Assignment3 Report AG03

Jarod Tang, Zac Lee, Xinyu Shen

May 2024

Contents

1 Overview **2**

2 Introduction **3**

3 Trajectory Design **4**
3.1 Derivation of Appropriate Polynomial 4
3.1.1 Translational Trajectory 4
3.1.2 Rotational Trajectory 5
3.2 Design Reasoning 7
3.2.1 Case 1: One via point 7
3.2.2 Case 2: Two via points 8
3.3 Trajectory Generation Algorithm 10

4 Trajectory Visualisation **11**
4.1 3D Trajectory Plot 12
4.2 XY Plane Trajectory 13
4.3 ZX Plane Trajectory 14

5 Feasibility and Simulation **15**
5.1 Robot Pose Simulation 15

1 Overview

As a continuation and brief recap of Assignment 1, the configuration of the 4 DoF (degrees of freedom) robotic manipulator being designed to move chess pieces on a chessboard is illustrated below in Figure 1.

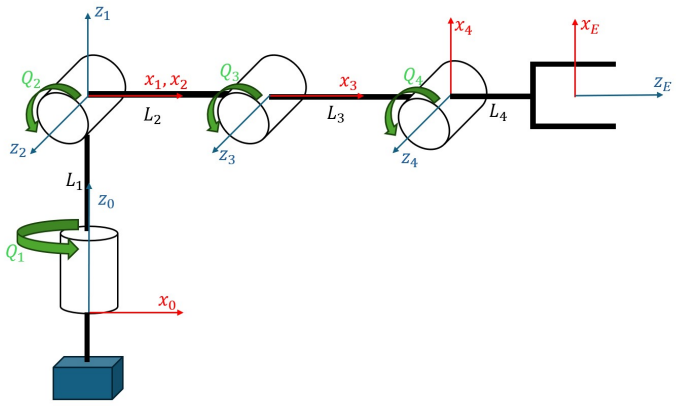


Figure 1: Robot Arm Configuration

The robot's base is elevated 15cm above the chessboard. Its position in the XY plane relative to the chessboard is depicted below in Figure 2 as a red dot, noting that this is different to the coordinates given in the assignment document. The reason for this is such that the chessboard coordinates are aligned with the robot coordinates (as seen in Figure 1). Furthermore, the robot is positioned $0.1m$ from the chessboard '0' position depicted as a blue dot. The robotic manipulator has four linkages. Each linkage length is as follows.

- $l_1 = 150$mm
- $l_2 = 250$mm
- $l_3 = 200$mm
- $l_4 = 120$mm

The dimensions of the chessboard being used are **36cm** $\times$ **36cm** $\times$ **0.3cm**, with each square having the dimensions of **3cm** $\times$ **3cm**. Each chess piece has a radius of **1cm**. The height of the tallest (queen) and shortest (pawn) chess pieces is listed below.

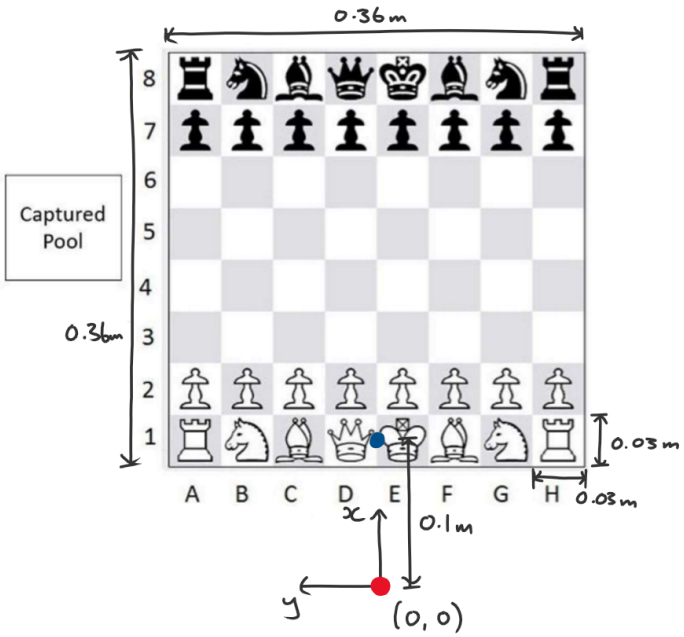


Figure 2: Robot Arm Position in XY Plane

- Queen = 56.3mm
- Pawn = 31.85mm

2 Introduction

This report aims to demonstrate and confirm the viability of the trajectory generation algorithm, which moves the end effector from a given initial pose x_0 and initial velocity $\dot{x}_0$ to a given final pose x_f and final velocity $\dot{x}_f$. Firstly, a trajectory generation algorithm is constructed, given the initial and final end-effector pose and velocities. Secondly, a trajectory is generated for the following move: knight from A5 to C6. Lastly, the feasibility of the result is demonstrated.

3 Trajectory Design

3.1 Derivation of Appropriate Polynomial

Polynomials are good options for interpolating the translational and rotational trajectory. They can be mapped to a smooth curve that meet constraints pertaining to the trajectory. The order of the polynomial would be dependent on how many constraints there are.

3.1.1 Translational Trajectory

To interpolate the translational trajectory, it is first important to recognise that there are 6 constraints. The initial and final values of acceleration, velocity and displacement. For 6 constraints, a polynomial with 6 variables is needed, hence the quintic polynomial shown in equation 1 is a suitable choice.

$$x(t) = at^5 + bt^4 + ct^3 + dt^2 + et + f \tag{1}$$

The initial and final acceleration values can be set to 0 for both the initial and final position, to regulate the amount of jerk. The velocity values will need to meet the given initial ($\dot{x_i}$) and final ($\dot{x_f}$) velocity values. The initial (x_i) and final (x_f) displacements should be given, depending on where the chess pieces are currently, and where they are moving to.

With these, the following 6 equations can be used to solve for the 6 parameters. Let t_f be the time at which the EE reaches the final location (the duration), x_i and x_f be the initial and the final displacement values respectively.

$$\begin{aligned} x(0) = x_i \quad \dot{x}(0) = \dot{x_i} \quad \ddot{x}(0) = 0 \\ x(t_f) = x_f \quad \dot{x}(t_f) = \dot{x_f} \quad \ddot{x}(t_f) = 0 \end{aligned} \tag{2}$$

With this, all 6 parameters for the quintic polynomial in the x-direction can be solved. This process needs to be repeated for all three axes to form three different polynomials that describe the motion of the EE in three dimensions.

As proof of concept, consider interpolating the process of moving something from 1 to 0 in the x-direction over a 3 second duration. The initial and final

velocity and accelerations are all defined as 0. Giving the plots in figure 3. Of course, this only covers x-direction, the same process needs to be repeated for y and z directions to fully describe the translational trajectory.

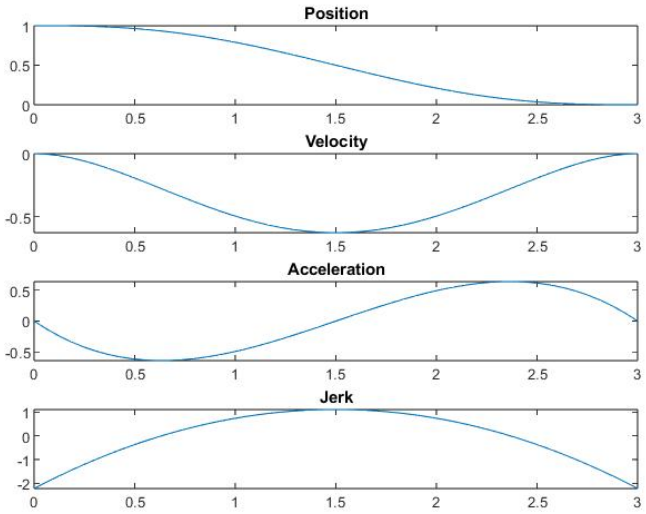


Figure 3: Trajectory interpolation example

3.1.2 Rotational Trajectory

The rotational aspect of the trajectory, from an initial orientation frame to a final orientation of a frame, can be defined around an axis $\hat{k}$, rotated by an angular displacement θ.

The rotation matrix that describes the relative rotation between the two frames, can be represented by ${}^{i}_{f}R$, where i represents the initial frame, and f represents the final frame. This can be obtained as indicated by Equation 3.

$$ {}^{i}_{f}R = {}^{0}_{i}R^{T}\,{}^{0}_{f}R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{3} $$

Let θ_f be the total amount of rotation. From here, the axis $\hat{k}$, and θ_f can be found with Equations 4 and 5.

$$\theta_f = 2\arccos\left(\frac{1}{2}\sqrt{1 + r_{11} + r_{22} + r_{33}}\right) \tag{4}$$

$$\hat{k} = \frac{1}{2\sin\theta_f}\begin{bmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{bmatrix} \tag{5}$$

Once the final value of theta has been obtained, it is then possible to interpolate a polynomial $\theta(t)$, representing the angular displacement as a function of time. The process is similar to section 3.1.1.

Initialising zero angular velocity at both the initial and final positions ensures smooth starting and stopping. The final angular displacement is given by θ_f, whereas the initial angular displacement θ_i can be calculated from the starting position of the chess piece, which should be known. Seeing as how there are 4 constraints, a cubic polynomial interpolation of $\theta(t)$ will be most suitable.

Now, the interpolated $(\theta(t), \hat{k})$ can be converted to Euler parameters that provide the reference orientation at a given time t. This process is outlined in Equation 6.

$$e_1 = k_x \sin\frac{\theta}{2}, \quad e_2 = k_y \sin\frac{\theta}{2}, \quad e_3 = k_z \sin\frac{\theta}{2}, \quad e_4 = \cos\frac{\theta}{2} \tag{6}$$

Finally, the relative rotation matrix between the initial EE frame to any time instance t (in terms of t) is given by the equation 7.

$${}^{i}_{t}R(t) = \begin{bmatrix} 1 - 2e_2^2 - 2e_3^2 & 2(e_1e_2 - e_3e_4) & 2(e_1e_3 + e_2e_4) \\ 2(e_1e_2 + e_3e_4) & 1 - 2e_1^2 - 2e_3^2 & 2(e_2e_3 - e_1e_4) \\ 2(e_1e_3 - e_2e_4) & 2(e_2e_3 + e_1e_4) & 1 - 2e_1^2 - 2e_2^2 \end{bmatrix} \tag{7}$$

and with respect to the initial coordinate frame 0:

$${}^{0}_{t}R = {}^{0}_{i}R\,{}^{i}_{t}R(t) \tag{8}$$

With this, we are able to know information pertaining to rotation of the end effector at any given time.

3.2 Design Reasoning

The trajectory from an arbitrary starting position to an arbitrary end position in task space might be a straight line. However, this trajectory might impede upon preexisting physical constraints. In the context of the required task, such constraints include joint space limitations, the robot's own workspace and the chess pieces being in the way.

To solve this problem, via points can be used. Via points are intermediary points on the trajectory that the robot must pass through en route to its final destination. This ensures that all the trajectory points do not violate the aforementioned physical constraints. The number of via points used can significantly affect the efficiency, smoothness and complexity of the required motion. For the sake of completeness, two cases will be considered and are listed below.

1. Case 1: One via point
2. Case 2: Two via points

3.2.1 Case 1: One via point

The generated trajectory is depicted by Figure 4 below. For the purposes of this explanation, let $h = 112.6$ and $\alpha = 3$cm, where h is height that the piece will be raised.

Consider a scenario where the chess piece will be moved on its line, two squares beyond itself, where a queen sits in the square between starting and end points, assuming that the starting and end points lie same line. During this move, the moving chess piece must not collide with the queen. The path between the starting position and point A is illustrated in Figure 5. As stated in Section 1, each piece has a radius of 1cm. That means for the edge of the moving piece to not collide with the queen, the base of the piece must clear 56.3mm before 10mm. However, as depicted in Figure 5, it makes an intersection with 56.3mm along the vertical axis at 15mm along the horizontal axis. The minimum height at which the chess piece is raised h, could be increased to avoid this specific intersection. However, when moving the chess piece further distance, the same problem will arise. Hence, this trajectory strategy is not suitable.

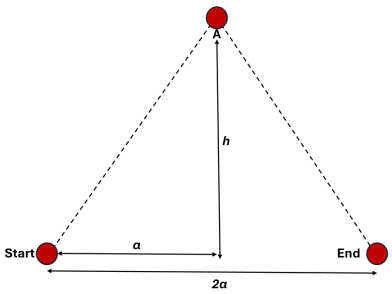


Figure 4: Case 1

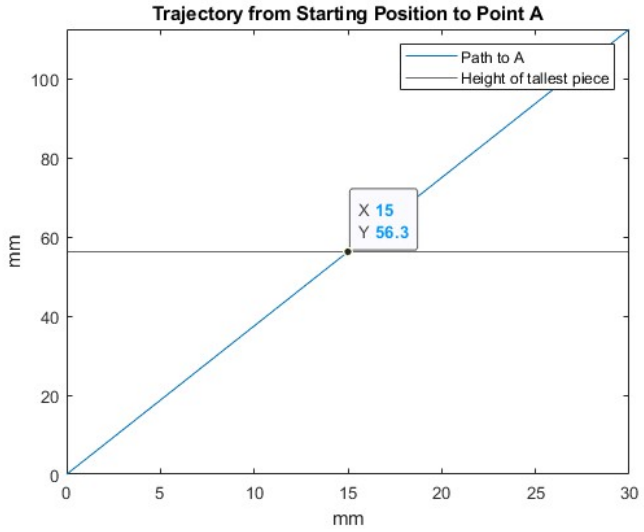


Figure 5: Trajectory from Starting Position to Point A

3.2.2 Case 2: Two via points

The generated trajectory is depicted by Figure 6 below. While this path is longer than the trajectory depicted in Figure 4, it is much safer as it guarantees no collisions. Therefore, this trajectory strategy is chosen.

As discussed in Section 3.1.1, quintic polynomials are used to design the translational trajectory of the robot. The movement strategy illustrated in

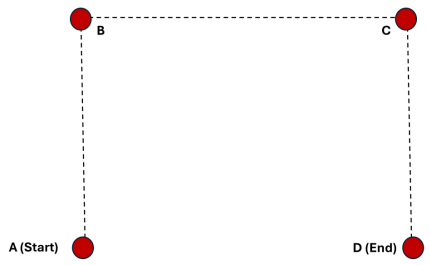


Figure 6: Case 2

Figure 6 has three segments and are as follows.

- A→B
- B→C
- C→D

The trajectory of each segment will have its own quintic polynomial. Each segment is defined below by Equation 9.

$$\begin{aligned} x_{AB}(t) &= a_5t^5 + a_4t^4 + a_3t^3 + a_2t^2 + a_1t + a_0 \\ x_{BC}(t) &= b_5t^5 + b_4t^4 + b_3t^3 + b_2t^2 + b_1t + b_0 \\ x_{CD}(t) &= c_5t^5 + c_4t^4 + c_3t^3 + c_2t^2 + c_1t + c_0 \end{aligned} \tag{9}$$

As mentioned in Section 3.1.1, each quintic polynomial has 6 coefficients that must be solved. Furthermore, since there are 3 segments, 18 coefficients are available. Moreover, there are 18 constraints and hence, 18 equations that must be solved. In addition, there should be continuity at the via point for the displacement, velocity and acceleration of each pair of adjacent segments. For the sake of brevity, only the constraints and subsequent equations for the segments AB and BC will be shown. The steps that follow can easily be repeated for BC and CD. The start and end positions, velocities and accelerations (at Point A and Point C) are defined below.

$$
\begin{aligned}
x_{AB}(0) &= x_{AB_0} \\
\dot{x}_{AB}(0) &= \dot{x}_{AB_0} \\
\ddot{x}_{AB}(0) &= \ddot{x}_{AB_0} \\
x_{BC}(t_{BC}) &= x_{BC_f} \\
\dot{x}_{BC}(t_{BC}) &= \dot{x}_{BC_f} \\
\ddot{x}_{BC}(t_{BC}) &= \ddot{x}_{BC_f}
\end{aligned}
\tag{10}
$$

The location of Point B (the point in between A and C) is illustrated below.

$$
\begin{aligned}
x_{AB}(t_{AB}) &= x_{AB_f} \\
x_{BC}(0) &= x_{BC_0}
\end{aligned}
\tag{11}
$$

Now there are 4 constraints left. The important constraint that must be specified is that the velocity and acceleration at the via point is continuous between the end velocity segment of AB and the starting velocity segment BC. The equations that yield a unique solution are depicted below, where V_{des} and A_{des} is specified velocity and acceleration at point B.

$$
\begin{aligned}
\dot{x}_{AB}(t_AB) &= V_{des} = \dot{x}_{BC}(0) \\
\ddot{x}_{AB}(t_AB) &= A_{des} = \ddot{x}_{BC}(0)
\end{aligned}
\tag{12}
$$

Specifying the velocity and acceleration yields a unique solution for the trajectory. Moreover, this approach ensures that the path through Point B can be uniquely determined and that the motion is smooth.

3.3 Trajectory Generation Algorithm

To generate the appropriate trajectory for the robot to complete a specific task, the coefficients of both the translational and rotational trajectories must be obtained. The following is the sequence of steps required to obtain the required trajectories.

1. Define the initial and final, positions, velocities and acceleration.
2. Define the final time to complete the movement.

3. Calculate the polynomial coefficients for both the translational and rotational trajectories.
4. Evaluate the polynomials that describe the linear and angular positions using the calculated coefficients.
5. Evaluate angular velocity by getting the derivative of the expression for angular position.
6. Derive linear velocity by deriving linear position.
7. Derive linear acceleration by deriving linear velocity.
8. Derive jerk by deriving linear acceleration.

4 Trajectory Visualisation

To exemplify the design reasoning in Section 3.2 for the use of two via points as well as the trajectory generation algorithm presented in Section 3.3: the trajectory of a simple chess move sequence is generated. In particular, a knight is to be moved from A5 to C6 while simultaneously avoiding a rook and a pawn in A6 and B6 (respectively). The trajectory is generated for the following sequence of movements.

1. Move the end-effector from the home position to be directly above the knight at A5.
2. Move vertically down to grab the knight.
3. Move vertically back up.
4. Move directly above the end position (C6).
5. Move down vertically to place the knight.
6. Move vertically back up.
7. Return to the home position.

Note that the maximum height in the z direction is chosen to be $z_{move\ height} = 0.12m$ in order to clear the top of the rook with the bottom of the picked up knight. Furthermore, the locations of the chessboard positions are defined as 0.5 increments of the square dimension away from the chessboard '0' position

- which is set to be on the intersection of columns D and E, and halfway up row 1. Thus, A5 is defined as $(x, y) = (4 \times 0.03 + 0.1, -3.5 \times 0.03)$ and C6 as $(x, y) = (5 \times 0.03 + 0.1, -1.5 \times 0.03)$. Finally, the constraints on starting and ending velocity and acceleration as outlined in Section 3.2.2 are all set to 0 (to allow the arm to rest smoothly at each via point before moving to the next point such that any oscillations generate throughout the move have subsided).

4.1 3D Trajectory Plot

Firstly, a plot of the reference x, y and z locations of the end-effector is displayed both spatially as a 3D trajectory plot, and separated as functions of time. The former is shown in Figure 7, where the vias are shown as the bold red dots. The latter is shown in Figure 8, where each colour represents a different segment of the trajectory with the via points highlighted in red.

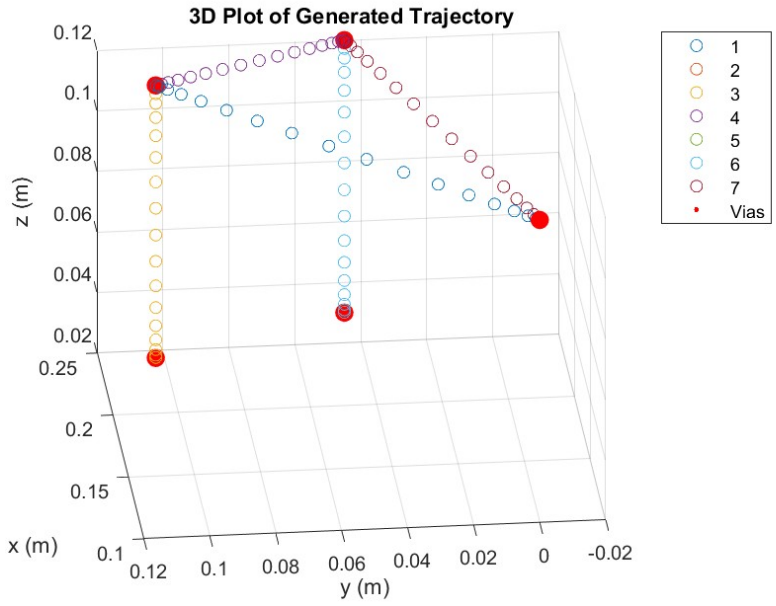


Figure 7: 3D Visualisation of Trajectory

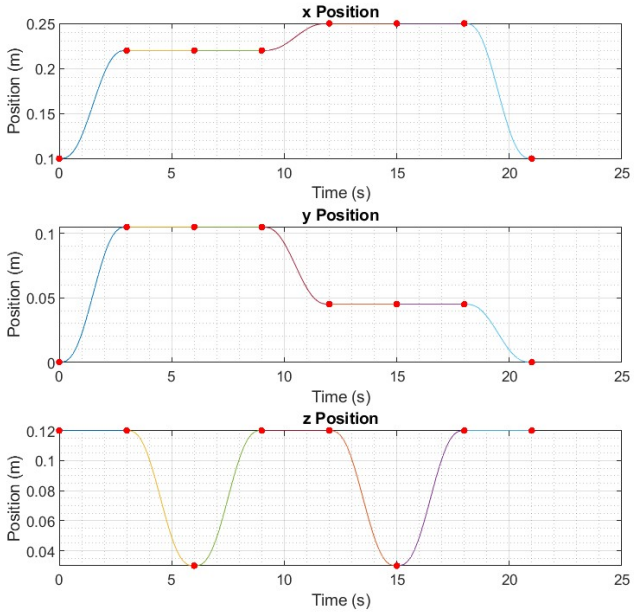


Figure 8: End-Effector Position as a Function of Time

4.2 XY Plane Trajectory

Next, the top-down projection of the trajectory given in Section 4.1 onto the XY plane is shown in Figure 9. As can be seen, the end-effector moves in straight lines between the home position (at $(x, y) = (0.1, 0)$) and the two chessboard positions from which the knight is picked up (at $(x, y) = (0.22, 0.105)$) and placed (at $(x, y) = (0.25, 0.045)$), which it does by moving vertically up and down (not seen in the figure). Again, the via points are highlighted in red.

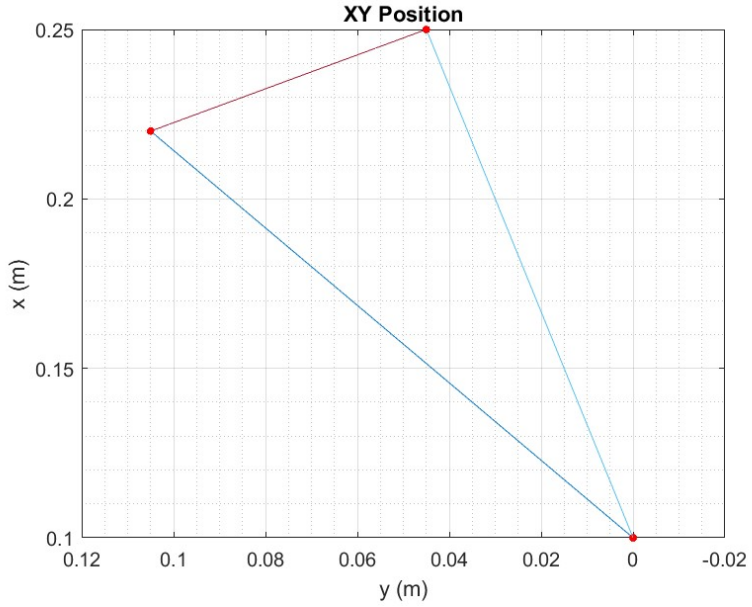


Figure 9: XY Projection of Trajectory

4.3 ZX Plane Trajectory

Finally, the side-on projection of the trajectory onto the ZX plane is shown in Figure 10. As can be seen, for all of the planar XY movement the end-effector is at a height of $z = 0.12m$ which is chosen to ensure that the bottom of a collected chess piece doesn't knock the top of another piece on the board. Then, at the specified x (as well as y, but this cannot be seen in the figure) position of the knight: the end-effector moves vertically downwards to a height of $z = 0.3m$ - which is the point above the chess board where the knight can be successfully grasped by the end-effector mechanism. Note that the perfectly vertical movement is to ensure that during the pick-up, no adjacent pieces are knocked by the end-effector. Again, the via points are highlighted in red.

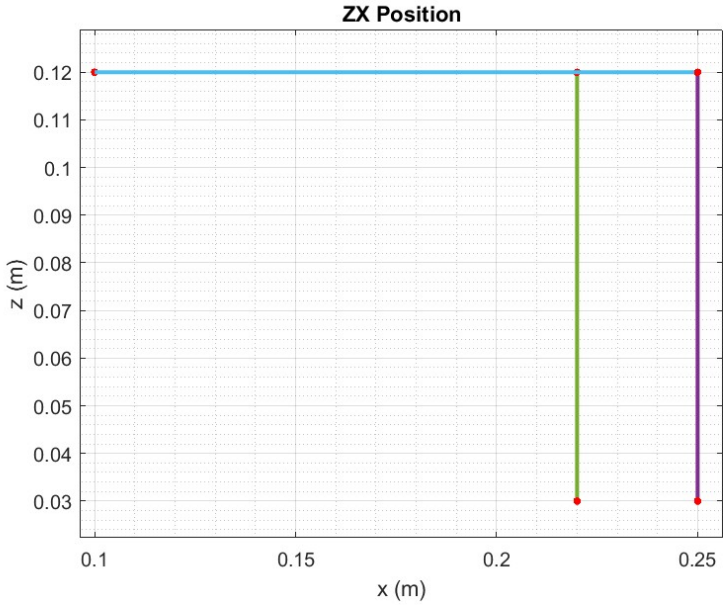


Figure 10: ZX Projection of Trajectory

5 Feasibility and Simulation

5.1 Robot Pose Simulation

To demonstrate the feasibility of the example trajectory presented in Section 4, a stick-figure representation of the robotic arm pose at various points along the trajectory can be superimposed onto Figures 7 and 10. This shows how the structure of the arm will need to be positioned to achieve the end-effector locations generated by the trajectory - which can be visually assessed to determine whether the required poses are feasible. These plots are shown in Figures 11 and 12, respectively. As can be seen, the robotic arm poses are feasible in all cases, whereby the joint angles are within acceptable ranges (i.e. always elbow-up, and configuration not near singularities), and the links do not intersect with each other (i.e. the arm can physically achieve the required pose).

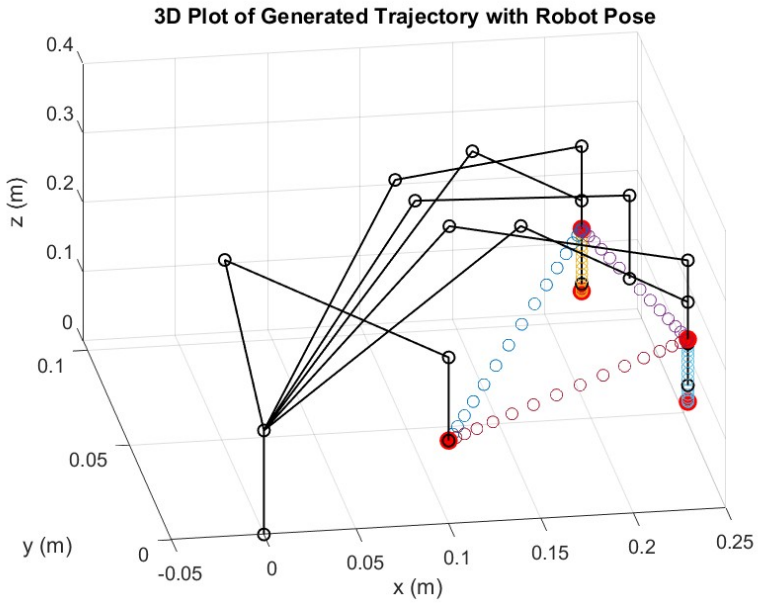


Figure 11: 3D Visualisation of Trajectory with Superimposed Arm Poses

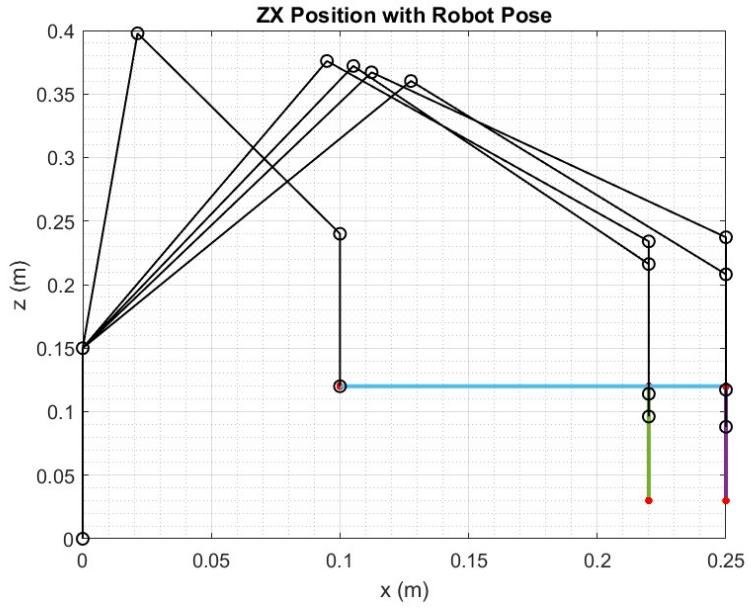


Figure 12: ZX Plane Projection of Trajectory with Superimposed Arm Poses