Automated Trajectory Planner of Industrial Robot for Pick-and-Place Task

Regular Paper

S. Saravana Perumaal1,* and N. Jawahar1

1 Department of Mechanical Engineering, Thiagarajar College of Engineering, Madurai, Tamilnadu, India
* Corresponding author E-mail: ssmech@tce.edu

Received 5 Aug 2012; Accepted 1 Oct 2012

DOI: 10.5772/53940

© 2013 Perumaal and Jawahar; licensee InTech. This is an open access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/3.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Abstract Industrial robots, due to their great speed, precision and cost-effectiveness in repetitive tasks, now tend to be used in place of human workers in automated manufacturing systems. In particular, they perform the pick-and-place operation, a non-value-added activity which at the same time cannot be eliminated. Hence, minimum time is an important consideration for economic reasons in the trajectory planning system of the manipulator. The trajectory should also be smooth to handle parts precisely in applications such as semiconductor manufacturing, processing and handling of chemicals and medicines, and fluid and aerosol deposition. In this paper, an automated trajectory planner is proposed to determine a smooth, minimum-time and collision-free trajectory for the pick-and-place operations of a 6-DOF robotic manipulator in the presence of an obstacle. Subsequently, it also proposes an algorithm for the jerk-bounded Synchronized Trigonometric S-curve Trajectory (STST) and the ‘forbidden-sphere’ technique to avoid the obstacle. The proposed planner is demonstrated with suitable examples and comparisons. The experiments show that the proposed planner is capable of providing a smoother trajectory than the cubic spline based trajectory.

Keywords Trajectory Planning, S-Curve Trajectory, Jerk, Collision Avoidance, Robotic Manipulator

1. Introduction

Industrial robots, due to their great speed, precision and cost-effectiveness in repetitive tasks, now tend to be used instead of manual workers in automated production lines. These powerful machines are hardly autonomous, in the sense that they require preliminary actions such as calibration and trajectory planning in order to achieve defined tasks [1]. The trajectory planning problem is a fundamental one in robotics and defines a temporal motion law with a given geometric path, so that certain requirements set for the trajectory are fulfilled. The aim of trajectory planning is to generate the input for the control system of the manipulator to execute a motion. Figure 1 describes the structure of the automated trajectory planning system [2]. The input data for any trajectory planning algorithm are: geometric path, kinematic and dynamic constraints. The geometric path describes the path to be followed by the end-effector for continuous motion such as welding, painting, etc.; for a point-to-point motion such as a pick-and-place operation, however, the path of the end-effector is immaterial in the absence of obstacles. In the presence of any obstacles, the user defines the way point’s position and the orientation of the end-effector or series of knot vectors to avoid the collision [3,4,5].
The kinematic constraints are the velocity and acceleration limits (and jerk limits, if available) of the actuators as given by the manipulator manufacturer. Most of the manufacturers provide velocity limits for both joint-space motion and for task-space motion (i.e., straight-line path motion limits). The dynamic constraints specify torque limits for each joint. The high-level trajectory planner computes a reference trajectory for each task by checking the way points and assigns specific velocities to these points. The output is the trajectory of the joints (or of the end-effector), expressed as a time history of position, velocity and acceleration. Then, the completed portions of the trajectory are sent to the controller through a set-point generator. The proposed trajectory planner is able to generate a collision-free trajectory without any user-defined intermediate way points or knot points. This paper focuses on an automated trajectory planner for PUMA 560, the most-used industrial robot in experiments with 6-DOF, to carry out the pick-and-place operation.

Almost every technique found in the scientific literature on the trajectory planning problem is based on the optimization of some parameter or objective function. The general optimality criteria are: minimum execution time, minimum energy (or actuator effort) and minimum jerk. Some hybrid optimality criteria such as time-energy optimal trajectory planning and jerk time optimal trajectory planning have also been proposed. Minimum-time algorithms are tightly linked to the need to increase productivity in the industrial sector [4]. Pick-and-place is a non-value-added activity which at the same time cannot be eliminated. Hence, it must show minimal lead time and give maximum production. Therefore, the minimum time is considered to be important for economic reasons. However, in order to implement the trajectory in real time, the method must have a low computational complexity.

Jerk directly influences the motion of the robot in many aspects and a minimal jerk ensures smooth motion. The benefits of jerk minimization are: reduction in trajectory tracking errors, less stress induced in actuators and the structure of the manipulator, limited excitation of resonance frequencies of the robot, and the bringing about of natural and coordinated robot motion. It has been established that limiting jerk in manipulator trajectories reduces manipulator wear and improves tracking accuracy and speed [2,6-9]. Direct tracking of a time-optimal trajectory leads to joint vibrations and overshoot of the nominal torque limits [10]. Jerk limitation (and, proportionally, torque rate) results in smoothed actuator loads [6]. This effectively reduces the excitation of the resonant frequencies of the manipulator and consequently reduces actuator wear [9]. Low jerk trajectories can be tracked faster and more accurately [10]. Macfarlane and Croft [2] also inferred that a sine wave template for accelerations of smooth trajectory ensures that the quintics do not oscillate. Jerk minimization is considered as equal as or sometimes more than the minimum time requirement. Smooth motion without any unwanted variations in the manipulator’s movement (i.e., with minimum jerk) is required for the applications such as semiconductor manufacturing, processing and handling of chemicals and medicines, fluid and aerosol deposition. On the above concerns, the generation of a smooth and minimum-time trajectory for a 6-DOF robot manipulator suitable for precise handling applications is the objective of the proposed trajectory planning system.

Various approaches have been attempted to obtain smooth and/or time-optimal trajectories. Initial work in robot trajectory planning focused mainly on time optimality [11]. Geering et al. [12] showed that the unconstrained time-optimal trajectory was either bang-bang or bang-singular-bang. Bobrow et al. [13] and Shin and Mckay [14] showed that the optimal parameterized path to Path-Constrained Time-Optimal Motion (PCTOM) was also bang-singular-bang. It is further proved that all optimal paths must be saturated in at least one actuator at all times [15]. Kieffer et al. [16] developed a torque-based controller attempting to track the PCTOM while limiting actuator chatter. Constantinescu and Croft [10] presented a method for calculating smooth and time-optimal path-constrained trajectories. Smooth and time-optimal motion was obtained by optimizing a base trajectory subject to actuator torque and torque rate limits. Gasparetto and Zanotto [17] suggested an indirect method to get the same results, namely to set an upper limit on the jerk, defined as the time derivative of the acceleration of the manipulator joints. Cubic splines are often used in smooth trajectory generation [15,18-27]. Since cubic splines can ensure the continuity of velocity and acceleration, they are widely used for interpolation and to prevent large oscillations of the trajectory, which can result in higher-order polynomials [18]. Cao et al. [25]...
optimized a piecewise cubic polynomial spline to obtain a smooth and time-optimal constrained motion. The time-optimal motion has been approximated using a Linear Segment with Parabolic Blends (LSPB) type trajectory with carefully selected switching points [10,13]. Chand and Doty [19] used polynomial splines to interpolate between joint target points. Higher order polynomials are generally not used due to their tendency to oscillate and, therefore, to generate retrograde motion [8,19]. Trigonometric Splines [26,28], Quintic Splines [2,4] and B-splines [29,30] have also been used to generate smooth trajectories. Various splines including the Cubic Spline, Trigonometric Spline, Polynomial Spline, Quintic Spline, B-spline are proposed for jerk-limited motion profiles. These profiles are generated with the help of knot vectors which are specified by the user for interpolation.

S-curve motion is another approach for jerk-limited motion [27-29]. The generation of S-curve trajectory does not require any knot points for interpolation. Tsay and Lin [27] developed an input strategy with fast acceleration and slow deceleration to minimize the residual response based on 3rd order polynomial S-curve motion profiles. Nguyen et al. [28] presented the generalized polynomial S-curve model together with the proposal of a general algorithm to design S-curve motion trajectories. They also made a comparison of the actual velocity of the 3rd, 4th and 5th order S-curve motions performed by the linear motor and the position errors of those S-curve motions. Their results indicate that the higher the order of the model, the better the performance of the linear motor. Experiments were also carried out on a linear motor system to study its performance with the motion profiles planned by the trigonometric model, which replaces the rectangular pulse in the jerk of the 3rd order S-curve model by a trigonometric function. Li et al. [29] proposed acceleration-bounded S-curve motion for jerk-limited operation and implemented it on a Digital Signal Processing (DSP) based motion controller for a voice coil motor. The review reveals that the trigonometric S-curve trajectory is:

- as simple as the 3rd order polynomial model and
- capable of producing a smooth trajectory as good as 5th order polynomial S-curves and quintic splines.

Taking the above points into consideration, this paper proposes a new jerk-bounded S-curve trajectory with a sine wave form of jerk profile and the motion profiles: displacement, velocity and acceleration are generated within the maximum jerk value so as to obtain the desired smoother movement without any abrupt changes. This paper also proposes a trajectory planner for a smooth and minimum-time collision-free trajectory under kinematic and obstacle constraints for the pick-and-place operations of a 6-DOF robotic manipulator. The rest of the paper is organized as follows: section two describes the problem description, section three outlines the proposed planner with the illustrative numerical examples, section four discusses the results and section five presents the conclusion.

2. Problem Description

2.1 Operational features

Figure 2 shows the operation sequence of a 6-DOF robotic manipulator between two destinations. The robot’s joints follow a Synchronized Trigonometric S-curve Trajectory (STST) between initial destination ‘P’ and final destination ‘Q’, which are described with Cartesian description. The STST has three phases: Acceleration, Constant Velocity and Deceleration.

![Figure 2. Operation sequence of a 6-DOF robotic manipulator between two destinations](image)

Let $t_1$, $t_2$, $t_3$ and $t_4$ be the absolute time instants when the robot is at the initial destination ‘P’, end of the acceleration phase ‘L’, start of the deceleration phase ‘S’ and final destination ‘Q’, respectively. Figure 3 shows the motion of each joint ‘$n$’ during the three phases, and the characteristic features are described in Table 1.

![Figure 3. Schematic Time History Trigonometric S-curve trajectory of a joint: a) Displacement, b) Velocity, c) Acceleration and d) Jerk](image)
2.2 Objective criterion

Generation of smooth trajectory for handling parts without vibration coupled with minimum time is the major objective under consideration in this paper. The proposed STST ensures jerk within the user-defined specifications. The minimum time depends on the sum of the time intervals during the three phases of the trajectory between the destinations ‘P’ and ‘Q’. Equation 1 provides the time taken by each joint to execute all the three phases.

\[
T_{PQ}^{3D} = \{T_{PQ}^L, T_{PQ}^V, T_{PQ}^S\} 
\]

(1)

However, it is necessary to start and end instants of the three different phases of all the joints should occur at the same time to ensure the robot’s end-effector follows an STST between any-destinations point ‘P’ and final destination ‘Q’. This is achieved by synchronizing all the joint motions with the joint that takes more time. Therefore, the execution time of the robot for the specified operation between the destinations ‘P’ and ‘Q’ is given in (2).

\[
T_{PQ}^{sync} = \max_{\forall n} \{T_{PQ}^L, T_{PQ}^V, T_{PQ}^S\} \quad (2)
\]

Hence, the objective function to the problem is the minimization of \( T_{PQ}^{sync} \) expressed in equation (2).

2.3 Constraints

The robot kinematic parameters control/limit the minimum execution time and become the constraints on the problem. Each joint of the robot has its position within its specified range \( [\theta_{\text{min}}, \theta_{\text{max}}] \) and the kinematic constraints are: maximum velocity, \( v_{\text{max}} \), maximum acceleration, \( a_{\text{max}} \) and maximum jerk, \( j_{\text{max}} \) values of all (six) joints. Another important consideration in the trajectory planning problem is the presence of obstacles. This work considers the fixed obstacle (OBS) as a cube where side a and the reference base corner \( P_r \) are known. \( P_1, P_2 \) and \( P_3 \) are the extreme points on the obstacle along the X, Y and Z directions in relation to \( P_r \) as shown in Figure 4. With the help of these points, a set of parametrically discretized points of the obstacle is defined as \( R(x_{ob}, y_{ob}, z_{ob}) \), expressed in:

\[
R(x_{ob}, y_{ob}, z_{ob}) = P_1 + u_1P_1 + u_2P_2 + u_3P_3 \quad (3)
\]

where \( 0 \leq u_1, u_2, u_3 \leq 1 \)

![Figure 4. Obstacle with its discretized set of points](image-url)
of the robot. \( P_B, P_{11}, P_{12}, P_{13} \) and \( P_{14} \) are the Cartesian points of the robot base, joints 1, 2, 3 and 4, respectively. Since the origins of the other two joints 5 and 6 are located at the origin of joint 4, the robot is considered to have four links. The links are created parametrically as lines connecting the joints of the robot, i.e., line \( L_1 \) connecting \( P_B \) and \( P_{11} \) is link1 and other links \( L_2, L_3 \) and \( L_4 \) are similarly defined between \( P_{11} \) and \( P_{12}, P_{12} \) and \( P_{13} \), and \( P_{13} \) and \( P_{14} \), respectively. The locations of robot joints and approximated links are specified in Figure 5, which depicts the link generation scheme and the zero position of the PUMA 560. Since the last three joints are intersected, the four links have been considered to represent the robot.

![Figure 5. Link generation scheme for the robot configuration at zero position of PUMA 560](image)

Let \( L(x,y,z,t) \) be the set of Cartesian points of these four lines, which represent the links of the manipulator at the time instant ‘\( t \)’. It is expressed in (5):

\[
L(x,y,z,t) = \begin{bmatrix} L_1 \cr L_2 \cr L_3 \cr L_4 \end{bmatrix} = \begin{bmatrix} P_B \cr P_{11} \cr P_{12} \cr P_{13} \cr P_{14} \end{bmatrix} + u \begin{bmatrix} \Delta P_{11} \cr \Delta P_{12} \cr \Delta P_{13} \cr \Delta P_{14} \end{bmatrix}, \quad 0 \leq u \leq 1
\]  

(5)

This restricts the free movement of the robot links and end-effector, and imposes an additional constraint.

2.4 Problem definition

The problem can be stated briefly thus:

"Determination of smooth (jerk-limited) and minimum-time trajectory of 6-DOF robotic manipulator, given the Cartesian position and orientation of end-effector at the pick-and-place destinations in its dexterous workspace, under the constraints of kinematic limitation on jerk, acceleration and velocity, and obstacle".

2.5 Mathematical Model

The mathematical formulation is as follows:

Objective Function:

\[
\text{Minimize} \left\{ T_{\text{config}} \right\}
\]

(6)

Subject to

\[
\theta_n^{\text{min}}(t) \leq \theta_n(t) \leq \theta_n^{\text{max}}(t), \quad \forall n = 1,2,...6 \quad \text{Position Constraint (7)}
\]

\[
\left| \dot{\theta}_n(t) \right| \leq \omega_n^{\text{max}}, \quad \forall n = 1,2,...6 \quad \text{Velocity Constraint (8)}
\]

\[
\left| \ddot{\theta}_n(t) \right| \leq \omega_n^{\text{max}}, \quad \forall n = 1,2,...6 \quad \text{Acceleration Constraint (9)}
\]

\[
\left| \dddot{\theta}_n(t) \right| \leq J_n^{\text{max}}, \quad \forall n = 1,2,...6 \quad \text{Jerk Constraint (10)}
\]

\[
L(x,y,z,t) \in R(x_{\text{eb}}, y_{\text{eb}}, z_{\text{eb}}) \quad \text{Obstacle Constraint (11)}
\]

3. Proposed Trajectory Planner

The manipulator joint-level information has been considered for generating optimal trajectories in most contributions to the literature. However, offline programming requires defining the pick-and-place destinations with Cartesian space description. Taking this into consideration, the proposed planner is designed to execute smooth, time-optimal pick-and-place operations with the use of Cartesian description of end-effector at pick-and-place destinations without any user-defined intermediate way points or knot points. An enumerative procedure is proposed in pursuit of a smooth and minimum-time trajectory of a 6-DOF PUMA robot for handling fragile parts. The algorithm involves two stages: stage 1 searches for feasible trajectories without a via-point. Where there is no feasible solution in stage 1, the planner proceeds to stage 2 to search for feasible paths with a via-point. Figure 6 depicts the two-stage structure of the proposed planner. This section illustrates the different modules of the two-stage automated trajectory planner.

3.1 Data Input

The data related to robot link and kinematic parameters, robot configurations at pick-and-place destinations, and the obstacle, are given as input. Consider the following data for illustration. The link parameters of PUMA 560 are shown in Table 2 [30] and its kinematic parameter limits are given in Table 3 [1]. Table 4 provides the position and orientations of the end-effector at pick-and-place destinations. The obstacle is assumed to be a cube of side 0.45 m and it is located with one of its base corners at a distance 0.2 m, 0 m and 0.8 m from the robot base.

<table>
<thead>
<tr>
<th>Joint, ( n )</th>
<th>Offset angle about X-axis, ( \phi_n ) (deg)</th>
<th>Offset distance along X-axis, ( d_n ) (mm)</th>
<th>Distance along Z-axis, ( d_z ) (mm)</th>
<th>Joint angle range about Z-axis, ( \theta_n ) (deg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>2</td>
<td>-160 to +160</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>431.8</td>
<td>189.09</td>
<td>-225 to +45</td>
</tr>
<tr>
<td>3</td>
<td>90</td>
<td>-20.32</td>
<td>0</td>
<td>-45 to +225</td>
</tr>
<tr>
<td>4</td>
<td>-90</td>
<td>433.87</td>
<td>-119 to +179</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>90</td>
<td>0</td>
<td>-100 to +100</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>56.25</td>
<td>-260 to +266</td>
<td></td>
</tr>
</tbody>
</table>

Table 2. Denavit Hartenberg (DH) parameters of PUMA 560 [30]
Table 3. Kinematic parameters of PUMA 560

<table>
<thead>
<tr>
<th>Kinematic Limits</th>
<th>Joint Description (Taken from [1])</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Joint 1</td>
</tr>
<tr>
<td>Velocity, $v_{\text{max}}$ rad/s</td>
<td>8</td>
</tr>
<tr>
<td>Acceleration, $a_{\text{max}}$ rad/s²</td>
<td>10</td>
</tr>
<tr>
<td>Jerk, $j_{\text{max}}$ rad/s³</td>
<td>30</td>
</tr>
</tbody>
</table>

Table 4. Position and orientation of the end-effector in Cartesian space

<table>
<thead>
<tr>
<th>Robot configuration</th>
<th>Position (in metres)</th>
<th>Orientation (in radians)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$P_x$</td>
<td>$P_y$</td>
</tr>
<tr>
<td>At pick destination, 'P'</td>
<td>0.607</td>
<td>-0.150</td>
</tr>
<tr>
<td>At place destination, 'Q'</td>
<td>0.149</td>
<td>0.042</td>
</tr>
</tbody>
</table>

Figure 6. Flow chart of the proposed trajectory planner
3.2 Determination of Cartesian coordinate of obstacle

When all parametric variables \( u_1, u_2, u_3 \) are zero, \( R(x_{c1}, y_{c2}, z_{c3}) \) represents the point \( P_1 \). If any one of these variables \( u_1, u_2, u_3 \) is one and other two variables are zero, \( R(x_{c1}, y_{c2}, z_{c3}) \) represents the extreme points \( P_1, P_2 \) and \( P_3 \) respectively. When all these variables reach their upper limit, one, then \( R(x_{c1}, y_{c2}, z_{c3}) \) denotes a corner of the cube which is diagonal to \( P_1 \). Therefore, a set of discretized points \( R(x_{c1}, y_{c2}, z_{c3}) \) is generated by varying each of the parametric variables whilst keeping the other two variables constant. In other words, each combination of the parametric variables provides a point on the obstacle. If each variable is incremented with 0.1 in all directions, this results in 1331 (11x11x11) points in the set \( R(x_{c1}, y_{c2}, z_{c3}) \).

3.3 Calculation of all eight possible configurations at the destinations

Given the Cartesian description of end-effector in dextrous workspace, there can be eight possible ways for a 6-DOF robotic manipulator to approach a particular destination through its inverse kinematic calculation. There are eight possible configurations as specified in the inverse kinematic solution tree for PUMA 560, as shown in Figure 7, at any Cartesian description of the end-effector in its dextrous workspace. The first three joints of PUMA 560 yield four possible positions with a combination of arm positions, ‘left’ and ‘right’ with ‘top’ and ‘down’, i.e., the first joint positions have four combinations of \( \theta_1, \theta_2 \) and \( \theta_3 \) and their suffixes, ‘a’, ‘b’, ‘c’, and ‘d’, represent label of combination.

![Figure 7. Inverse kinematic solution tree for PUMA 560 robotic manipulator](image)

Similarly, the wrist with \( \theta_4, \theta_5 \) and \( \theta_6 \) has two kinds of orientation, ‘no flip’ and ‘flip’, for each combination of arm positions. The expressions (12), (13) and (14) show the relationship between wrist orientations ‘no flip’ and ‘flip’ [9]:

\[
\theta_{4a} = \theta_{4b} + 180^\circ \quad (13)
\]

\[
\theta_{5a} = \theta_{5b} + 180^\circ \quad (14)
\]

Table 5 and Table 6 provide the possible robot configurations at pick, ‘P’ and place, ‘Q’ destinations, respectively for the position and orientations of the end effector, as described in Table 4.

<table>
<thead>
<tr>
<th>Joint configurations</th>
<th>Index</th>
<th>Elbow</th>
<th>Shoulder</th>
<th>Wrist</th>
<th>( \phi_1 ) (rad)</th>
<th>( \phi_2 ) (rad)</th>
<th>( \phi_3 ) (rad)</th>
<th>( \phi_4 ) (rad)</th>
<th>( \phi_5 ) (rad)</th>
<th>( \phi_6 ) (rad)</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>Left</td>
<td>Up</td>
<td>No Flip</td>
<td>2.657</td>
<td>2.140</td>
<td>-0.236</td>
<td>-1.665</td>
<td>2.499</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P2</td>
<td>Left</td>
<td>Up</td>
<td>Flip</td>
<td>2.657</td>
<td>2.140</td>
<td>0.000</td>
<td>-2.906</td>
<td>1.665</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P3</td>
<td>Left</td>
<td>Down</td>
<td>No Flip</td>
<td>2.657</td>
<td>2.665</td>
<td>-0.048</td>
<td>-1.027</td>
<td>-0.276</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P4</td>
<td>Left</td>
<td>Down</td>
<td>Flip</td>
<td>2.657</td>
<td>3.656</td>
<td>0.000</td>
<td>-2.114</td>
<td>0.390</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P5</td>
<td>Right</td>
<td>Up</td>
<td>No Flip</td>
<td>0.000</td>
<td>1.001</td>
<td>-1.048</td>
<td>-3.142</td>
<td>-1.523</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P6</td>
<td>Right</td>
<td>Up</td>
<td>Flip</td>
<td>0.000</td>
<td>1.001</td>
<td>0.000</td>
<td>1.523</td>
<td>2.094</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P7</td>
<td>Right</td>
<td>Down</td>
<td>No Flip</td>
<td>0.000</td>
<td>-0.524</td>
<td>0.000</td>
<td>1.523</td>
<td>-1.244</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P8</td>
<td>Right</td>
<td>Down</td>
<td>Flip</td>
<td>0.000</td>
<td>-0.524</td>
<td>0.000</td>
<td>-2.045</td>
<td>0.000</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 6. All possible robot configurations at ‘Q’

3.4 Generation of 64 possible paths for all combinations of joint configurations between destinations

Considering all combinations of robot configurations at pick-and-place destinations, there are 64 possible ways to move between destinations, i.e., combination of each robot configuration of ‘P’ with eight robot configurations of ‘Q’ results in 64 (8x8) paths between the destinations given in Table 7.

Table 7. All 64 paths between the destinations ‘P’ and ‘Q’

3.5 Evolution and Evaluation of trajectories

This module generates STST, finds the execution time and checks for feasibility (i.e., collision-free) for all 64 paths. Figure 8 provides the sequence of steps involved, which are delineated in this section.

3.5.1 Input: Joint configuration

The joint description (position and orientation) of the robot’s joints \( \theta_1^P \) and \( \theta_1^Q \) at pick destination ‘P’ and place destination ‘Q’ is the input for the generation of STST. As an example, Table 8 provides the joint configuration of path no. 62 with its configurations P8 and Q6 at pick and place destinations respectively.
Each trajectory requires a specific displacement $\Delta \theta_{PQ}^n$ for the move from ‘P’ to ‘Q’, and this is found using (15):

$$\Delta \theta_{PQ}^n = \theta_{Q}^n - \theta_{P}^n$$

The required displacements of the joints for path no. 62 (Table 8) are:

$$\Delta \theta_{PQ}^n = [\Delta \theta_1^PQ, \Delta \theta_2^PQ, \Delta \theta_3^PQ, \Delta \theta_4^PQ, \Delta \theta_5^PQ, \Delta \theta_6^PQ]$$

$$= [1.600, 3.142, 2.450, 2.623, 2.187, -0.347] \text{ rad}$$

### 3.5.3 Calculation of minimum execution time and kinematic parameters of all joints

First, the lower bound of execution time is calculated. The kinematic parameters $T_{\text{max}}^n$, $A_{\text{max}}^n$, $V_{\text{max}}^n$ limit the time for executing the motions during each of the three phases. The minimum time $(T_n^{L})^i$ and its corresponding maximum displacement $(\Delta \theta_n^{L})^u$ during the acceleration phase are found using (16) and (17) respectively:

$$(T_n^{L})^i = t_4 - t_3 = \frac{A_{\text{max}}^n \pi}{T_{\text{max}}^n}$$

$$(\Delta \theta_n^{L})^u = \frac{1}{4} A_{\text{max}}^n \left( (T_n^{L})^i \right)^2$$

Similarly, the minimum time $(T_n^{S})^i$ and its maximum displacement $(\Delta \theta_n^{S})^u$ during the deceleration phase are given in (18) and (19) respectively:

$$(T_n^{S})^i = t_4 - t_3 = \frac{A_{\text{max}}^n \pi}{T_{\text{max}}^n}$$

$$(\Delta \theta_n^{S})^u = \frac{1}{4} A_{\text{max}}^n \left( (T_n^{S})^i \right)^2$$

The required displacement during the acceleration and deceleration phases are calculated as in (20) and (21):

$$\Delta \theta_n^i = \begin{cases} \frac{\Delta \theta_n^{L}}{2} & \text{if } \Delta \theta_n^{L} \geq \left| \left( \Delta \theta_n^{L} \right)^u \right| + \left| \left( \Delta \theta_n^{S} \right)^u \right| \\ \frac{\Delta \theta_n^{S}}{2} & \text{if } \Delta \theta_n^{S} < \left| \left( \Delta \theta_n^{L} \right)^u \right| + \left| \left( \Delta \theta_n^{S} \right)^u \right| \end{cases}$$

The displacement during the constant velocity phase $\Delta \theta_n^V$ is calculated using (20) and (21) as given in (22):

$$\Delta \theta_n^V = \begin{cases} \Delta \theta_n^{PQ} - \Delta \theta_n^{L} - \Delta \theta_n^{S} & \text{if } \Delta \theta_n^{PQ} > \left| \left( \Delta \theta_n^{L} \right)^u \right| + \left| \left( \Delta \theta_n^{S} \right)^u \right| \\ 0 & \text{if } \Delta \theta_n^{PQ} < \left| \left( \Delta \theta_n^{L} \right)^u \right| + \left| \left( \Delta \theta_n^{S} \right)^u \right| \end{cases}$$

Therefore, the minimum time for the constant velocity phase is given in (23).

$$(T_n^{V})^i = t_3 - t_2 = \frac{\Delta \theta_n^V}{V_{\text{max}}^n}$$

The minimum execution time of the joint ‘$n$’ during a robot motion cycle is given in (24):

$$(T_n^{PQ})^i = (T_n^{L})^i + (T_n^{V})^i + (T_n^{S})^i$$

Since the minimum time for the execution of the acceleration phase cannot be further reduced without increasing the jerk, the kinematic parameters $A_{\text{max}}^n$, $V_{\text{max}}^n$, $T_{\text{max}}^n$ are to be adjusted/tuned to $A_{n}^L$, $I_{n}^L$ and $V_{n}^L$ for the required displacement, and are provided in (25), (26) and (28) respectively:

$$A_{n}^L = \frac{4 \Delta \theta_n^{L}}{(T_n^{L})^i}$$

$$I_{n}^L = \frac{A_{n}^L \pi}{(T_n^{L})^i}$$

$$V_{n}^L = \frac{A_{n}^L \pi}{(T_n^{L})^i}$$
The jerk and acceleration values will affect the velocity of each joint, calculated as in (27):

$$V_n^L = \left(\frac{A_n^L}{2f_n^L}\right)^2 \pi$$  \hspace{1cm} (27)

During the constant velocity phase, the acceleration and jerk are zero-valued, i.e., $A_n^V = 0$ and $J_n^V = 0$, and the velocity $V_n^V$ equals the velocity limit of the acceleration phase $V_n^L$. It is kept constant throughout this phase.

$$V_n^V = V_n^L$$  \hspace{1cm} (28)

The shape of the deceleration phase is symmetrical and has negative kinematic parameters. The equations (29) and (30), respectively, provide the acceleration and jerk limits for the deceleration phase.

$$A_n^S = \frac{44\phi_n^S}{\left(\frac{\pi}{V_n^S}\right)^2}$$  \hspace{1cm} (29)

$$J_n^S = \frac{A_n^S\pi}{\left(\frac{\pi}{V_n^S}\right)}$$  \hspace{1cm} (30)

Then, the velocity of the deceleration phase is calculated as in (31):

$$V_n^S = \frac{\left(\frac{A_n^S}{\pi}\right)^2}{2f_n^S}$$  \hspace{1cm} (31)

Considering the kinematic bounds as described in Table 2, the time and kinematic limits required for each joint in respect to the displacements are calculated using expressions (15) to (31) for an unsynchronized motion for the given input (Table 8). It is assumed that the time for both the acceleration and deceleration phases is equal, i.e., $T_n^a = T_n^d$. The time and corresponding kinematic parameters are calculated for all joints and tabulated in Table 9 as follows:

<table>
<thead>
<tr>
<th>Parameters</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\phi_n^S$ (rad)</td>
<td>1.000</td>
<td>3.142</td>
<td>2.450</td>
<td>2.025</td>
<td>2.187</td>
<td>-0.347</td>
</tr>
<tr>
<td>$\phi_n^L$ (rad)</td>
<td>2.742</td>
<td>2.665</td>
<td>2.665</td>
<td>3.158</td>
<td>3.158</td>
<td>3.158</td>
</tr>
<tr>
<td>$\phi_n^S$ (rad)</td>
<td>0.800</td>
<td>0.571</td>
<td>0.225</td>
<td>1.312</td>
<td>1.094</td>
<td>-0.174</td>
</tr>
<tr>
<td>$\phi_n^L$ (sec)</td>
<td>1.047</td>
<td>0.942</td>
<td>0.942</td>
<td>1.265</td>
<td>1.265</td>
<td>1.265</td>
</tr>
<tr>
<td>$\phi_n^S$ (sec)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$\phi_n^L$ (sec)</td>
<td>1.047</td>
<td>0.942</td>
<td>0.942</td>
<td>1.265</td>
<td>1.265</td>
<td>1.265</td>
</tr>
<tr>
<td>$\phi_n^S$ (sec)</td>
<td>2.094</td>
<td>1.885</td>
<td>1.885</td>
<td>2.513</td>
<td>2.513</td>
<td>2.513</td>
</tr>
<tr>
<td>$\phi_n^L$ (sec)</td>
<td>2.918</td>
<td>7.074</td>
<td>5.516</td>
<td>3.322</td>
<td>2.770</td>
<td>-0.439</td>
</tr>
<tr>
<td>$\phi_n^S$ (sec)</td>
<td>8.754</td>
<td>23.582</td>
<td>10.388</td>
<td>8.305</td>
<td>6.925</td>
<td>-1.099</td>
</tr>
<tr>
<td>$\phi_n^L$ (sec)</td>
<td>3.056</td>
<td>6.668</td>
<td>5.169</td>
<td>4.175</td>
<td>3.481</td>
<td>-0.552</td>
</tr>
<tr>
<td>$\phi_n^S$ (sec)</td>
<td>0.800</td>
<td>1.571</td>
<td>1.225</td>
<td>1.312</td>
<td>1.094</td>
<td>-0.174</td>
</tr>
<tr>
<td>$\phi_n^L$ (sec)</td>
<td>-2.918</td>
<td>-7.074</td>
<td>-5.516</td>
<td>-3.322</td>
<td>-2.770</td>
<td>0.439</td>
</tr>
<tr>
<td>$\phi_n^S$ (sec)</td>
<td>-0.754</td>
<td>-23.582</td>
<td>-10.388</td>
<td>-8.305</td>
<td>-6.925</td>
<td>1.099</td>
</tr>
<tr>
<td>$\phi_n^L$ (sec)</td>
<td>3.056</td>
<td>6.668</td>
<td>5.169</td>
<td>4.175</td>
<td>3.481</td>
<td>-0.552</td>
</tr>
</tbody>
</table>

Table 9. Time and kinematic information of unsynchronized motion for the 62nd path

3.5.4 Calculation of Synchronized execution time of S-curve trajectory and its kinematic joint limits

The trajectories are controlled by the combined acceleration and jerk values for the acceleration and deceleration phases. Specifically, these phases are influenced by the ratio of jerk and acceleration values. When jerk reaches its maximum bound, the ratio of jerk to acceleration is considered to be the trajectory frequency $\omega_n^L$, with which the pattern of trajectory of each kinematic parameter will be set, given in (27).

$$\omega_n^L = \frac{2f_n^L}{A_n^L}$$  \hspace{1cm} (32)

The trajectory frequency of the acceleration phase for the values given in Table 9 is: $\omega_n^L = [6.000 \ 6.667 \ 6.667 \ 5.000 \ 5.000 \ 5.000] \text{ (cycles/sec)}$.

Similarly to the acceleration phase, the deceleration phase of the trajectory has the frequency $\omega_n^S$, which is given in (33):

$$\omega_n^S = \frac{2f_n^S}{A_n^S}$$  \hspace{1cm} (33)

Since the times for the acceleration and the deceleration phases are equal ($T_n^a = T_n^d$) for the values given in Table 9, the trajectory frequency of the deceleration is $\omega_n^S$, i.e., $\omega_n^S = [6.000 \ 6.667 \ 6.667 \ 5.000 \ 5.000 \ 5.000] \text{ (cycles/sec)}$. However, the motions of all joints during each phase should be time-synchronized for the robot to reach ‘L’ and ‘S’ points at the time instants $t_2$ and $t_3$ with STS. Synchronization means that the execution of the required displacement of each phase of motion of all joints is accomplished with the same start and end times. The kinematic parameters can be further reduced by synchronizing the motions of all joints with the slowest joint, the joint with the highest minimum execution time. In order to synchronize the joint motions, the execution times of three phases are calculated based on the slowest joint in their respective phases using (34) to (36).

$$T_n^{L_{sync}} = \max\left(T_n^{L_{j}}\right) \quad n \in \{1,2...6\}$$  \hspace{1cm} (34)

$$T_n^{V_{sync}} = \max\left(T_n^{V_{j}}\right) \quad n \in \{1,2...6\}$$  \hspace{1cm} (35)

$$T_n^{S_{sync}} = \max\left(T_n^{S_{j}}\right) \quad n \in \{1,2...6\}$$  \hspace{1cm} (36)

Then, total execution time for pick and place task is calculated as in (37).

$$T_n^{PQ_{sync}} = T_n^{L_{sync}} + T_n^{V_{sync}} + T_n^{S_{sync}}$$  \hspace{1cm} (37)
Therefore, other joints are operated with lesser kinematic values in order to synchronize their motion and for executing the smooth motion. In this work, it is considered that all the phases of the trajectory (acceleration, constant velocity and deceleration phases) are synchronized i.e. all joints execute the same phase of the trajectory at any instant of time. In order to synchronize all the joints, the synchronized trajectory frequency \( \omega_{\text{sync}} \) is set as minimum value among the trajectory frequencies of the joints and is expressed in (38).

\[
\omega_{\text{sync}} = \min \left( \omega_n \right) \tag{38}
\]

It is assumed that acceleration and deceleration phases operate with equal time interval \( T_{\text{sync}}^L = T_{\text{sync}}^S \) and hence, they have same trajectory frequency, i.e. \( \omega_{\text{sync}} = \omega_{\text{sync}}^S \). The resultant the kinematic values during the acceleration and deceleration phases, are equal in their magnitude but differ in the sign. Then, the kinematic parameters (acceleration \( A_{\text{sync}}^n \), jerk \( J_{\text{sync}}^n \) and velocity \( V_{\text{sync}}^n \)) are updated for synchronization with the help of the synchronized trajectory frequency \( \omega_{\text{sync}}^L \) as given in (39), (40) and (41), respectively. The synchronized motion for the inputs given in Table 8 can be achieved through the expressions (34) to (41) and are given in Table 10.

\[
A_{\text{sync}}^n = \frac{4 \Delta \theta_n^L}{(T_{\text{sync}}^L)^2} \tag{39}
\]

\[
J_{\text{sync}}^n = \frac{\omega_{\text{sync}}^L A_{\text{sync}}^n}{2} \tag{40}
\]

\[
V_{\text{sync}}^n = \frac{A_{\text{sync}}^n \pi}{\omega_{\text{sync}}^L} \tag{41}
\]

<table>
<thead>
<tr>
<th>Parameters</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \Delta \theta ) (rad)</td>
<td>1.600</td>
<td>3.142</td>
<td>2.450</td>
<td>2.023</td>
<td>2.187</td>
<td>-0.347</td>
</tr>
<tr>
<td>( \Delta \theta_n ) (deg)</td>
<td>0.800</td>
<td>1.571</td>
<td>1.225</td>
<td>1.312</td>
<td>1.094</td>
<td>-0.174</td>
</tr>
<tr>
<td>( T_{\text{sync}}^L ) (sec)</td>
<td>1.216</td>
<td>1.753</td>
<td>2.513</td>
<td>5.000</td>
<td>1.273</td>
<td>2.500</td>
</tr>
<tr>
<td>( T_{\text{sync}}^S ) (sec)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( V_{\text{sync}}^n ) (rad/sec)</td>
<td>5.000</td>
<td>9.948</td>
<td>7.757</td>
<td>8.305</td>
<td>6.925</td>
<td>-1.099</td>
</tr>
<tr>
<td>( A_{\text{sync}}^n ) (rad/sec^2)</td>
<td>2.026</td>
<td>3.979</td>
<td>3.105</td>
<td>3.322</td>
<td>2.770</td>
<td>-0.439</td>
</tr>
<tr>
<td>( J_{\text{sync}}^n ) (rad/sec^3)</td>
<td>5.066</td>
<td>9.378</td>
<td>7.757</td>
<td>8.305</td>
<td>6.925</td>
<td>-1.099</td>
</tr>
</tbody>
</table>

Table 10. Time and kinematic information of synchronized motion for the 62nd path

3.5.5 Generation of time history of kinematic parameters of all joints for the synchronized motion

Once, the kinematic limits are identified, the time history (trajectory) of each kinematic parameter is generated for synchronized motion of the robot in carrying out the pick-and-place operation. Time history of joints may be with or without a constant velocity phase. For example, consider the following: \( \theta_0^P = 0 \text{deg} \) and \( \theta_0^Q = 100 \text{deg} \) are joint positions at initial destination ‘P’, and final destination ‘Q’, respectively, and whose maximum kinematic limits are \( V_{\text{max}}^1 = 10 \text{deg/sec} \); \( A_{\text{max}}^1 = 4 \text{deg/sec}^2 \); \( J_{\text{max}}^1 = 2.5 \text{deg/sec}^3 \). Figure 9 depicts the time history of STST with the constant velocity phase generated using the expressions (15) to (31).

Figure 9. Time history of joint (with constant velocity phase): a) Displacement, b) Velocity, c) Acceleration, d) Jerk

If the maximum kinematic limits of the joint are \( V_{\text{max}} = 20 \text{deg/sec} \); \( A_{\text{max}} = 8 \text{deg/sec}^2 \); \( J_{\text{max}} = 5 \text{deg/sec}^3 \) with the same joint positions at ‘P’ and ‘Q’, then the corresponding STST is generated without the constant velocity phase, as shown in Figure 10.

Figure 10. Time history of joint (without constant velocity phase): a) Displacement, b) Velocity, c) Acceleration, d) Jerk

With the help of the expressions for synchronization (15) to (41) and by operating at kinematic values given in Table 3, the STST for the configurations of the 62nd path are generated and are shown in Figure 11. The time for executing the path between these configurations given in Table 8 is 2.513 sec and minimum of maximum jerk value is 9.9485 rad/sec^3.

Figure 11. STST of PUMA 560 for 62nd path: a) Displacement, b) Velocity, c) Acceleration, d) Jerk
3.5.6 Generation of Cartesian coordinates for links at all time instants

With the help of the trajectory that has been generated in joint space, the Cartesian coordinate values of the links have been calculated/generated at every time instant of the synchronized motion. For the known robot joint space information, the Cartesian coordinate values of the joints are calculated using suitable transformation matrices of the forward kinematics, as described in section 2.3. The information on the links has been updated at every STST time instant.

3.5.7 Simulation of Collision check

In a working environment with an obstacle as shown in Figure 12, the synchronized trajectories generated in section 3.5.5 may be feasible, i.e., collision-free (represented by the continuous line) or infeasible (represented by the dotted line). For the known joint configurations at every instant of time, the path of the end-effector is generated with the help of forward kinematics. In order to generate a feasible trajectory, a collision check procedure is proposed between the links and the obstacle at all time instants of synchronized time during the robot’s operation.

![Figure 12. Schematic sketch of robot positions in its trajectory](image)

The path that is generated between the robot configurations of the destinations is evaluated for interference/collision of links with the obstacle and is labelled as ‘feasible’ or ‘infeasible’. Cartesian coordinate values of these links \( L(x,y,z,t) \) as generated in section 2.3 are compared with the Cartesian coordinate values of the obstacle \( R(x_ob,y_ob,z_ob) \); the difference between these Cartesian values should be greater than 0.01 m (assuming that the maximum thickness of the links of the robot is approximately 0.01 m) along X, Y and Z directions at all time instants. If the difference is less than 0.01 m, it indicates that the link has a contact with the obstacle and the corresponding path is labelled as ‘infeasible’. Otherwise, the collision check procedure will be evaluated for the next time instant and this process will be continued throughout the synchronized execution time \( T_{sync} \). The successful path in the collision check will be labelled as ‘feasible’. In order to reduce the algorithm’s computational effort, once collision is detected at any time instant, the proposed algorithm will exit immediately from the time loop, avoiding unwanted computations. Path no. 62, presented in section 3.5, has been labelled as ‘infeasible’ after the collision check, i.e., the robot makes contact with the obstacle during its motion along this path. All paths are evaluated and labelled in this way. Table 12 provides the execution time, maximum jerk and feasibility status of 64 paths generated by the algorithm.

3.5.8 Selection of best feasible path and configurations

It is found that 48 feasible solutions (paths) are available among 64 solutions. The best feasible solution is selected based on minimum execution time and minimum jerk value. The path number 16 with pick-and-place configurations, P2 and Q8 respectively, whose path time of 2.513 sec and minimum jerk value of 6.151 rad/sec\(^3\) represent the best solution for the input given in section 3.1. This is the only path with both minimum time and minimum jerk value. The torque values have been calculated using Robotics Toolbox [31] to understand the dynamic behaviour of this proposed trajectory for the feasible paths.

![Table 11. Results of the algorithm for 64 possible paths from Cartesian space description](image)

Figure 13 depicts the corresponding trajectories of displacement, velocity, acceleration, jerk and torque for the optimal solution, i.e., trajectories for all the joints of PUMA 560 robot during its motion along path no. 16 with configuration P2 and Q8.

3.6 Generation of feasible trajectories with via-point

When all the 64 paths become ‘infeasible’ for the particular input, stage two of the planner searches for feasible path(s) between the pick and place destinations with a via-point.

In order to illustrate the above case, the data on the end-effector given in Table 12 have been considered without any change in robot link parameters and using kinematic limits as specified in Tables 2 and 3.
3.6.1 Selection of the via point for feasible trajectory generation

Under circumstances where all 64 paths generated from the Cartesian description of the end-effector fail, a via-point ‘V’ is needed to obtain a feasible path. The following procedure is proposed in this work to select a ‘via-point’ ‘V’ in the robot’s dexterous workspace to generate a collision-free path.

3.6.1.1 Generation of forbidden sphere

The work proposes a ‘forbidden sphere’ technique for the selection of via-point to generate a feasible trajectory. A forbidden enclosing sphere connecting the corners of cube (the obstacle) is developed which encapsulates the obstacle. The centre of the forbidden sphere \( P_f(x_f,y_f,z_f) \) and its radius \( r_f \) are calculated from \( P_c(x_c,y_c,z_c) \), the Cartesian position of the reference base corner of the cube, and are given as in (42), (43) (44) and (45):

\[
\begin{align*}
x_c &= x_r + a / 2 \\
y_c &= y_r + a / 2 \\
z_c &= z_r + a / 2 \\
r_f &= \frac{\sqrt{3}}{4}a
\end{align*}
\]

The points on the surface of the forbidden sphere \( (x_f,y_f,z_f) \) are calculated using its spherical coordinates \( (r_f,\phi_f,\psi_f) \), and their corresponding expressions are given in (46), (47) and (48):

\[
\begin{align*}
x_f &= x_c + r_f \sin \phi_f \cos \psi_f \\
y_f &= y_c + r_f \sin \phi_f \sin \psi_f \\
z_f &= z_c + r_f \cos \phi_f
\end{align*}
\]

where \( 0 \leq \phi_f \leq \pi \) and \( 0 \leq \psi_f \leq 2\pi \)

3.6.1.2 Generation of via-point sphere

A set of three via-point spheres concentric to the forbidden sphere (sphere encompasses the obstacle) has been constructed to facilitate the selection of a via-point which emulates the workspace of the articulated robotic manipulator (a partial sphere in shape). Figure 14 depicts the arrangement of forbidden sphere and via-point sphere (only one via-point sphere has been shown for ease of understanding). A via-point sphere is generated from the centre of the forbidden sphere, i.e., \( P_f(x_c,y_c,z_c) \) and its calculated radius \( r_v \), as given in (49):

\[
r_v = r_f + \text{offset}
\]
Two more via-point spheres are generated with the multiples of the same offset distances. Offset value is selected in such a way that the via-point spheres fall in the robot’s dexterous workspace. Each sphere has been discretized to have 200 Cartesian coordinate points. All the points on the via-point spheres are stored as \( R(x, y, z) \), the pool of points for via-point selection. The trajectory planner selects a via-point randomly as required from a look-up table \( R(x, y, z) \). This selected via-point is considered as the Cartesian position of the end-effector and its orientation is assumed to be the average of the orientations at the pick-and-place destination. Position and orientation form the required transformation matrix which provides the Cartesian description at ‘V’. Table 13 presents the position and orientation of one such selected via-point and its corresponding transformation matrix. Similar to pick-and-place destinations, the possible configurations at a selected via-point destination is given in Table 14.

### Table 13. Position and orientation of the end-effector for the selected via-point

<table>
<thead>
<tr>
<th>Configuration Index</th>
<th>Position (in metres)</th>
<th>Orientation (in radians)</th>
<th>End-effector description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Px</td>
<td>Py</td>
<td>Pz</td>
</tr>
<tr>
<td>V1</td>
<td>4.318</td>
<td>1.243</td>
<td>0.330</td>
</tr>
<tr>
<td>V2</td>
<td>4.318</td>
<td>1.243</td>
<td>0.330</td>
</tr>
<tr>
<td>V3</td>
<td>4.318</td>
<td>3.099</td>
<td>2.905</td>
</tr>
<tr>
<td>V4</td>
<td>4.318</td>
<td>3.099</td>
<td>2.905</td>
</tr>
<tr>
<td>V5</td>
<td>2.123</td>
<td>1.899</td>
<td>2.905</td>
</tr>
<tr>
<td>V6</td>
<td>2.123</td>
<td>1.899</td>
<td>2.905</td>
</tr>
<tr>
<td>V7</td>
<td>2.123</td>
<td>0.043</td>
<td>0.330</td>
</tr>
<tr>
<td>V8</td>
<td>2.123</td>
<td>0.043</td>
<td>0.330</td>
</tr>
</tbody>
</table>

### Table 14. All eight possible robot configurations at via destination

#### 3.6.2 Calculation of execution time for trajectory with via-point

There are two STSTs in which the first trajectory is generated between pick and via-point destinations, and another trajectory between the via-point and place destinations. Each of these trajectories has an acceleration, constant velocity and deceleration phase, and the corresponding time for each trajectory has been calculated as specified in section 3.5.3. The sum of time required for these trajectories is the time required for the execution of the pick-and-place operation and is calculated as in (50):

\[
T_{\text{sync}}^{1Q} = T_{\text{sync}}^{PV} + T_{\text{sync}}^{VQ}
\]

In this case, the time for the trajectory between pick and place, \( T_{\text{sync}}^{1Q} \), depends on the location of via-point ‘V’.

#### 3.6.3 Generation of trajectories with via-point and evaluation of paths for feasibility

By considering all possible configurations at ‘V’, the resultant 512 (8x8x8) paths are generated by this algorithm connecting pick, via and place destinations. These paths are in turn evaluated for collision, as specified in section 3.5.7, and the feasible path(s) is/are identified. Table 15 provides the execution time, jerk and feasibility status of 512 paths generated by the algorithm.

![Figure 14. Via-point generation for obstacle avoidance](image)

3.6.4 Selection of best feasible path and configurations

The least time-consuming path with minimum jerk is selected as the optimal path interpolating pick, via and place destinations. For the above-given input data, the minimum execution time is calculated as 5.03 seconds, which is achieved in 321 instances out of 512. The path number 434 with respective pick, via and place configurations P7, V7 and Q2, and the path number 505 with respective pick, via and place configurations P8, V8 and Q1, whose minimum jerk limit was 12.872 rad/sec and minimum path time 5.03 sec, are the best solutions for the given inputs. These paths show the lowest time and jerk values among all other feasible paths.

![Figure 15 and Figure 16 depict the corresponding trajectories of displacement, velocity, acceleration, jerk and torque for the optimal solutions, i.e., trajectories for all the joints of the PUMA 560 robot during its motion along the paths 434 and 505, respectively.](image)
When it fails to obtain a feasible path, the algorithm selects another via-point randomly from $R(x_v,y_v,z_v)$ and identification of feasible paths will be carried out again. This procedure will be continued until a feasible path is obtained. The proposed algorithm has the difficulty of ensuring the optimal path; also, it is assumed that the orientation at via-point is the average of the orientation of destinations, which may not ensure the best configuration at via-point.

4. Results and Discussion

This section first discusses the performance of the proposed S-curve trajectory generated with the sine wave form of the jerk profile by comparing the cubic spline trajectory with the constant jerk profile presented by Chettibi et al. [1] for the weighted cost function, giving a relative importance to minimizing transfer time and joint efforts or power.

The results given in [1], with the various weighting coefficients of transfer time function $\mu$ and joint efforts $\alpha$ between 1 and 0, are used for the comparison. Table 16

![Figure 15. STST of PUMA 560 for one of the optimal solutions, i.e., path no. 434 among 512 paths: a) Displacement, b) Velocity, c) Acceleration, d) Jerk, e) Torque](image1)

![Figure 16. STST of PUMA 560 for one of the optimal solutions, i.e., path no. 505 of 512 paths: a) Displacement, b) Velocity, c) Acceleration, d) Jerk, e) Torque](image2)
compares the performance of the proposed algorithm with the results presented in [1]. It reveals that the proposed trajectory performs better than the cubic spline trajectory when joint efforts are given more weightage than the execution time. Hence, this S-curve trajectory can be implemented in applications where smoothness in object handling is of great significance, to ensure motion with less vibration.

<table>
<thead>
<tr>
<th>S.No</th>
<th>Performance Parameter</th>
<th>Proposed S-curve trajectory</th>
<th>Cubic spline trajectory of Chubiri et al. (2004) [1]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Joint Displacement</td>
<td>Smooth and Gradual</td>
<td>Wavy</td>
</tr>
<tr>
<td>2</td>
<td>Joint Velocity</td>
<td>Single modal with one peak (valley) velocity</td>
<td>Multi-modal with many peaks and valley velocities</td>
</tr>
<tr>
<td>3</td>
<td>JointAcceleration</td>
<td>Smooth wave form without abrupt corners</td>
<td>Various corners with abrupt corners</td>
</tr>
<tr>
<td>4</td>
<td>Joint Jerk</td>
<td>Smooth and sine wave template</td>
<td>Contrast step template</td>
</tr>
<tr>
<td>5</td>
<td>Maximum Joint Jerk</td>
<td>Less than the maximal specified jerk</td>
<td>Equal to the specified maximum jerk value</td>
</tr>
<tr>
<td>6</td>
<td>Execution Time</td>
<td>2.51 sec</td>
<td>Chart’s [1] results</td>
</tr>
</tbody>
</table>

Table 16. Performance comparison of proposed STST

Secondly, the performance is tested for the situations that require a via-point to provide a collision-free trajectory. In order to check the influence of via-point on the optimality of the solution, six experiments, with the same input as used in stage two (trajectory with via-point) (section 3.0), are conducted with via-points selected around the obstacle. Table 17 shows the results.

It can be observed that certain points lead to infeasible solutions and the jerk values for the optimal solution vary with via-point. It can also be seen that there are solutions with the same minimum execution time but they differ minimum jerk value, based on the location of the via-point.

Although execution time is the same for the optimal solutions at various locations of via-point, a differentiation in the jerk values can also be observed in the results. A method is therefore required to identify the best via-point in the dexterous workspace for achieving the optimal solution with both the minimum jerk and time.

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Location of the via point</th>
<th>Best Execution Time (sec)</th>
<th>jerk Limit (rad/sec)</th>
<th>Feasibility Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-0.626</td>
<td>0.328</td>
<td>0.428</td>
<td>5.63</td>
</tr>
<tr>
<td>2</td>
<td>0.194</td>
<td>-0.195</td>
<td>0.340</td>
<td>-1</td>
</tr>
<tr>
<td>3</td>
<td>0.260</td>
<td>0.328</td>
<td>0.428</td>
<td>5.63</td>
</tr>
<tr>
<td>4</td>
<td>0.260</td>
<td>0.328</td>
<td>0.714</td>
<td>5.63</td>
</tr>
<tr>
<td>5</td>
<td>0.260</td>
<td>0.000</td>
<td>0.214</td>
<td>5.63</td>
</tr>
<tr>
<td>6</td>
<td>-0.500</td>
<td>-0.500</td>
<td>0.100</td>
<td>5.63</td>
</tr>
</tbody>
</table>

Table 17. Results of the experiments on the search algorithm for various via-points

It can also be observed that the jerk, acceleration and velocity profiles are similar for alternate optimal solutions (Figure 15 and Figure 16) of the algorithm, although there will be changes in robot pose. With variations in the location of the via-point, the algorithm provides the better solution as illustrated and sometimes provides infeasible solutions as well.

5. Conclusion

An automated trajectory planner for near-optimal trajectory of 6-DOF robotic manipulators for the given Cartesian description of end-effector at pick and place destinations under kinematic constraints has been described in this work. Li et al.’s (2007) acceleration-bounded S-curve trajectory is modified as a jerk-bounded S-curve trajectory (i.e., all trajectories, displacement, velocity and acceleration, are generated within the specified maximum jerk value) so as to obtain a smoother movement without any abrupt changes. A mathematical model with the objective of minimum execution time for pick-and-place operations is formulated by including a ‘forbidden sphere’ technique to avoid obstacles and a jerk-bounded synchronized trigonometric S-curve trajectory to ensure smooth motion. A two-stage algorithm to find the optimal trajectory with or without via-point is proposed. The performance of the proposed trajectory planner is analysed through comparison with the results of [1]. This reveals that the proposed S-curve trajectory is smoother than the cubic spline trajectory. Also, the proposed STST provides better results when joint efforts are considered with higher priority than execution time. Further, the possibility to set maximum bounds for the jerk before executing the trajectory allows for a large stability and safety margin. Since the proposed algorithm ensures that the jerk of the resulting trajectory will be kept under the limit values set a priori by the user, the resulting vibrations will also be kept low. Hence, the S-curve trajectory is suitable for applications in which smoothness (low vibration) in object handling is of great significance. The proposed planner is able to handle the trajectory with and without via-point and traces the smooth, collision-free, near-time-optimal trajectory. The collision-free trajectory is ensured not only for the robot’s end-effector but also for its links. The performance test carried out for the situations requiring a via-point to provide a collision-free trajectory indicates that the optimal solutions with different via-points are the same for minimum time, but differ in terms of jerk values. This requires searches with many via-points in the dexterous
space for a smooth and minimum-time trajectory. A method for identifying suitable via-points to obtain the optimal solution quickly is therefore required. This work has also considered the enumeration technique to identify the minimal robot execution time with all possible configurations of pick, via and place destinations in the dexterous workspace – this is computationally difficult and huge amounts of data are involved. Future work will be devoted to extending this algorithm to find out the optimal via-point by means of suitable intelligent heuristics. This work has considered a single objective function, minimization of execution time which can be extended with more than one objective function, such as minimization of time and jerk and minimization of time and energy.

6. Acknowledgements

The authors thank the Management of Thiagarajar College of Engineering for their co-operation and encouragement, and the use of all the facilities needed to carry out this work. The authors are also very thankful to anonymous referees for valuable suggestions and comments that helped to improve the quality of this work.

7. References