Fig. 4 Path to be followed by end-effector of the robot in Cartesian space

Table 2 The joint via-points for trajectory planning

Via-points	Joints (°)					
	1	2	3	4	5	6
1	10	15	45	5	10	6
Virtual						
2	60	25	180	20	30	40
3	75	30	200	60	− 40	80
4	130	− 45	120	110	− 60	70
5	110	− 55	15	20	10	− 10
6	100	− 70	− 10	60	50	10
7	− 10	− 10	100	− 100	− 40	30
Virtual						
8	− 50	10	50	− 30	10	20

Table 3 Kinematical constraints of the robot joints

Joints no	1	2	3	4	5	6
Velocity (°/s)	100	95	100	150	130	110
Acceleration (°/s^2)	60	60	75	70	90	80
Jerk (°/s^3)	60	66	85	70	75	70

3.2 Higher-order polynomial splines

In trajectory planning, it is possible to generate the smooth path and to improve the movement stability interpolating the trajectory by means of a reasonable-order polynomial function. The smooth trajectory can be achieved by obtaining the continuity of acceleration and jerk. At the beginning and ending points, cubic polynomial curve provides continuous displacement, velocity and acceleration, but this curve does not support the continuity of the jerk. Hence, this case may cause vibrations especially at the initial and the rest points. Accordingly, accurate trajectory may be reduced by this discontinuous jerk. At the beginning and ending of the trajectory, zero velocity, acceleration and jerk are desired especially by the robot designers for obtaining the faster and more accurately trajectory. It is possible to specify zero jerk at the start and end points of the trajectory by using 5th-, 7th- and 9th-order polynomials. As examined the study in [21], among these polynomials, minimum jerk trajectory in terms of linear and angular motion is provided with the 7th-order polynomial.

The 7th-order polynomial used to describe the joint displacement profile can be written as

$$\theta(t) = a_7(t-t_0)^7 + a_6(t-t_0)^6 + a_5(t-t_0)^5 + a_4(t-t_0)^4 + a_3(t-t_0)^3 + a_2(t-t_0)^2 + a_1(t-t_0) + a_0 \qquad (6)$$

The profiles of velocity, acceleration and jerk of the robot joints are obtained by means of the first, second and third derivative of the above equation, respectively, with eight limits specified as $\theta(t_0) = \theta_0$, $\theta(t_f) = \theta_f$, $\dot{\theta}(t_0) = \dot{\theta}_0$, $\dot{\theta}(t_f) = \dot{\theta}_f$, $\ddot{\theta}(t_0) = \ddot{\theta}_0$, $\ddot{\theta}(t_f) = \ddot{\theta}_f$, $\dddot{\theta}(t_0) = \dddot{\theta}_0$, $\dddot{\theta}(t_f) = \dddot{\theta}_f$. The initial time, position, velocity, acceleration and jerk are denoted by t_0, θ_0, $\dot{\theta}_0$, $\ddot{\theta}_0$ and $\dddot{\theta}_0$, while t_f, θ_f, $\dot{\theta}_f$, $\ddot{\theta}_f$ and $\dddot{\theta}_f$ illustrate the final time, position, velocity, acceleration and jerk, respectively. By combining these profiles of the joints with the eight constraints, eight equations can be yielded with eight unknown coefficients.

3.3 Trigonometric splines

Trigonometric spline was firstly proposed by Schoenberg [34]. Later, the trigonometric spline was transformed into the periodic function form by Koch [35], Lyche and Winther [36]. Intermediate points between start and end points in Cartesian space are defined for each joint. All of these points defined in Cartesian space are transformed into joint space by using inverse kinematics. The points transformed into joint space are expressed as joint angles. Each joint has $n+1$ joint angles, and a trigonometric spline consisting of n segments is formed by means of the trigonometric polynomial given in Eq. 7 between two consecutive sets of angles. This method is applied to all joints.

The m^{th} trigonometric spline defined in the range of $[t_i, t_{i+1}]$ is given as the following:

$$x_i(t) = a_{i,0} + \sum_{k=1}^{m-1} \left(a_{i,k}\cos kt + b_{i,k}\sin kt\right) + a_{i,m}\cos m(t-\gamma_i) \quad i = 1, 2, \ldots, n+1 \qquad (7)$$

In the function, the uniqueness of the solution is provided by $\gamma_i = \sum_{j=0}^{2m-1} \frac{\tau_{ij}}{2m}$ defined in the range of $[t_i, t_{i+1}]$ and τ_{ij} are the values of t where the related spline segment

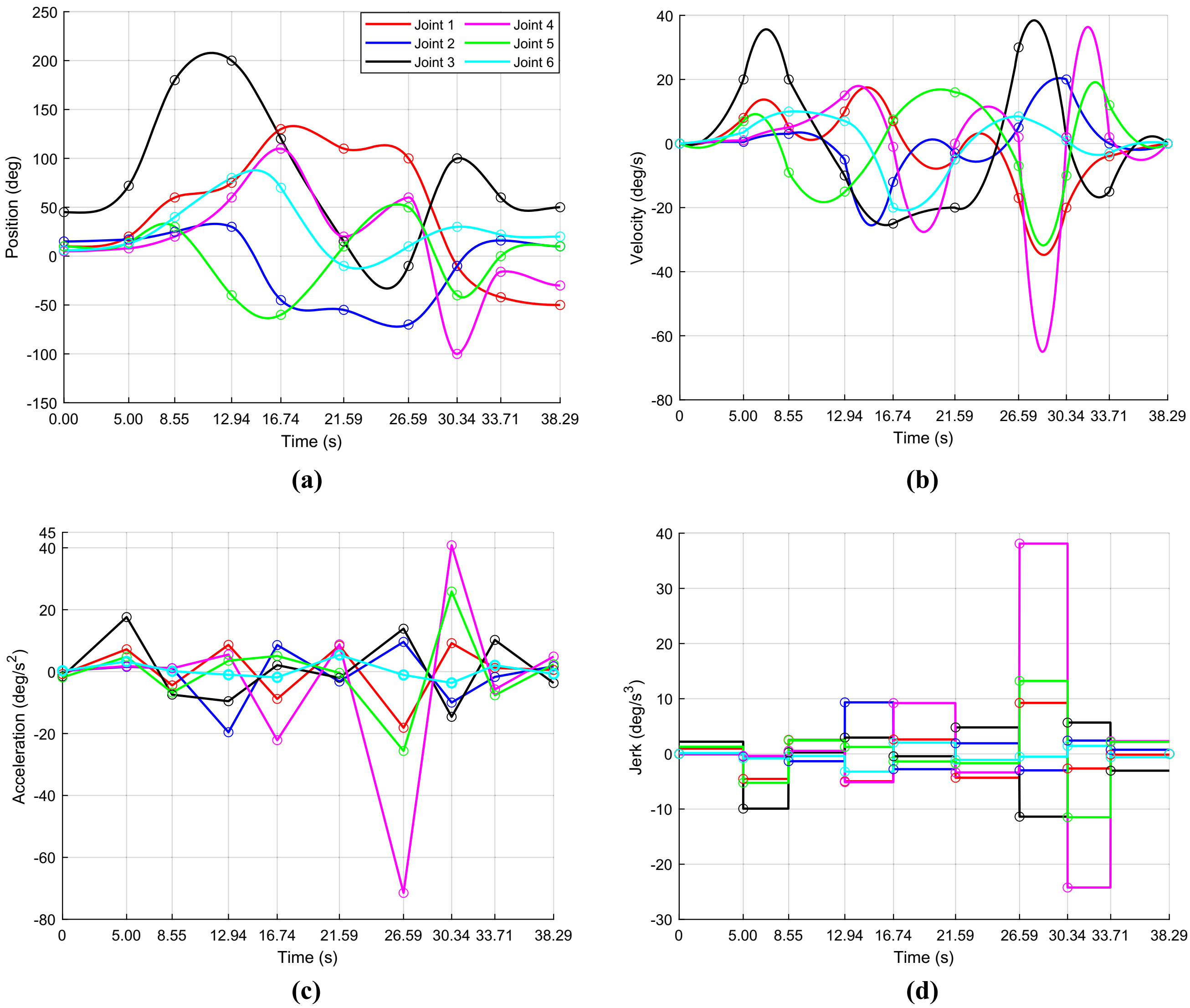

Fig. 5 Resulting joint positions **a**, velocities **b**, accelerations **c** and jerks **d** based on cubic spline for time optimal trajectory planning

Table 4 Time intervals and total execution time of the time optimal trajectory planning approach

Time interval (The interpolation methods)	Interval no									Total execution time
	h_1	h_2	h_3	h_4	h_5	h_6	h_7	h_8	h_9	t_f
Cubic spline (s)	5.00	3.55	4.39	3.80	4.85	5.00	3.74	3.36	4.58	38.29
Trigonometric spline (s)	5.00	4.59	5.00	5.00	3.71	3.84	3.59	4.53	4.86	40.15
Cubic spline with 7th-order polynomial (s)	4.94	3.09	3.49	3.91	4.64	4.99	3.89	4.32	4.99	38.61

has a constraint applied. The continuity of position, velocity, acceleration and jerk can be provided by a fourth-order trigonometric polynomial. For each segment, *2 m* coefficients can be obtained due to the expression γ_i. Accordingly, the 4th-order trigonometric spline consists of 8 coefficients for each segment. As a result, the eight coefficients are determined for each segment in the time interval assumed that $t_i = 0$ and $t_{i+1} = \pi/4$. In this case, γ_i is set to $\pi/8$ and also the time values are assumed to be equal for each other that $\tau_{i0} = \tau_{i1} = \tau_{i2} = \tau_{i3} = 0$ and $\tau_{i4} = \tau_{i5} = \tau_{i6} = \tau_{i7} =$

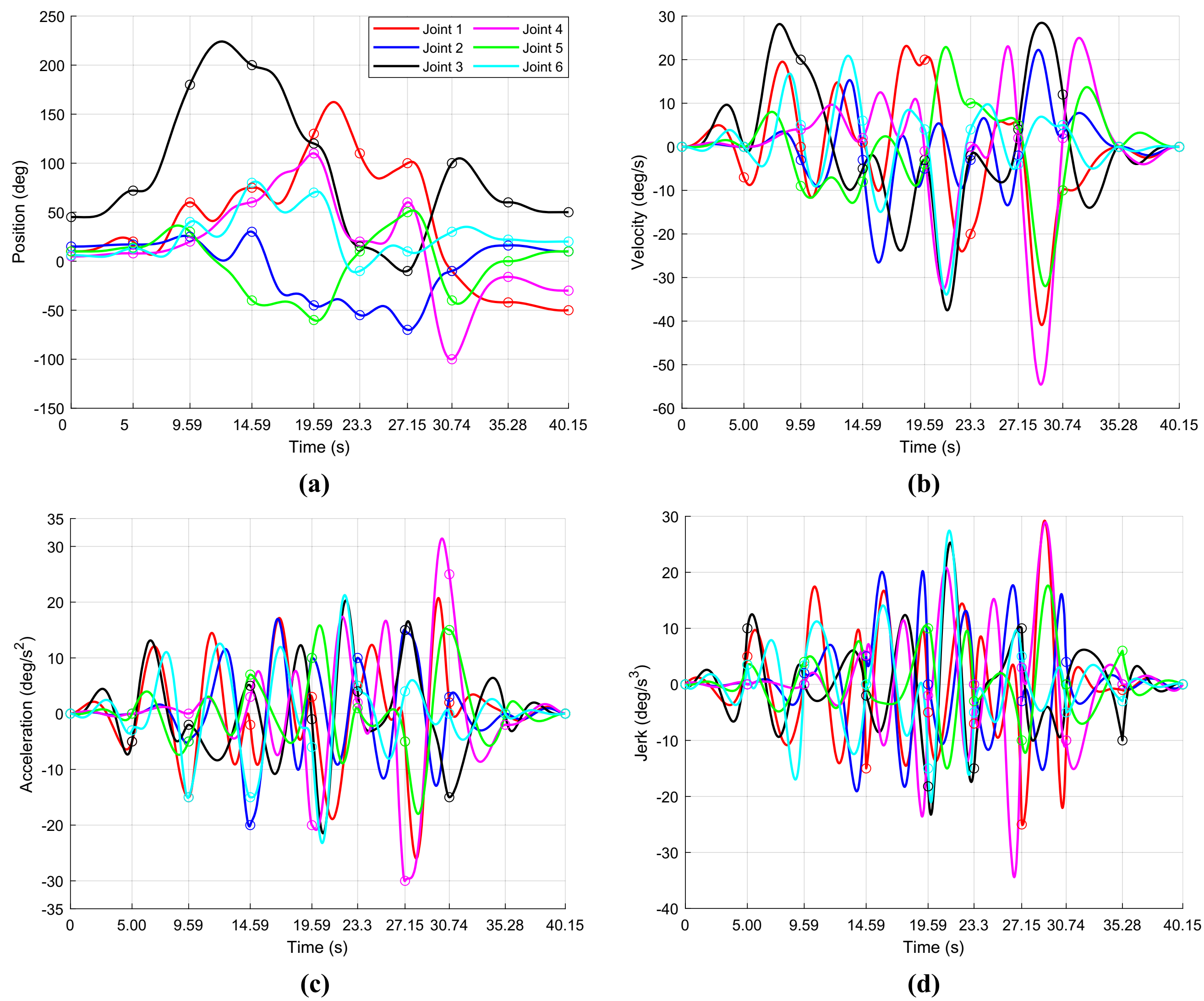

Fig. 6 Resulting joint positions **a**, velocities **b**, accelerations **c** and jerks **d** based on trigonometric spline for time optimal trajectory planning

$\pi/4$. According to these values, the following equation can be obtained as:

$$x_i(t) = a_{i,0} + \sum_{k=1}^{3} \left(a_{i,k} \cos kt + b_{i,k} \sin kt \right) + a_{i,m} \cos mt \tag{8}$$

4 The proposed optimization algorithms

In this section, mathematical definitions of optimization algorithms for optimal trajectory planning are provided.

4.1 PSO

PSO is a population-based optimization technique developed by Eberhart and Kennedy in 1995 [37] inspired by the behaviour of bird swarm. This technique, which is based on a robust stochastic population, has been used in many engineering applications successfully for solving numerical optimization problems [25].

In the PSO algorithm, each individual is called a particle, swarm comprises of these particles. The number of particles is determined according to the size of the problem to be solved. Each particle in the population has a position vector that represents a potential solution to the problem. The particles are initialized at random positions p_0 and velocities v_0 throughout the search space. The velocity vector calculated during each iteration of the algorithm is used to update the position of each particle. The velocity of each particle in the

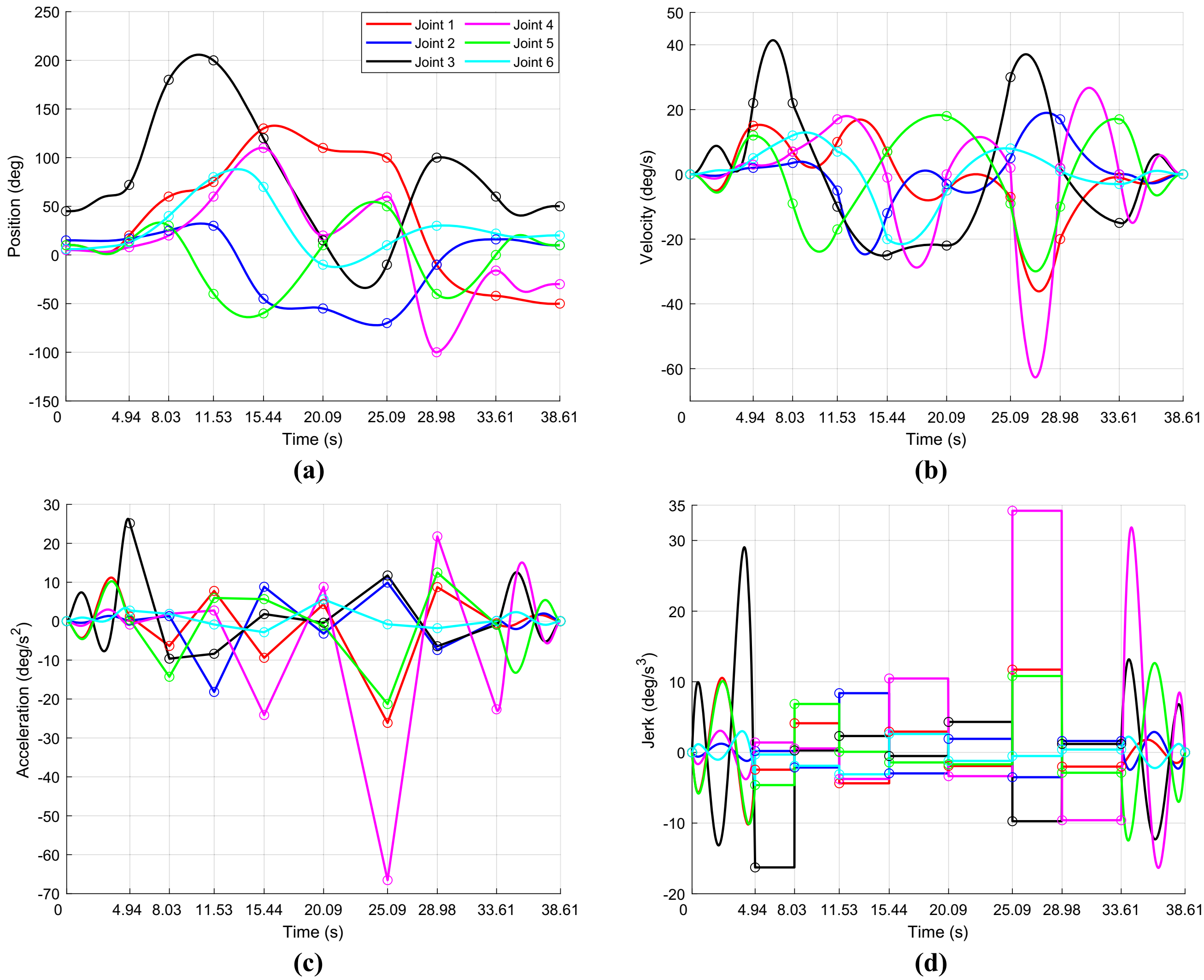

Fig. 7 Resulting joint positions **a**, velocities **b**, accelerations **c** and jerks **d** based on the combination of cubic spline and 7th-order polynomial for time optimal trajectory planning

swarm is influenced by the particle's own experience as well as the experience of its neighbours. The fitness value of the i^{th} particle, calculated in the k iteration, is determined as the individual best $(pbest)$, and the best fitness among all particles is called $gbest$.The previous position p_k^i and the updated velocity vector v_{k+1}^i are used, as given in Eq. 9, to update the position vector p_{k+1}^i as given in Eq. 10 of each particle.

$$v_{k+1}^i = \xi\left(\omega_k v_k^i + \eta_1 r_1\left[p_k^p - p_k^i\right] + \eta_2 r_2\left[p_k^g - p_k^i\right]\right) \quad (9)$$

$$p_{k+1}^i = p_k^i + v_{k+1}^i \quad (10)$$

where subscript k represents a unit step increment. ξ is constriction factor,ω_k is inertia weighting factor which controls the searching abilities of the swarm, η_1 and η_2 are learning factor how much confidence the current particle has in itself and how much confidence it has in the swarm, respectively. r_1 and r_2 are terms of the uniformly distributed value between 0 and 1. v_{k+1}^i,p_{k+1}^i, p_k^p and p_k^g express the particle velocity, particle position, personal best position and global best position, respectively.

4.2 CS

Yang and Deb in 2009 [38] improved a new population-based metaheuristic optimization technique known as cuckoo search algorithm which is inspired by the brood parasitic breeding strategy of certain species of cuckoos by laying their eggs in the nests of other host birds.

In the CS optimization algorithm, each nest is considered a suitable solution candidate and the optimum nest is selected as the best nest. There are three important rules of this newly developed algorithm. These can be stated as follows [38].

Table 5 Maximum joint kinematic values obtained from optimization algorithms for time optimal trajectory planning

Algorithm	Method		Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6	Average
Proposed CS	Trigonometric	V_{max}	23.15	22.24	28.46	25.00	22.91	20.91	23.78
		A_{max}	20.74	17.01	20.31	31.41	15.83	21.26	21.09
		J_{max}	29.22	20.22	25.31	28.92	17.64	27.45	24.79
Proposed CS	Cubic	V_{max}	17.52	20.43	38.45	36.39	19.16	10.03	23.66
		A_{max}	16.53	15.90	22.00	71.47	25.88	8.03	26.64
		J_{max}	9.24	9.34	5.67	38.12	13.23	2.04	12.94
Proposed CS	Cubic + 7th-order polynomial	V_{max}	16.92	18.99	41.43	26.72	18.30	12.93	**22.55**
		A_{max}	11.20	9.91	26.29	21.78	12.49	5.60	**14.54**
		J_{max}	11.72	8.38	29.04	34.20	12.65	3.01	**16.50**
PSO [25]	Cubic + 7th-order polynomial	V_{max}	28.36	39.49	54.53	30.03	39.22	22.13	35.62
		A_{max}	20.29	32.27	29.95	27.78	36.18	16.44	27.15
		J_{max}	17.22	24.15	29.05	24.81	29.04	11.06	22.55
SQP [13]	Cubic	V_{max}	25.89	20.59	45.29	45.84	29.05	22.57	31.54
		A_{max}	21.34	15.23	42.75	32.46	24.29	35.54	28.60
		J_{max}	14.22	30.16	37.50	43.56	46.03	21.52	32.17
QP [17]	Trigonometric	V_{max}	23.93	22.88	45.80	37.33	30.38	19.70	30.00
		A_{max}	21.34	15.23	42.75	32.46	24.29	35.54	28.60
		J_{max}	32.71	20.79	57.46	65.15	28.94	56.95	43.67

The values indicated in bold in the average of the maximum kinematic values column indicate the best values

Table 6 Mean kinematic values obtained from optimization algorithms for time optimal trajectory planning

Algorithm	Method		Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6	Average
Proposed CS	Trigonometric	V_{mean}	8.99	5.28	10.45	8.88	6.99	6.04	7.77
		A_{mean}	6.02	4.68	5.66	6.17	3.98	5.05	5.26
		J_{mean}	6.47	5.14	5.45	5.80	3.71	5.10	5.28
Proposed CS	Cubic	V_{mean}	8.28	5.76	15.55	12.90	10.25	6.17	9.82
		A_{mean}	3.98	3.05	6.24	8.55	5.12	2.27	4.87
		J_{mean}	3.38	2.33	4.19	8.36	4.02	1.11	3.90
Proposed CS	Cubic + 7th-order polynomial	V_{mean}	8.32	5.70	15.65	13.28	11.16	6.20	10.05
		A_{mean}	4.22	2.99	6.89	8.83	5.88	2.42	5.20
		J_{mean}	3.95	2.51	5.75	8.76	4.53	1.40	4.48
PSO [25]	Cubic + 7th-order polynomial	V_{mean}	10.27	8.14	17.59	16.05	12.77	7.87	12.11
		A_{mean}	4.32	4.88	8.63	8.78	7.16	3.80	6.26
		J_{mean}	4.94	4.94	6.79	7.10	6.63	2.29	5.49
SQP [13]	Cubic	V_{mean}	13.03	9.05	23.45	21.25	16.11	9.65	15.43
		A_{mean}	7.25	6.85	12.77	10.00	12.33	6.89	11.02
		J_{mean}	8.33	8.95	14.67	28.76	16.70	9.01	14.40
QP [17]	Trigonometric	V_{mean}	9.84	5.64	13.42	11.89	9.21	6.12	9.35
		A_{mean}	6.29	4.88	9.94	9.57	6.61	6.16	7.24
		J_{mean}	8.61	6.24	16.01	15.38	8.39	7.58	10.37

Table 7 Maximum joint kinematic values obtained from optimization algorithm for jerk optimal trajectory planning

Algorithm	Method		Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6	Average
Proposed CS	Trigonometric	V_{max}	14.05	18.98	24.93	26.74	22.74	8.50	19.33
		A_{max}	7.59	10.12	17.95	23.37	14.57	12.38	14.33
		J_{max}	9.64	11.95	17.27	28.38	12.94	8.40	14.76
Proposed CS	Cubic	V_{max}	13.53	16.10	29.19	24.33	17.04	8.67	18.14
		A_{max}	14.00	12.80	18.40	40.80	14.80	14.00	19.13
		J_{max}	6.00	5.28	4.80	16.32	6.00	4.56	7.16
Proposed CS	Cubic + 7th-order polynomial	V_{max}	12.40	14.39	28.07	24.20	18.00	8.33	**17.56**
		A_{max}	8.72	8.40	10.17	17.76	12.18	5.60	**10.47**
		J_{max}	4.65	3.12	12.69	22.07	13.48	2.54	**9.76**

The values indicated in bold in the average of the maximum kinematic values column indicate the best values

Table 8 Mean kinematic values obtained from optimization algorithm for jerk optimal trajectory planning

Algorithm	Method		Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6	Average
Proposed CS	Trigonometric	V_{max}	5.88	5.30	10.15	8.46	7.85	4.52	7.03
		A_{max}	3.21	3.76	4.82	5.48	4.53	2.62	4.07
		J_{max}	2.84	3.44	4.37	5.27	3.90	2.30	3.68
Proposed CS	Cubic	V_{max}	6.80	4.69	12.13	10.97	8.59	5.07	8.04
		A_{max}	2.52	1.97	4.19	5.78	3.54	1.70	3.28
		J_{max}	1.77	1.34	2.20	4.42	2.10	0.84	2.11
Proposed CS	Cubic + 7th-order polynomial	V_{max}	6.91	4.98	13.17	11.19	9.99	5.55	8.63
		A_{max}	2.66	2.10	4.37	6.30	4.42	1.86	3.62
		J_{max}	1.99	1.44	2.92	5.37	2.88	0.98	2.60

- Each cuckoo randomly selects one of the nests in its environment and lays one egg there.
- The best nests with high-quality eggs will be carried over to the next generation.
- The number of host nests in the selected environment is fixed and the eggs released by the cuckoo may be recognized by the host, probability (p_a) between 0 and 1. In such a case, the host bird may throw the foreign eggs out of the nest or leave the existing nest to establish a new nest.

The cuckoo finds the new nest with a general random walk using the Lévy flight law [39]. The Lévy flight process is basically a random walk which is derived from the Lévy distribution with infinite variance and infinite mean [40]. According to the Lévy flight, let x_k^{n-1} be the current solution for k th cuckoo; then new solution x_k^n is presented as given

$$\begin{cases} x_k^n = x_k^{n-1} + \propto \otimes levy(\lambda), & k = 1, 2, \ldots, n \\ levy(\lambda) = t^{-\lambda}, & (1 < \lambda \leq 3) \end{cases} \tag{11}$$

where $\alpha > 0$ is the step size for the scale of the problem; t presents the current iteration. The Lévy flight, one of the most outstanding features of cuckoo search, produces a new candidate solution by a random walk. The Lévy flight represented by $levy(\lambda)$ is the main parameter of the CS algorithm and is used for both local search and global search.

Summarizing the CS algorithm, the initial population is generated using random numbers. The error involved in each population is then specified. Some quality solutions obtained at each iteration are stored for further processing. Then, by using Lévy's flight, the rest is updated.

4.3 SQP

SQP is one of the most effective methods for nonlinearly constrained optimization problems. This technique is convenient for small and large problems, and it is also used to solve the problems with substantial nonlinearities. The form of the SQP methodology for nonlinear optimization problems (NPL) is given below:

$$\min f(x) \quad x = (x_1, x_2, \ldots, x_n) \in \mathbb{R}^n \tag{12}$$

$$\text{subjectto} \begin{cases} h(x) = 0 \\ g(x) \leq 0 \end{cases} \tag{13}$$

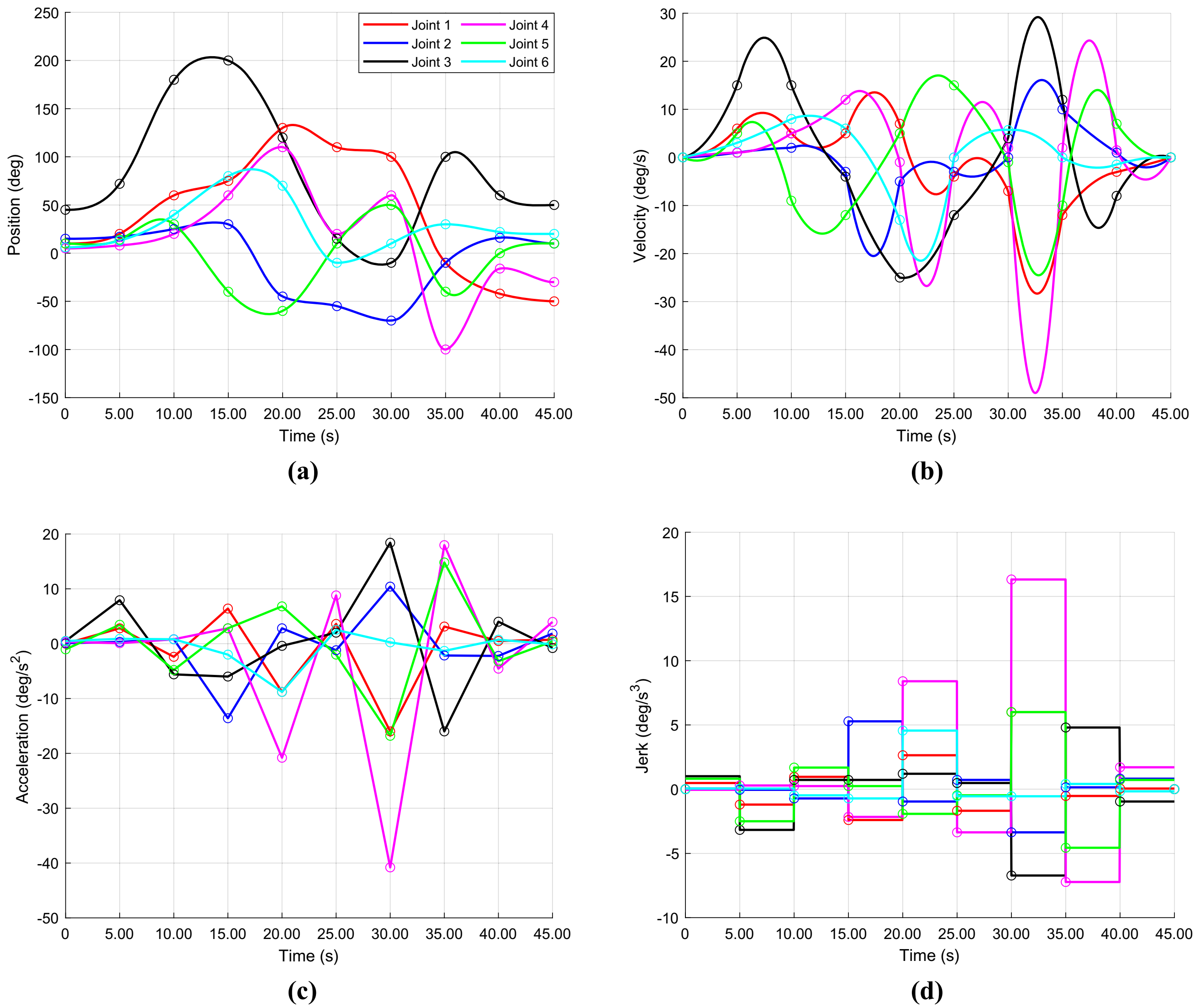

Fig. 8 Resulting joint positions **a**, velocities **b**, accelerations **c** and jerks **d** based on cubic spline for jerk optimal trajectory planning

where $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is the objective functional,$h : \mathbb{R}^n \rightarrow \mathbb{R}^m$ is the equality constrains and $g : \mathbb{R}^n \rightarrow \mathbb{R}^p$ is the inequality constraints. The NLP shown above in Eqs. 12–13 comprises as special cases linear and quadratic programming problems when f is linear or quadratic and the constraint functions h and g are affine [41].

The proposed study uses the implementation of SQP of MATLAB's Optimization Toolbox to solve the limited nonlinear trajectory planning optimization outlined. The process steps of SQP firstly obtain the result of the quadratic subproblem, and secondly, the solution of the quadratic subproblem is utilized to compose a new group of optimization variables for the next iteration of the original problem [12].

5 Implementation of methodology

In this section, optimal trajectory planning for a PUMA 560 structured 6-DOF industrial robot manipulator given in Fig. 2 is performed by using the aforementioned trajectory planning methods through the implementation of the PSO, CS and SQP optimization algorithms. The optimization algorithms and the planning methods have been simulated in MATLAB R2018b executing on an Intel Core i5 laptop with 8250U at 3.40 GHz.

The flow chart of the optimal trajectory planning is shown in Fig. 3. In this chart, the optimal trajectory planning consists of three basic sections: the input section for robot where initial and final configuration are obtained using inverse kinematics, optimization section where the optimization algorithms and spline method are applied using the

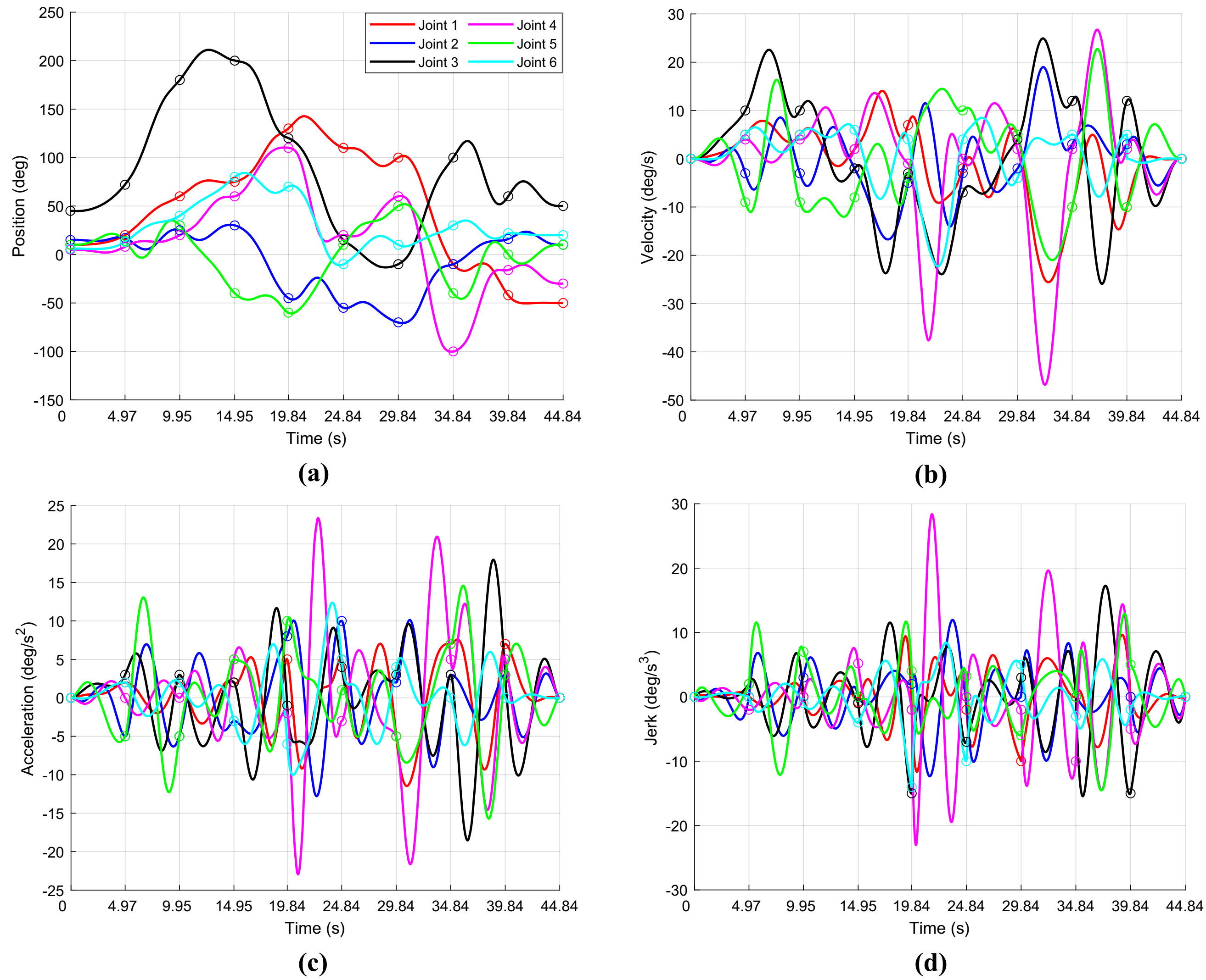

Fig. 9 Resulting joint positions **a**, velocities **b**, accelerations **c** and jerks **d** based on trigonometric spline for jerk optimal trajectory planning

objective functions, and the last section where the position, velocity, acceleration and jerk profiles are calculated for each joint.

Firstly, the Cartesian trajectory of the end-effector is designed for having 8 via-points and 7 segments including two virtual points between the first and the last two via-points as shown in Fig. 4. The virtual points are introduced here for considering the starting and final condition for acceleration and jerk. Secondly, from these via-points defined in Cartesian space, the joint via-points are calculated by applying inverse kinematics transformation. Table 2 gives the values of the angle for each joint at each via-point. Also, the kinematics constraints of the joints, including the limits of velocity, acceleration and jerk for PUMA 560 structured 6-DOF industrial robot manipulator, are given in Table 3. Thirdly, after defining the constraints, one of the optimization algorithms such as PSO, CS and SQP introduced here is selected for solving the robot optimum trajectory planning problem. The proposed CS optimization algorithm can contribute to superior performance for the robot trajectory planning optimization. Then, the appropriate parameters related to the algorithms are assigned to the selected algorithms for the proper operation. During the optimization, three objective functions given in this work are chosen based on either optimal time, optimal jerk or optimal time-jerk robot trajectory planning. Following the selection of the optimization method and the objective function, the joint trajectory is interpolated by using an appropriate spline such as cubic, trigonometric and high-order introduced here for interconnecting the intermediate points.

In robotic applications, the smoothness of movement during start and finish is important. Due to the nature of the cubic spline, it provides continuous position, velocity and acceleration at the beginning and end points, but does not support

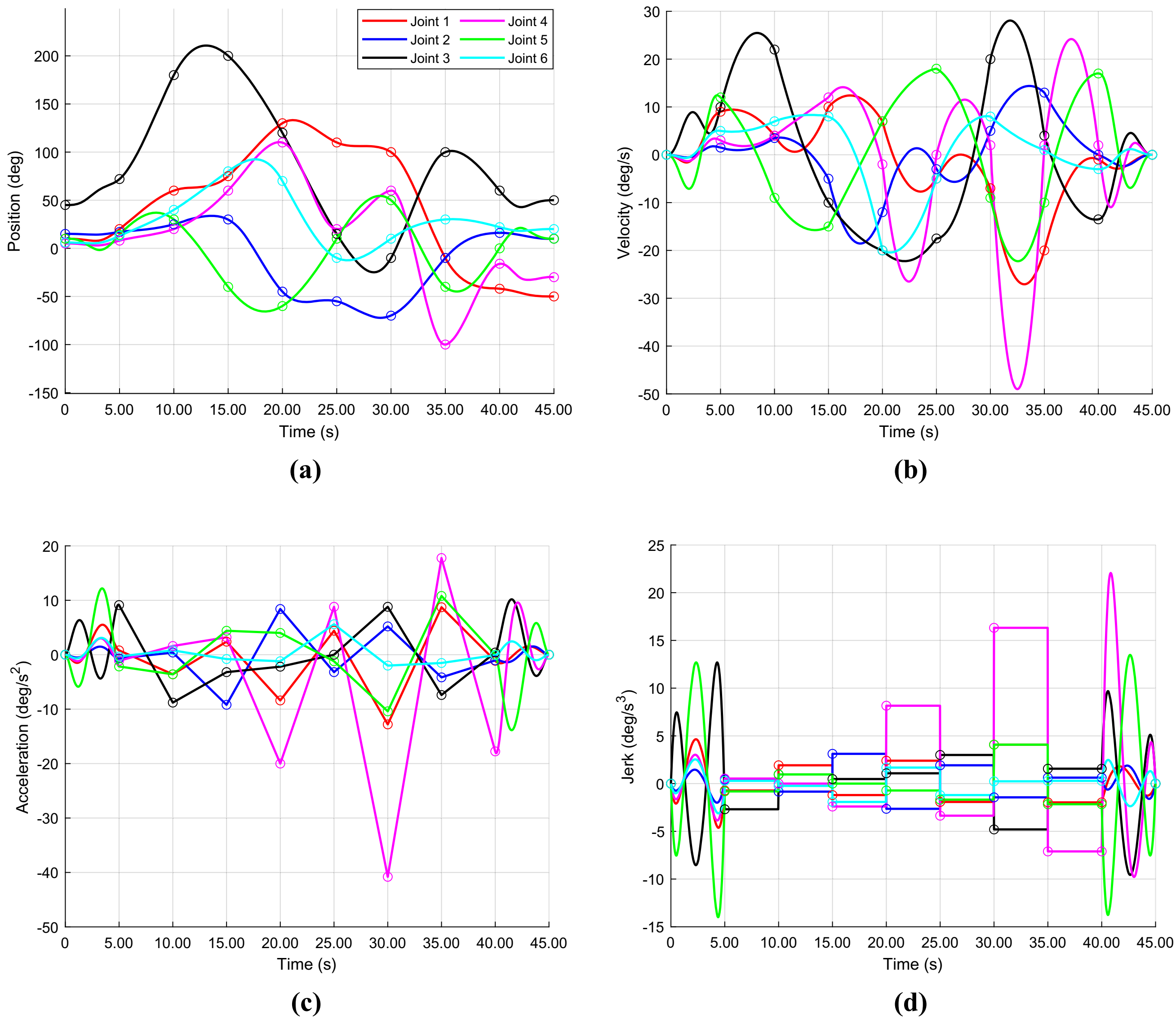

Fig. 10 Resulting joint positions **a**, velocities **b**, accelerations **c** and jerks **d** based on the combination of cubic spline and 7th-order polynomial for jerk optimal trajectory planning

continuous jerk, which may cause vibrations, especially at the beginning and end points. Robot designers have developed some methods to eliminate this undesirable effect since the trajectory with the jerk at the beginning and end points can lead to worse tracking [17]. Accordingly, it is the most crucial that the values of jerk are zero particularly at the initial moment and final moment. One of the developed methods is that cubic spline is combined with the 7th-order polynomial for the trajectory beginning and ending with zero jerk. Thus, cubic spline is replaced with the 7th-order polynomial at the beginning and end of the trajectory. In this way, the undesired impacts at the start and end points can be removed.

As compared to cubic and higher-order polynomials splines, smooth trajectory and continuous jerk can be provided with trigonometric splines at only third derivative of position. However, it has produced a high speed within the segment, but a low speed when approaching the intermediate point, in order to quickly take the robot from the starting point to the target point. This means that the robot speeds up and slows down as much as the number of waypoints. This may shorten the life of the robot's actuator.

Consequently, according to the selection of interpolation spline technique and objective function, the profiles of the position, velocity, acceleration and jerk for each joint are obtained with the chosen optimization algorithm between the initial and intermediate positions and also intermediate and final positions defined in the robot trajectory.

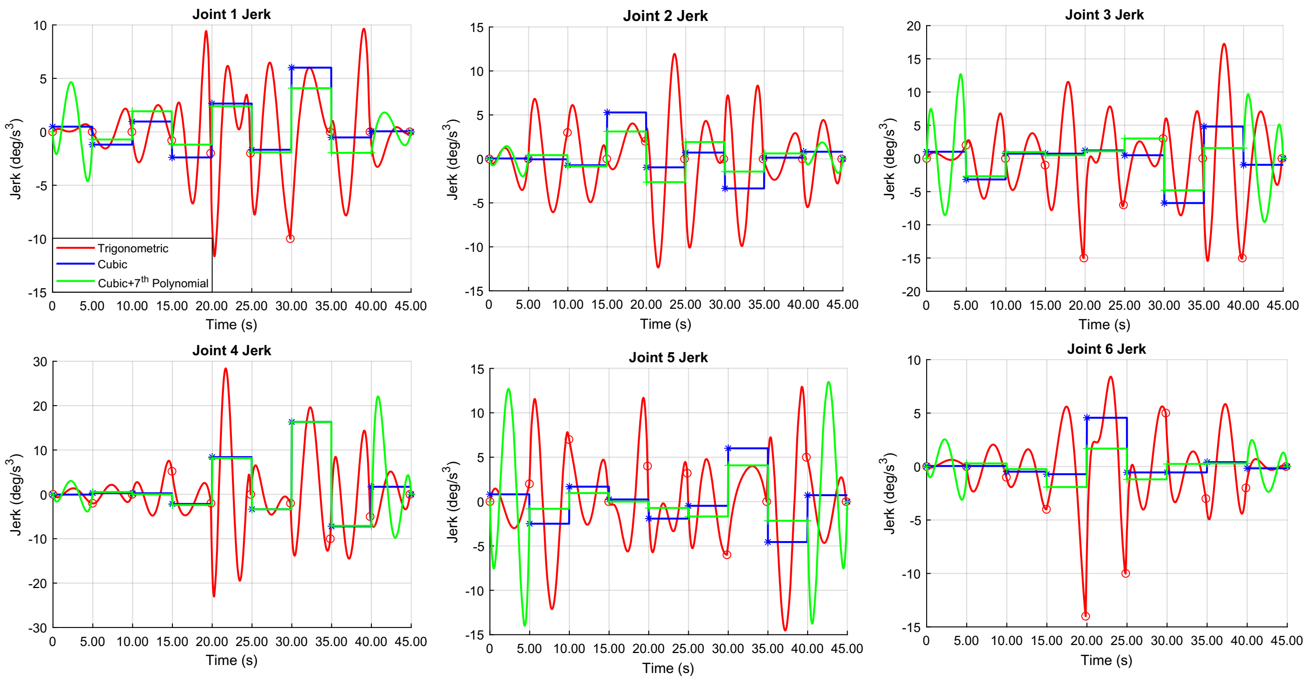

Fig. 11 Resulting joint jerks based on the interpolation methods for jerk optimal trajectory planning

6 Simulation results and discussions

Having carried out the runs of the proposed CS, PSO and SQP algorithms on solving the 6-DOF robot trajectory optimization problem, the performance comparison is investigated based on time optimal, jerk optimal and time-jerk optimal trajectory planning. The parameters of optimization algorithms are selected as follows: For PSO, population size is chosen as 20 particles. Number of iterations is selected as 150. The inertia weighting factor, constriction factor and learning factor are set to 0.8, 0.729 and 2, respectively. For CS, number of nests is chosen as 20. Number of iterations is selected as 150. Discover rate of foreign eggs is set to 0.25. For SQP, the *fmincon* function of MATLAB™ is used.

The results obtained from the proposed CS algorithm are compared to the proposed optimization techniques QP in [13], SQP in [17] and PSO in [25] based on the used interpolation methods in Sect. 3 and the used objective functions in Sect. 2. Also, the comparison results are comprehensively given in Tables 5, 6, 7, 8, 9, 10 and 11 in terms of the maximum and mean kinematic values and their average values.

To verify the effectiveness of the proposed approach, the simulation tests are implemented according to the time optimal, smoothness optimal and optimal time-jerk motion, respectively, using the aforementioned spline methods. The first example represents time optimization for solving minimum time optimal trajectory planning problem based on the cubic spline, trigonometric spline and the combination of cubic spline with 7th-order polynomial. The lower and upper bounds of time interval (h_i) are set to 3 s and 5 s, respectively. After optimization with CS based on the time optimal by using the interpolation methods, the displacements, velocities, accelerations and jerks of each joint are obtained as in Figs. 5, 6 and 7. Table 4 gives the values of the consecutive time intervals and total execution time optimized with the proposed algorithm for time optimal trajectory planning based on the used interpolation methods.

As can be observed from Figs. 5, 6 and 7, a motion that has the velocity, acceleration and jerk of zero at the start and end points of the trajectory is obtained based on the trigonometric spline and the combination of cubic and 7th-order polynomial. On the other hand, while jerk curves generated by the trigonometric spline are continuous along the whole trajectory for all joints, jerk continuous profiles are realized by using the combination of cubic spline and 7th-order polynomial at only beginning and end of the trajectory.

From Table 5 it is noticeable that the maximum kinematic values of each joint based on the proposed algorithm are almost much lower as compared with those yielded by the algorithms in [13, 17, 25]. Also, the average values of the maximum velocities, accelerations and jerks of all joints are obtained lower compared to those obtained from [13, 17, 25]. As a result, in case of using the proposed algorithm and the combination of cubic spline and 7$^{\text{th}}$-order polynomial, the better results are provided as compared with that of [13, 17, 25].

From Table 6, the mean joint velocities, accelerations and jerks yielded by the proposed algorithm as well as their aver-

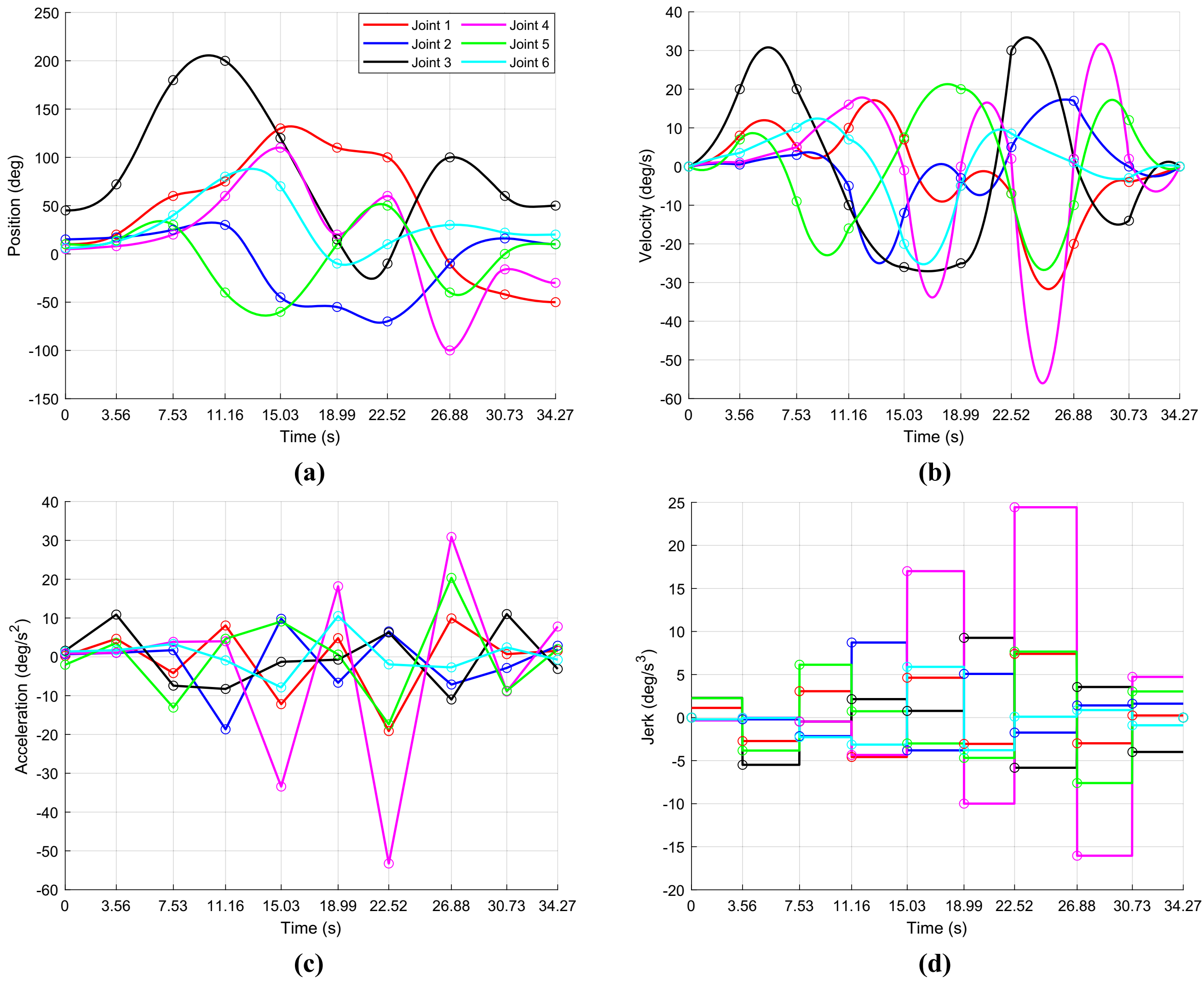

Fig. 12 Resulting joint positions **a**, velocities **b**, accelerations **c** and jerks **d** based on cubic spline for time-jerk optimal trajectory planning

age values are much lower compared with those of [13, 17, 25] based on each interpolation method. CS can perform better compared with those algorithms proposed by [13, 17, 25] in terms of time optimization.

The second example is done to deal with the jerk optimal trajectory planning problem based on the interpolation methods aforementioned above. From the results given in Tables 7 and 8 and shown in Figs. 8, 9 and 10, some remarkable observations can be made.

Firstly, all of the average values of the maximum kinematic variables are obtained lower with the combination of cubic spline and 7$^{\text{th}}$-order polynomial compared to the other methods. The mean value of the joint jerks and their average values according to the trigonometric spline are quite higher than that of the other methods.

Secondly, from Table 8, the better jerk results are provided with the cubic spline compared to the other methods. On the other hand, it is found from Figs. 8, 9 and 10 that the generated accelerations and jerks for each joint turn out to have much smoother in case of using the trigonometric spline. Also, at the initial and final via-points, there are lesser fluctuations in the joint jerks with respect to the trigonometric spline compared to the combination of cubic spline and 7$^{\text{th}}$-order polynomial. However, the high magnitude of jerks generated by the trigonometric spline can cause adverse vibrations.

In order to make the results of jerk more visual for each joint, the resulting jerks are compared as shown in Fig. 11 in terms of the used interpolation methods based on the jerk optimization. From the figure, it is revealed that the trajectory based on the cubic spline has smaller values of jerk for each joint, while the smooth trajectory with jerk continuous is not provided with the cubic spline. On the other hand, it is clear that continuous and smooth path at the initial and final via-points is obtained with the combination of the cubic spline

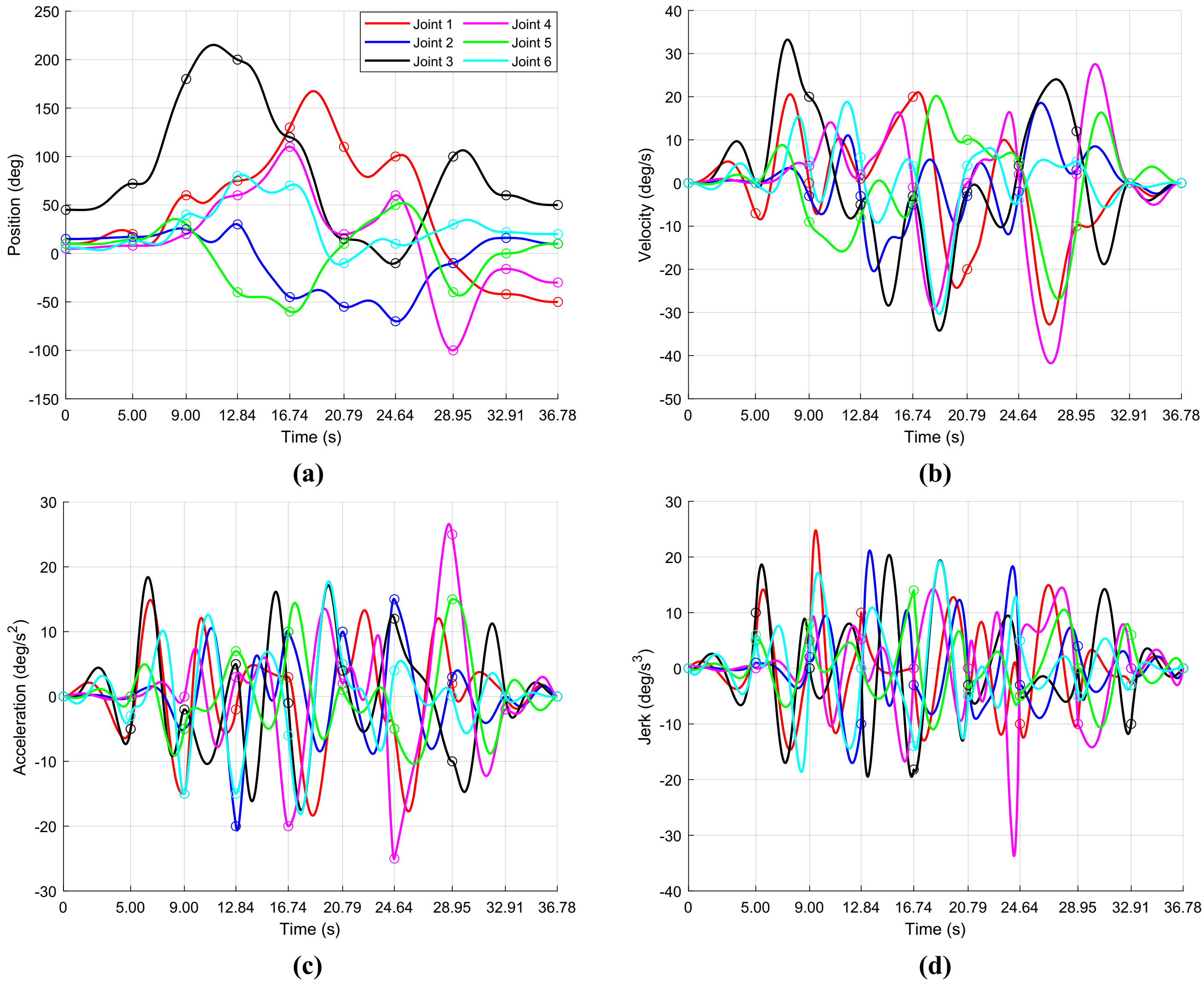

Fig. 13 Resulting joint positions **a**, velocities **b**, accelerations **c** and jerks **d** based on trigonometric spline for time-jerk optimal trajectory planning

and 7th-order polynomial generating almost the same jerk values with the cubic spline.

The optimal time-jerk trajectory planning problem is chosen as another example to show trajectory generations based on the used interpolation methods and the proposed optimization techniques.

From a comparative analysis obtained with Figs. 12, 13 and 14 based on the used interpolation methods, some criticisms can be made. Firstly, it is apparent that the optimal trajectory obtained using the cubic spline method provides the quickest trajectory execution with 34.27 s time spent, while smooth jerk profiles for each joint cannot be obtained from the method. Secondly, it is found that the profiles of the acceleration and jerk of the joint trajectories end up smoother than that of the cubic spline at the initial and final via-points in case of using the combination of the cubic spline and 7th-order polynomial. However, the joint profiles with the acceleration and jerk generated by the method exhibit larger vibrations at the initial and final via-points than that of the trigonometric spline method even though the former method has lower values of the acceleration and jerk. On the other hand, the motion smoothness perspectives are still found in the trigonometric spline trajectory.

As shown in Table 9, when making a comparison in terms of the used methods, the proposed algorithm produces better results than that of the algorithms proposed in [13, 17, 25] in terms of the maximum kinematic values of each joint and their average values. It can also be observed from Table 10 that the mean values of the kinematic variables for each joint are obtained much lower in case of using both the proposed algorithm and the combination of the cubic spline and 7$^{\text{th}}$-order polynomial as compared to the others.

Based on the generated mean jerk values as given in Table 10, the proposed algorithm yields lower for each joint than

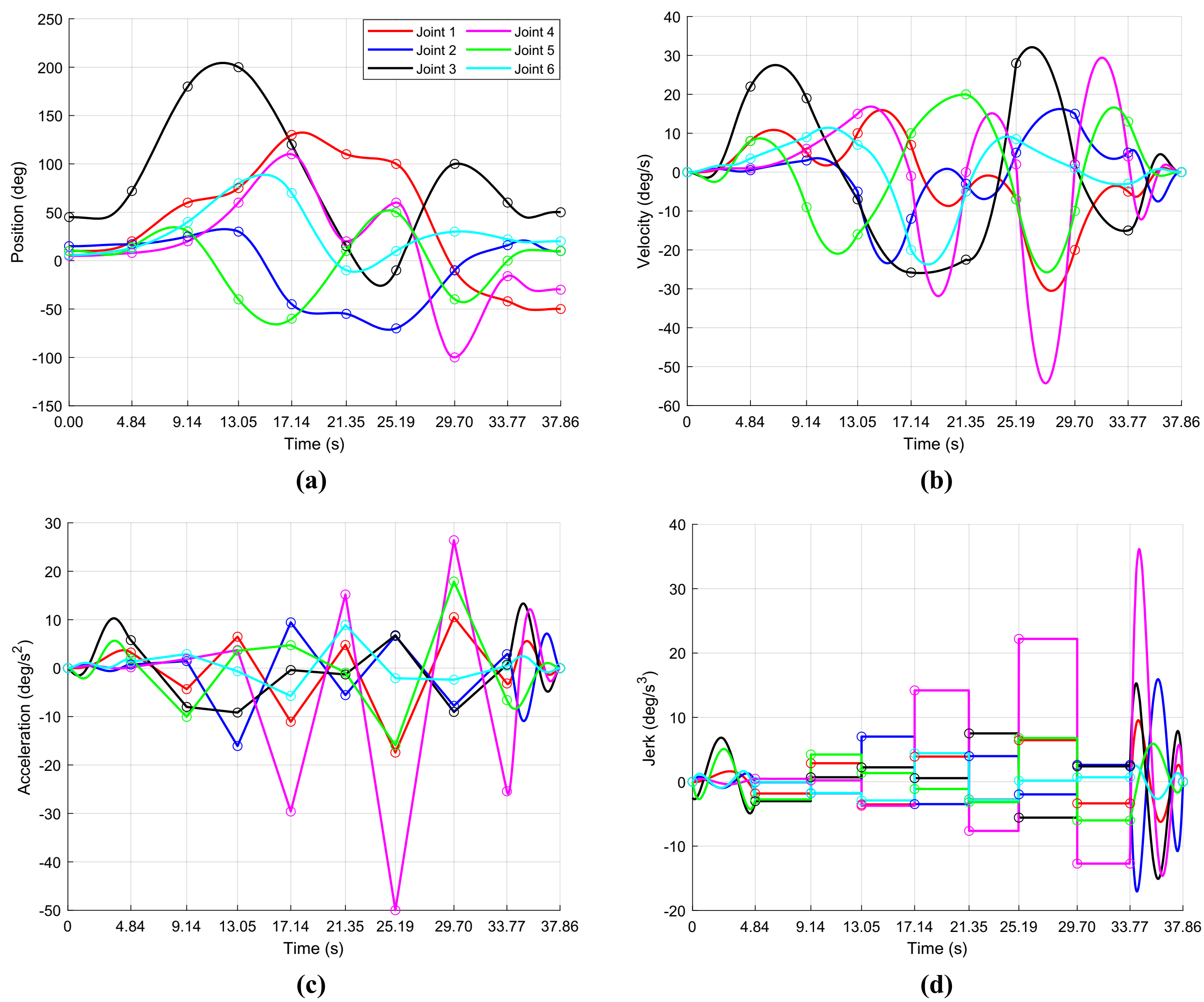

Fig. 14 Resulting joint positions **a**, velocities **b**, accelerations **c** and jerks **d** based on the combination of cubic spline and 7th-order polynomial for time-jerk optimal trajectory planning

the algorithms suggested by [13, 17, 25] in all of the used methods. When viewed from this aspect, the proposed CS algorithm is able to exhibit the better optimized results and provide the comparative solutions than the algorithms proposed in [13, 17, 25].

With the proposed CS optimization algorithm, in addition to previous studies, snap analysis is performed as a derivative of jerk to obtain smoother trajectory planning. Figure 15 shows the results achieved by the trigonometric spline and the combination of cubic spline and 7th-order polynomial at the first segment and the last segment for optimal time, optimal jerk and optimal time-jerk trajectory planning. Also, Table 11 compares the trajectory planning methods with regard to the maximum snap value for each joint.

From the figure and the table, it is apparent that the combination of cubic spline and 7th-order polynomial has a visible impact on decreasing the peak snap values at the starting and ending segments. As a result, the optimized trajectory for each joint generated by the different interpolation methods used above is satisfactorily achieved on the 6-DOF robot manipulator, thereby validating the applicability of the proposed CS algorithm.

7 Conclusion

In this paper, the commonly used interpolation methods for the trajectory generation of the industrial robots have been adopted in order to obtain a time optimal trajectory, a jerk optimal trajectory and a time-jerk optimal trajectory. All the methods have been tested on PUMA robot manipulator by using both the proposed CS algorithm and the algorithms

Table 9 Maximum joint kinematic values obtained from optimization algorithms for time-jerk optimal trajectory planning

Algorithm	Method		Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6	Average
Proposed CS	Trigonometric	V_{max}	21.03	18.58	33.24	27.56	20.18	18.84	23.24
		A_{max}	14.89	15.07	18.42	26.61	15.00	17.75	17.96
		J_{max}	24.79	21.15	20.36	14.50	14.00	19.23	19.01
Proposed CS	Cubic	V_{max}	17.13	17.31	33.38	31.73	21.30	12.41	22.21
		A_{max}	13.16	15.06	31.93	53.25	20.35	15.46	24.87
		J_{max}	7.40	8.72	9.26	24.43	7.66	5.89	10.56
Proposed CS	Cubic + 7th-order polynomial	V_{max}	15.97	16.24	32.12	29.42	20.00	11.41	**20.86**
		A_{max}	10.52	9.45	13.34	26.41	17.88	8.90	**14.42**
		J_{max}	9.56	15.93	15.29	36.15	6.79	4.43	**14.69**
PSO [25]	Cubic + 7th-order polynomial	V_{max}	24.37	21.04	40.81	47.24	24.03	9.03	27.75
		A_{max}	20.90	11.81	34.81	70.78	42.10	6.82	31.20
		J_{max}	16.88	25.19	16.11	45.30	16.25	4.66	20.73
SQP [13]	Cubic	V_{max}	19.94	20.00	35.54	36.30	23.32	15.40	25.08
		A_{max}	16.44	21.57	38.54	60.25	25.76	21.54	30.68
		J_{max}	9.59	13.82	12.31	29.45	13.83	9.80	14.80
QP [17]	Trigonometric	V_{max}	20.90	19.51	39.38	28.82	22.88	23.19	25.78
		A_{max}	14.96	15.09	46.91	27.11	15.01	24.79	23.98
		J_{max}	24.82	15.41	71.35	42.46	17.24	29.62	33.48

The values indicated in bold in the average of the maximum kinematic values column indicate the best values

Table 10 Mean kinematic values obtained from optimization algorithms for time-jerk optimal trajectory planning

Algorithm	Method		Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6	Average
Proposed CS	Trigonometric	V_{mean}	9.06	5.20	11.11	9.61	7.49	5.43	7.98
		A_{mean}	5.54	3.98	6.43	5.88	4.08	4.61	5.09
		J_{mean}	5.18	4.57	6.59	5.48	3.69	5.12	5.11
Proposed CS	Cubic	V_{mean}	8.88	6.27	16.65	14.40	11.30	6.82	10.72
		A_{mean}	3.91	3.32	6.28	9.49	5.75	2.96	5.28
		J_{mean}	3.43	2.78	3.76	8.97	4.38	1.90	4.20
Proposed CS	Cubic + 7th-order polynomial	V_{mean}	8.11	5.99	15.34	13.13	10.59	6.26	9.90
		A_{mean}	3.56	3.25	5.64	8.54	5.05	2.55	4.76
		J_{mean}	3.30	3.38	3.82	8.36	3.48	1.64	4.00
PSO [25]	Cubic + 7th-order polynomial	V_{mean}	8.94	6.52	17.05	14.42	11.61	6.67	10.87
		A_{mean}	5.15	4.17	9.38	11.35	6.22	2.93	6.53
		J_{mean}	4.73	3.37	6.18	13.72	5.80	1.80	5.99
SQP [13]	Cubic	V_{mean}	10.21	7.15	18.85	16.59	12.85	7.72	12.23
		A_{mean}	4.87	4.39	8.17	12.31	7.41	3.95	6.85
		J_{mean}	4.69	4.22	5.37	13.65	7.04	3.38	6.39
QP [17]	Trigonometric	V_{mean}	9.03	5.25	11.60	10.13	7.93	6.11	8.34
		A_{mean}	5.11	4.38	7.79	6.76	4.60	5.70	5.72
		J_{mean}	6.37	4.33	11.60	9.17	4.58	4.63	6.78

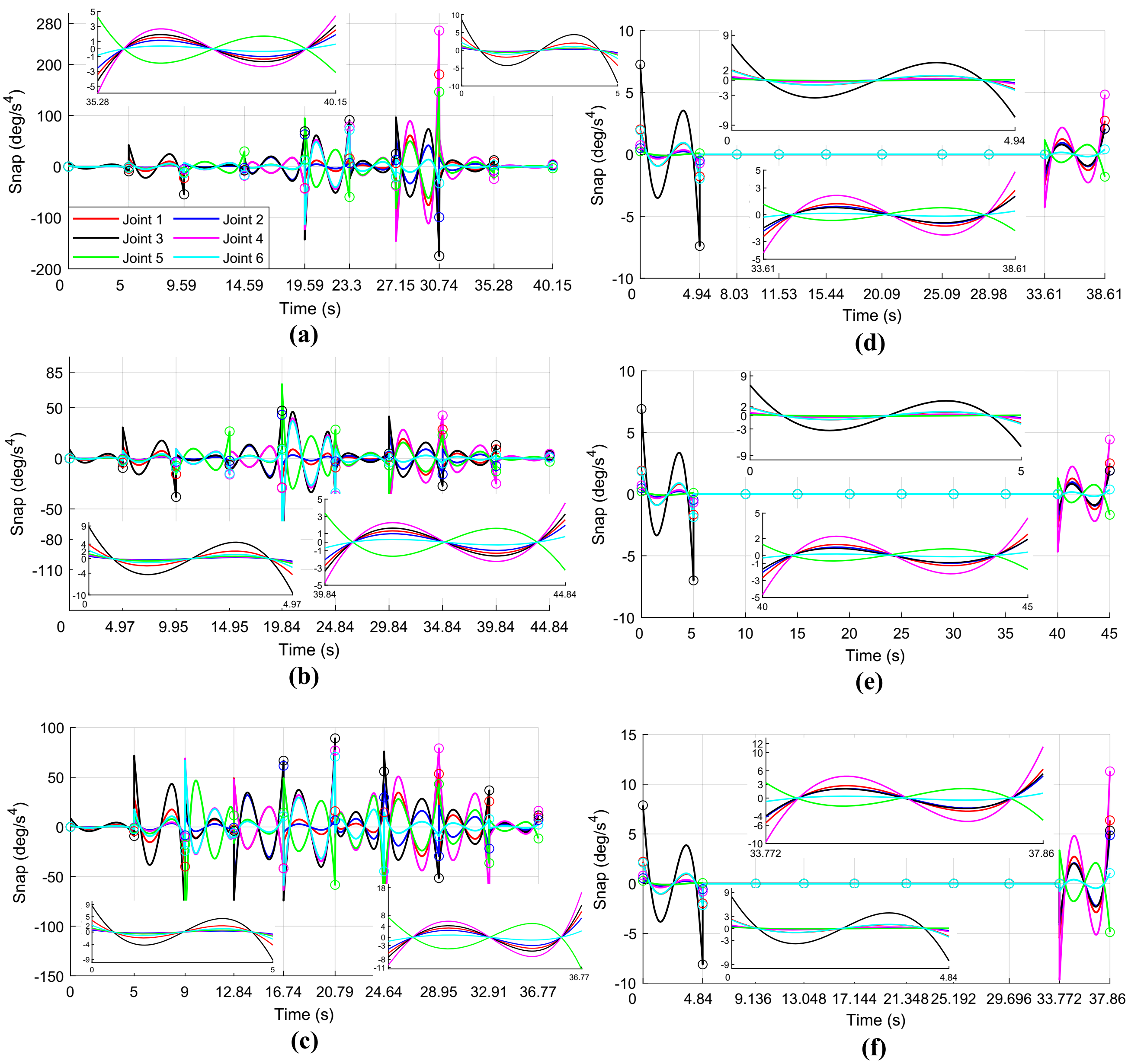

Fig. 15 Joint snap profiles generated by trigonometric spline **a**–**c** and the combination of cubic spline and 7th-order polynomial **d–f** based on time optimal **a**, **d**, jerk optimal **b**, **e** and time-jerk optimal **c**, **f** trajectory planning

suggested by [13, 17, 25] with the aforementioned objective functions. Also, the results from the proposed algorithm have been compared with those by executing the algorithms provided in [13, 17, 25] for each joint. The conclusions can be summarized as follows:

(1) As can be observed from the all results, the kinematic values of each joint generated by the interpolation methods using the proposed algorithm meet the kinematic limits of the joint velocity, acceleration and jerk. This is crucial for dynamic trajectory planning.

(2) At the start and ending points, smooth motion trajectories with the zero jerk for each joint can be obtained in case of using the interpolation methods such as the trigonometric spline and the combination of cubic spline and 7th-order polynomial. With this smooth motion, the trajectory tracking errors are close to zero. Accordingly, it is useful to improve a control for accuracy positioning.

(3) CS optimization algorithm outperforms the algorithms proposed by previous works in terms of obtaining the mean kinematic values for both of time optimal and time-jerk optimal trajectory planning.

Table 11 Maximum snap values for different trajectory planning methods at the beginning and last segments

Performance measure	Method	Segment		Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6
Time	Trigonometric	*First*	$S_{\max}$	3.82	0.64	8.62	0.96	1.28	2.21
	Cubic + 7th-order polynomial			2.01	0.50	7.25	0.74	0.23	1.93
	Trigonometric	*Last*	$S_{\max}$	1.90	3.16	4.41	4.24	0.63	1.90
	Cubic + 7th-order polynomial			2.08	2.05	4.82	1.16	0.40	2.08
Jerk	Trigonometric	*First*	$S_{\max}$	3.91	0.65	8.82	0.98	1.31	2.26
	Cubic + 7th-order polynomial			1.91	0.47	6.90	0.71	0.21	1.84
	Trigonometric	*Last*	$S_{\max}$	2.62	1.96	3.27	4.56	3.37	0.65
	Cubic + 7th-order polynomial			2.49	1.90	1.90	4.42	1.29	0.37
Time-Jerk	Trigonometric	*First*	$S_{\max}$	3.82	0.64	8.62	0.96	1.28	2.21
	Cubic + 7th-order polynomial			2.21	0.54	7.88	0.81	0.28	2.09
	Trigonometric	*Last*	$S_{\max}$	9.34	7.01	11.67	16.30	7.42	2.33
	Cubic + 7th-order polynomial			6.36	4.86	5.34	11.29	3.35	1.06

(4) All of the average values of the maximum kinematic variables for each joint based on the aforementioned methods have been obtained lower as compared with that of previous works in case of time optimal and time-jerk optimal trajectory planning.

(5) In terms of the used interpolation methods, the resulting jerks have been compared based on the jerk optimization. The results demonstrate that the proposed algorithm is good at minimizing the jerk values of the joints.

(6) For optimal time, optimal jerk and optimal time-jerk trajectory planning, the resulting snaps have been compared based on the trigonometric spline and the combination of cubic spline and 7th-order polynomial. The combination of cubic spline and 7th-order polynomial has a visible impact on decreasing the peak snap values at the starting and ending segments.

(7) Having considered the used interpolation methods, the combination of the cubic spline and 7th-order polynomial provides continuous and smooth path at the initial and final via-points.

(8) The simulation results have shown that the used trajectory planning methods with the proposed CS algorithm are very effective and suitable for optimal time, optimal jerk and optimal time-jerk trajectory planning.

References

1. Rubio F, Llopis-Albert C, Valero F, Suñer JL (2016) Industrial robot efficient trajectory generation without collision through the evolution of the optimal trajectory. Robot Auton Syst 86:106–112. https://doi.org/10.1016/j.robot.2016.09.008
2. Bobrow JE (1988) Optimal robot plant planning using the minimum-time criterion. IEEE J Robot Autom 4(4):443–450. https://doi.org/10.1109/56.811
3. Bobrow JE, Dubowsky S, Gibson JS (1985) Time-optimal control of robotic manipulators along specified paths. Int J Robot Res 4(3):3–17. https://doi.org/10.1177/027836498500400301
4. Shin K, McKay N (1985) Minimum-time control of robotic manipulators with geometric path constraints. IEEE Trans Autom Control 30(6):531–541. https://doi.org/10.1109/TAC.1985.1104009
5. Chen Y, Desrochers AA (1989) Structure of minimum-time control law for robotic manipulators with constrained paths. In: 1989 International conference on robotics and automation (pp. 971–976). IEEE. https://doi.org/10.1109/ROBOT.1989.100107.
6. Barre PJ, Bearee R, Borne P (2005) Influence of a jerk controlled movement law on the vibratory behaviour of high-dynamics systems. J Intell Robot Syst 42(3):275–293. https://doi.org/10.1007/s10846-004-4002-7
7. Kyriakopoulos KJ, Saridis GN (1988) Minimum jerk path generation, Robotics and Automation. In: IEEE international conference on robotics and automation (pp. 364–369). IEEE. https://doi.org/10.1109/ROBOT.1988.12075
8. Mattmüller J, Gisler D (2009) Calculating a near time-optimal jerk-constrained trajectory along a specified smooth path. Int J Adv Manuf Technol 45(9):1007–1016. https://doi.org/10.1007/s00170-009-2032-9
9. von Stryk O, Schlemmer M (1994) Optimal Control of the Industrial Robot Manutec r3. In: Bulirsch R, Kraft D (eds), Computational optimal control, ISNM international series of numerical mathematics (Vol. 115, pp. 367–382). Basel, Switzerland: Birkhäuser Basel
10. Hirakawa AR, Kawamura A (1996) Proposal of trajectory generation for redundant manipulators using variational approach applied to minimization of consumed electrical energy. In: Proceedings of 4th IEEE international workshop on advanced motion control—AMC '96—MIE (pp. 687–692). IEEE. https://doi.org/10.1109/AMC.1996.509331
11. Lin CS, Chang PR, Luh JYS (1983) Formulation and optimization of cubic polynomial joint trajectories for industrial robots. IEEE Trans Autom Control 28(12):1066–1073. https://doi.org/10.1109/TAC.1983.1103181
12. Aribowo W, Terashima K (2014) Cubic spline trajectory planning and vibration suppression of semiconductor wafer transfer robot arm. Int J Autom Technol 8(2):265–274. https://doi.org/10.20965/ijat.2014.p0265
13. Gasparetto A, Zanotto V (2008) A technique for time-jerk optimal planning of robot trajectories. Robot Comput-Integr Manuf 24(3):415–426. https://doi.org/10.1016/j.rcim.2007.04.001

14. Gasparetto A, Zanotto V (2010) Optimal trajectory planning for industrial robots. Adv Eng Softw 41(4):548–556. https://doi.org/10.1016/j.advengsoft.2009.11.001
15. Liu H, Lai X, Wu W (2013) Time-optimal and jerk-continuous trajectory planning for robot manipulators with kinematic constraints. Robot Comput-Integr Manuf 29(2):309–317. https://doi.org/10.1016/j.rcim.2012.08.002
16. Chen CS, Chen SK (2019) Synchronization of tool tip trajectory and attitude based on the surface characteristics of workpiece for 6-DOF robot manipulator. Robot Comput-Integr Manuf 59:13–27. https://doi.org/10.1016/j.rcim.2019.01.016
17. Simon D, Isik C (1991) Optimal trigonometric robot trajectories. Robotica 9(4):379–386. https://doi.org/10.1017/S0263574700000552
18. Perumaal SS, Jawahar N (2013) Automated trajectory planner of industrial robot for pick-and-place task. Int J Adv Rob Syst 10(2):1–17. https://doi.org/10.5772/53940
19. Gasparetto A, Zanotto V (2007) A new method for smooth trajectory planning of robot manipulators. Mech Mach Theory 42(4):455–471. https://doi.org/10.1016/j.mechmachtheory.2006.04.002
20. Liu L, Chen C, Zhao X, Li Y (2016) Smooth trajectory planning for a parallel manipulator with joint friction and jerk constraints. Int J Control Autom Syst 14:1022–1036. https://doi.org/10.1007/s12555-014-0495-4
21. Boryga M, Graboś A (2009) Planning of manipulator motion trajectory with higher-degree polynomials use. Mech Mach Theory 44(7):1400–1419. https://doi.org/10.1016/j.mechmachtheory.2008.11.003
22. Machmudah A, Parman S, Zainuddin A, Chacko S (2013) Polynomial joint angle arm robot motion planning in complex geometrical obstacles. Appl Soft Comput 13(2):1099–1109. https://doi.org/10.1016/j.asoc.2012.09.025
23. Huang J, Hu P, Wu K, Zeng M (2018) Optimal time-jerk trajectory planning for industrial robots. Mech Mach Theory 121:530–544. https://doi.org/10.1016/j.mechmachtheory.2017.11.006
24. Wang M, Luo J, Yuan J, Walter U (2018) Coordinated trajectory planning of dual-arm space robot using constrained particle swarm optimization. Acta Astronaut 146:259–272. https://doi.org/10.1016/j.actaastro.2018.03.012
25. Kucuk S (2018) Optimal trajectory generation algorithm for serial and parallel manipulators. Robot Comput-Integr Manuf 48:219–232. https://doi.org/10.1016/j.rcim.2017.04.006
26. Lu S, Zhao J, Jiang L, Liu H (2017) Time-jerk optimal trajectory planning of a 7-DOF redundant robot. Turk J Electr Eng Comput Sci 25:4211–4222. https://doi.org/10.3906/elk-1612-203
27. Rout A, Dileep M, Mohanta GB, Deepak B, Biswal B (2018) Optimal time-jerk trajectory planning of 6 axis welding robot using TLBO method. Procedia Comput Sci 133:537–544. https://doi.org/10.1016/j.procs.2018.07.067
28. Savsani P, Jhala RL, Savsani VJ (2013) Optimized trajectory planning of a robotic arm using teaching learning based optimization (TLBO) and artificial bee colony (ABC) optimization techniques. In: 2013 IEEE ınternational systems conference (SysCon). https://doi.org/10.1109/syscon.2013.6549910
29. Savsani P, Jhala RL, Savsani VJ (2016) Comparative study of different metaheuristics for the trajectory planning of a robotic arm. IEEE Syst J 10(2):697–708. https://doi.org/10.1109/jsyst.2014.2342292
30. Wang M, Luo J, Fang J, Yuan J (2018) Optimal trajectory planning of free-floating space manipulator using differential evolution algorithm. Adv Space Res 61(6):1525–1536. https://doi.org/10.1016/j.asr.2018.01.011
31. Bureerat S, Pholdee N, Radpukdee T, Jaroenapibal P (2019) Self-adaptive MRPBIL-DE for 6D robot multiobjective trajectory planning. Expert Syst Appl 136:133–144. https://doi.org/10.1016/j.eswa.2019.06.033
32. Kolter JZ, Andrew YN (2009. Task-space trajectories via cubic spline optimization. In: IEEE international conference on robotics and automation (pp. 1675–82). IEEE. https://doi.org/10.1109/ROBOT.2009.5152554
33. Kucuk S (2016) Maximal dexterous trajectory generation and cubic spline optimization for fully planar parallel manipulators. Comput Electr Eng 56:634–647. https://doi.org/10.1016/j.compeleceng.2016.07.012
34. Schoenberg IJ (1964) On trigonometric spline interpolation. J Math Mech 13(5):795–825
35. Koch PE (1984) Error bounds for interpolation by fourth order trigonometric splines. In: Singh SP, Burry JWH, Watson B (eds) Approximation theory and spline functions, vol 136. Springer, Dordrecht, pp 349–360
36. Lyche T, Winther R (1979) A stable recurrence relation for trigonometric B-splines. J Approx Theory 25(3):266–279. https://doi.org/10.1016/0021-9045(79)90017-0
37. Kennedy J, Eberhart RC (1995) Particle swarm optimization. In: Proceedings of ICNN'95—international conference on neural networks, pp1942–1948. IEEE. https://doi.org/10.1109/ICNN.1995.488968
38. Yang XS, Deb S (2009) Cuckoo search via Levy flights. In: 2009 World congress on nature & biologically inspired computing (NaBIC), pp 210–214. IEEE. https://doi.org/10.1109/NABIC.2009.5393690
39. Soneji H, Sanghvi RC (2012) Towards the improvement of Cuckoo search algorithm. In: World Congress on Information and Communication Technologies (WICT), 2012, pp 878–883. IEEE
40. Yang XS, Deb S (2010) Engineering optimisation by cuckoo search. Int J Math Modell Numer Optim 1(4):330–343
41. Buskens C, Maurer H (2000) SQP methods for solving optimal control problems with control and state constraints: adjoint variables, sensitivity analysis and real time control. J Comput Appl Math 120(1–2):85–108. https://doi.org/10.1016/S0377-0427(00)00305-8
42. Azarfar A, Azarfar B, Vahedi M (2021) Self-tuning fuzzy task space controller for puma 560 robot. J Electr Eng Technol 16:579–589. https://doi.org/10.1007/s42835-020-00576-6