Lab 4: Velocity Kinematics

Zheyuan Xie, Zixuan Lan

October 31, 2018

1 Introduction

In this lab we derived the forward and inverse velocity kinematics for the Lynx robot, and designed a trajectory generation method for moving the end effector. We evaluated our method by conducting simulation experiment in MATLAB. We did not use a physical robot.

2 Pre-lab Tasks

Let's derive the forward velocity kinematics for the Lynx robot. The Lynx robot is a 5-link manipulator arm with joint variables q_1, q_2, q_3, q_4, q_5. Let

$$T_5^0(q) = \begin{bmatrix} R_5^0(q) & o_5^0(q) \\ 0 & 1 \end{bmatrix}$$

denote the transformation from the end-effector frame to the base frame, where $q = (q_1, \cdots, q_5)^T$ is the vector of the joint variables.

Let $S(\omega_5^0) = \dot{R}_5^0(R_5^0)^T$ define the angular velocity vector ω_e^0 of the end-effector, and let $v_e^0 = \dot{o}_e^0$ denote the linear velocity of the end effector. We seek expressions of the form

$$v_5^0 = J_v \dot{q}$$
$$\omega_5^0 = J_\omega \dot{q}$$

where J_v and J_ω are 3×5 matrices. The above two equations can be written together to yield the velocity kinematics for the Lynx robot:

$$\xi = J\dot{q} \tag{1}$$

in which ξ and J are given by

$$\xi = \begin{bmatrix} v_5^0 \\ \omega_5^0 \end{bmatrix} \quad \text{and} \quad J = \begin{bmatrix} J_v \\ J_\omega \end{bmatrix}$$

The end effector position o_5^q can be obtained from forward kinematics which is completed in Lab 1.

$$o_5^0 = \begin{bmatrix} c_1(s_2a_2 + c_{23}a_3 + c_{234}d_5) \\ s_1(s_2a_2 + c_{23}a_3 + c_{234}d_5) \\ c_2a_2 - s_{23}a_3 - s_{234}d_5 \end{bmatrix}$$

The linear velocity Jacobians J_v is calculated by doing the partial derivative of o_5^0 respect to five joint variables.

$$\begin{aligned} J_v &= \begin{bmatrix} \frac{\partial o_5^0}{\partial q_1} & \frac{\partial o_5^0}{\partial q_2} & \frac{\partial o_5^0}{\partial q_3} & \frac{\partial o_5^0}{\partial q_4} & \frac{\partial o_5^0}{\partial q_5} \end{bmatrix} \\ &= \begin{bmatrix} \frac{\partial x}{\partial q_1} & \frac{\partial x}{\partial q_2} & \frac{\partial x}{\partial q_3} & \frac{\partial x}{\partial q_4} & \frac{\partial x}{\partial q_5} \\ \frac{\partial y}{\partial q_1} & \frac{\partial y}{\partial q_2} & \frac{\partial y}{\partial q_3} & \frac{\partial y}{\partial q_4} & \frac{\partial y}{\partial q_5} \\ \frac{\partial z}{\partial q_1} & \frac{\partial z}{\partial q_2} & \frac{\partial z}{\partial q_3} & \frac{\partial z}{\partial q_4} & \frac{\partial z}{\partial q_5} \end{bmatrix} \\ &= \begin{bmatrix} -s_1(s_2a_2 + c_{23}a_3 + c_{234}d_5) & c_1(c_2a_2 - s_{23}a_3 - s_{234}d_5) & c_1(-s_{23}a_3 - s_{234}d_5) & -c_1s_{234}d_5 & 0 \\ c_1(s_2a_2 + c_{23}a_3 + c_{234}d_5) & s_1(c_2a_2 - s_{23}a_3 - s_{234}d_5) & s_1(-s_{23}a_3 - s_{234}d_5) & -s_1s_{234}d_5 & 0 \\ 0 & -s_2a_2 - c_{23}a_3 - c_{234}d_5 & -c_{23}a_3 - c_{234}d_5 & -c_{234}d_5 & 0 \end{bmatrix} \end{aligned}$$

The angular velocity Jacobians J_ω is given by horizontally stack the rotation axis of each joint. The calculation is done both by hand and by using the MALTAB symbolic toolbox to confirm its correctness.

$$\begin{aligned} J_\omega &= \begin{bmatrix} z_0 & z_1 & z_2 & z_3 & z_4 \end{bmatrix} \\ &= \begin{bmatrix} z_0 & R_1^0 z_0 & R_2^0 z_0 & R_3^0 z_0 & R_4^0 z_0 \end{bmatrix} \\ &= \begin{bmatrix} 0 & -s_1 & -s_1 & -s_1 & c_1 c_{234} \\ 0 & c_1 & c_1 & c_1 & s_1 c_{234} \\ 1 & 0 & 0 & 0 & -s_{234} \end{bmatrix} \end{aligned}$$

3 Method

3.1 Forward Velocity Kinematics

Joint velocity and end-effector velocity are related by a Jacobian matrix which we have already derived in the Pre-lab section. Now substitute J_v and J_ω into Equation 1, the full forward velocity kinematics of Lynx is written as

$$\begin{bmatrix} v_5^0 \\ \omega_5^0 \end{bmatrix} = \begin{bmatrix} -s_1(s_2a_2 + c_{23}a_3 + c_{234}d_5) & c_1(c_2a_2 - s_{23}a_3 - s_{234}d_5) & c_1(-s_{23}a_3 - s_{234}d_5) & -c_1 s_{234} d_5 & 0 \\ c_1(s_2a_2 + c_{23}a_3 + c_{234}d_5) & s_1(c_2a_2 - s_{23}a_3 - s_{234}d_5) & s_1(-s_{23}a_3 - s_{234}d_5) & -s_1 s_{234} d_5 & 0 \\ 0 & -s_2a_2 - c_{23}a_3 - c_{234}d_5 & -c_{23}a_3 - c_{234}d_5 & -c_{234}d_5 & 0 \\ 0 & -s_1 & -s_1 & -s_1 & c_1c_{234} \\ 0 & c_1 & c_1 & c_1 & s_1c_{234} \\ 1 & 0 & 0 & 0 & -s_{234} \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \\ q_5 \end{bmatrix}$$

3.2 Inverse Velocity Kinematics

The Jacobian relationship $\xi = J\dot{q}$ specifies the end-effector velocity that will result when the joints move with velocity $\dot{q}$. When $N = 6$, the Jacobian matrix is square. The velocity vector for a square Jacobian can be solved by using the equation:

$$\dot{q} = J^{-1}\xi$$

However, Lynx robot only has 5 Joints, which means the robot loses one rank.Therefore, we need to use the pseudoinverse of the jacobian matrix, which is denoted as J^+. By using J^+, we can then avoid the velocity singularities and get the least square solutions. The equations used are:

$$\dot{q} = (J^+)^{-1}\xi J^+ = (J^T J)^{-1} J^T$$

Since J is a 6x5 matrix, the resulting J^+ would then be a 5x6 matrix. As a result, $\dot{q}$ would be a 1x5 vector. This is reasonable since the robot lacks 1 DOF and can only control the velocities at 5 joints. To be consistent with other matrix dimensions, we assume the 6th element of $\dot{q}$ as 0 and make it a 1x6 vector. To get pseudoinverse matrix of J, we use MATLAB's symbolic tool to first get J. However, due to the long computation time to compute the inverse of $J^T J$, we substitute the variables ($\theta 1 \sim \theta 5$) and constants ($d1$,$a2$,$a3$,$d5$,lg) with numerical values before we execute the inversion.

3.3 Trajectory Generation for the End Effector

We should notice that the end effector is not able to achieve any desired orientation due to physical constraints of the robot. As we have derived in Lab2: Inverse Kinematics, if the end effector's orientation

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

must satisfy

$$\text{atan2}(y_c, x_c) = \text{atan2}(r_{23}, r_{13}) \tag{2}$$

if any solution exists, where x_c and y_c is the coordinate of the wrist center. A physical intuition of this limit is that the projection of z_e axis on the x_0y_0 plane must went through the origin O_0.

We use Euler angle to represent the orientation of the end-effector. Given the end effector frame at zero configuration:

1. First rotate a yaw angle θ about the x_e axis;
2. Second rotate a pitch angle ϕ about the new y_e axis;
3. Then rotate a roll angle ψ about the new z_e axis.

The rotation matrix R_e^0 can be constructed from the Euler angles

$$\begin{aligned} R &= R_e^0(q_0)R_x(\theta)R_y(\phi)R_z(\psi) \\ &= \begin{bmatrix} -c_\theta s_\phi c_\psi + s_\theta s_\psi & c_\theta s_\phi s_\psi + s_\theta c_\psi & c_\theta c_\phi \\ -c_\theta s_\psi - s_\theta s_\phi c_\psi & s_\theta s_\phi s_\psi - c_\theta c_\psi & c_\phi s_\theta \\ c_\phi c_\psi & -c\phi s_\psi & s_\phi \end{bmatrix} \end{aligned} \tag{3}$$

where

$$R_e^0(q_0) = \begin{bmatrix} 0 & 0 & 1 \\ 0 & -1 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$

is the rotation matrix of the end effector at zero configuration.

We want the position and orientation of the end effector to trace a given trajectory

$$T(t) = \begin{bmatrix} x(t) \\ y(t) \\ z(t) \\ \theta(t) \\ \phi(t) \\ \psi(t) \end{bmatrix} \quad \dot{T}(t) = \begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{z}(t) \\ \dot{\theta}(t) \\ \dot{\phi}(t) \\ \dot{\psi}(t) \end{bmatrix} \tag{4}$$

The constraint in Equation 2 will be $\theta = q_2 = \text{atan2}(y, x)$. Therefore

$$\theta(t) = \text{atan2}(y(t), x(t)) \tag{5}$$

$$\dot{\theta}(t) = \frac{-y(t)\dot{x}(t) + x(t)\dot{y}(t)}{x^2(t) + y^2(t)} \tag{6}$$

If we do not care about the orientation of the end effector during tracing a trajectory, we can set $\phi(t)$ and $\psi(t)$ to a constant or 0. If we want the robot to achieve maximum workspace, we can set

$$\phi(t) = \text{atan2}\left(y(t) - d_1, \sqrt{x^2(t) + y^2(t)}\right) \tag{7}$$

A physical intuition of Equation 7 is the extension line of z_e intersect with the center of joint 2. This will get the wrist center as close to the center of Joint 2 as possible.

The angular velocity ω is related to derivatives of the Euler angles by a matrix $\mathbf{W}$

$$\begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \mathbf{W} \begin{bmatrix} \dot{\theta} \\ \dot{\phi} \\ \dot{\psi} \end{bmatrix}, \quad \mathbf{W} = \begin{bmatrix} 0 & 0 & 1 \\ s_\theta & -c_\theta & 0 \\ c_\theta c_\phi & s_\theta c_\phi & s_\phi \end{bmatrix} \tag{8}$$

For simplicity, if we do not interested in the orientation of the end effector, we can only consider the situation when the end effector's z_e axis remains parallel to the ground. Therefore, $\omega_x = \omega_y = 0$ and $\omega_z = \dot{\psi} = (-y\dot{x} + x\dot{y})/(x^2 + y^2)$.

4 Evaluation

4.1 Forward & Inverse Velocity Kinematics

As stated before, Forward Velocity Kinematics and Inverse Velocity Kinematics are calculated based on the Jacobian matrix J or J^+. The first verification to run is to test whether the input of velocity FK and the output of velocity IK are consistent with each other. Here, we denote the input of velocity FK as q and $\dot{q_1}$. The velocity IK function then takes in the output, e_{vel}, from velocity FK. The output of the velocity output is denoted as $\dot{q_2}$. The object is to compare $\dot{q_1}$ and $\dot{q_2}$. We arbitrarily choose q within joint limit and $\dot{q_1}$. The values of $\dot{q_1}$ and $\dot{q_2}$ are expected to be the same in the absence of singularities; if singularities occur, the two values should still be close to each other. We run 3 trials to test the outputs of FK and IK. Below is a table showing all 3 trials.

Table 1: Comparision between Velocity FK and IK

	q	$\dot{q_1}$	$\dot{q_2}$
trial1	[1, 0, 0, 0, 1, 0]	[0.5, 1, 1.5, 2.5, 3, 0]	[0.5, 1, 1.5, 2.5, 3, 0]
trial2	[1, 1.1, 1.2, 1.3, 1.4, 1.5]	[0.5, 1, 1.5, 2.5, 3, 0]	[0.5, 1, 1.5, 2.5, 3, 0]
trial3	[1, 1.1, 1.2, 1.3, 1.4, 1.5]	[0.3, 1.2, 1.1, 0, 6.7 0]	[0.3, 1.2, 1.1, 0, 6.7 0]
error (%)	0	0	0

As shown in the table, the results from velocity FK and IK match with each other, regardless of how q changes. Moreover, the FK and IK functions are valid under different velocities.

4.2 Verification at Zero Position

Here we evaluate 5 cases, and in each case only one of the five joints moves. Figure ... shows the configurations of when only $joint_i$ moves, $i = 1, 2, 3, 4, 5$.

Table 2 shows the body velocity, ξ, when only one of the 5 joints move.

Table 2: ξ for each joint

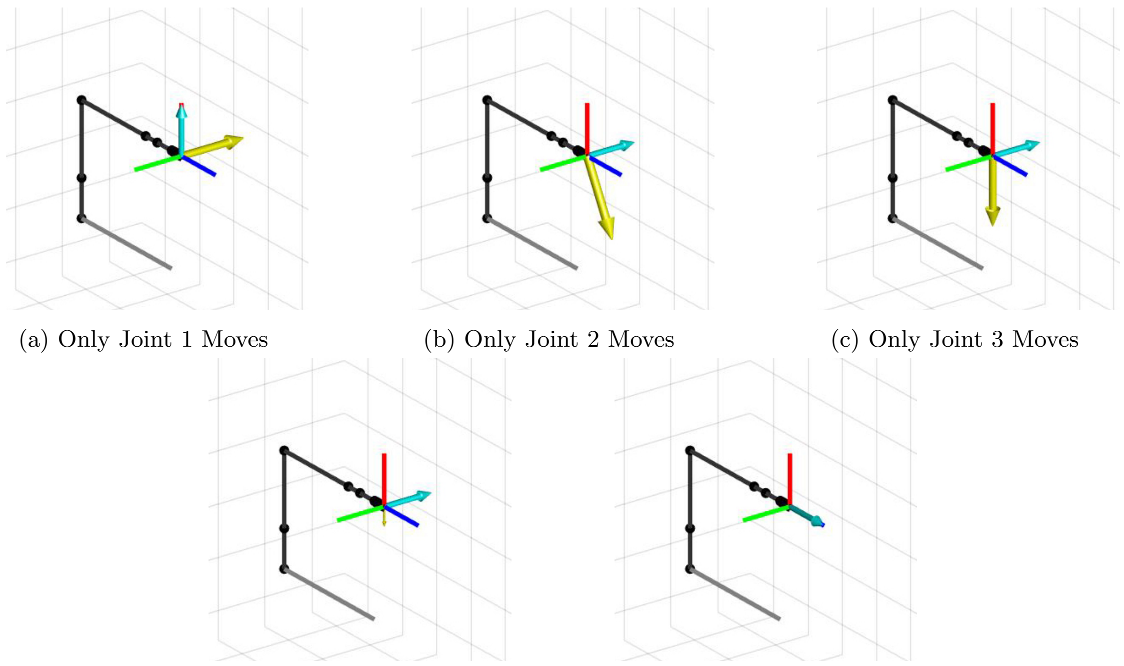
	$\dot{q_1}$	$\dot{q_2}$	$\dot{q_3}$	$\dot{q_4}$	$\dot{q_5}$
ξ	$\begin{bmatrix} 0 \\ \dot{q_1}(a3+d5) \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$	$\begin{bmatrix} a2\dot{q_2} \\ 0 \\ -\dot{q_2}(a3+d5) \\ 0 \\ \dot{q_2} \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0 \\ -\dot{q_3}(a3+d5) \\ 0 \\ \dot{q_3} \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0 \\ -\dot{q_4}(a3+d5) \\ 0 \\ \dot{q_4} \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0 \\ 0 \\ \dot{q_5} \\ 0 \\ 0 \end{bmatrix}$

1. Moves only Joint 1

   First, we assume only joint 1 moves; $\dot{q} = [\dot{q_1}, 0, 0, 0, 0, 0]$. From the geometric intuition, the expected motion should be the end effector moving along y direction and has an angular velocity along z, as this is consistent with the moving direction of joint 1 at 0 pose. Then we run the simulation.

   As shown in Fig 1a, when only joint 1 can move, the robot only has linear velocity along y direction and angular velocity along z. The resulting ξ from velocity FK is $[0, \dot{q_1}(a3+d5), 0, 0, 0, 1]$. As we can see, only the y component of linear velocity and z component of angular velocity have value; this is consistent with the simulation we have in Fig 1a .

   Similarly, we can analyze the body velocity vector for joint 2 - joint 5.


(a) Only Joint 1 Moves (b) Only Joint 2 Moves (c) Only Joint 3 Moves

(d) Only Joint 4 Moves (e) Only Joint 5 Moves

Figure 1: Linear and angular velocity of the end effector when only one joint moves. (The blue arrow indicates angular velocity and the yellow arrow indicates linear velocity.)

2. Moves only Joint 2

   In Fig 1b, the end effector's linear velocity vector is pointing in positive x and negative z direction. It also has an angular velocity pointing y.

   As shown in Table 2 , joint 2 has a linear velocity along x and z axis; it also has a angular velocity along y axis. This is consistent with the our observation basing on the figure.
3. Moves only Joint 3

   In Fig 1c, the end effector's linear velocity vector is pointing in negative z direction. It also has an angular velocity pointing y.

   As shown in Table 2 , joint 3 has a linear velocity along z axis; it also has a angular velocity along y axis. This is consistent with the our observation basing on the figure.
4. Moves only Joint 4

   In Fig 1d, the end effector's linear velocity vector is pointing in negative z direction. It also has an angular velocity along y.

   As shown in Table 2 , joint 4 has a linear velocity along z axis; it also has a angular velocity along y axis. This is consistent with the our observation basing on the figure.
5. Moves only Joint 5

   In Fig 1e, the end effector only has an angular velocity pointing y.

   As shown in Table 2 , joint 5 doesn't have any linear velocity along z axis; it only has a angular velocity along x axis. This is consistent with the our observation basing on the figure.

4.3 Singularities

Velocity IK doesn't have a solution when the Jacobian matrix J loses ranks, which means the robot has fewer DOF then 5. This normally happens when two or more rotations axis of the joints align with others. Specifically, one of the singularities occurs when joint 3, $q_3 = \pi/2$. As shown in Fig 2a, the rotation axis of joint 1 and joint 5 align with each other. Now the robot can only have 3DOF, including linear motion along x, rotation along y and z axis.

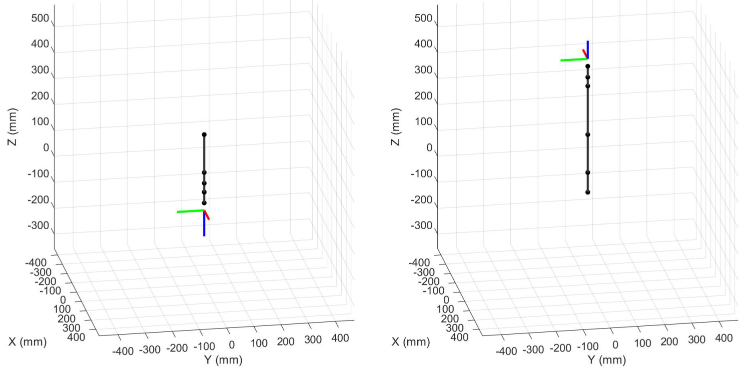
To check this geometric intuition, we plug in the angle position into Jacobian matrix, and we get:

$$J = \begin{bmatrix} 0 & -117.475 & -263.525 & -76.2 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & -1 \end{bmatrix}$$

Therefore, the rank of J is 3. This matches with the geometric analysis we have.

Another case is when $q_3 = -\pi/2$. Similarly, in Fig 2, the rotation axis of joint 1 and joint 5 align with each other. Now the robot can only have 3DOF, including linear motion along x, rotation along y and z axis. To check this geometric intuition, we plug in the angle position into Jacobian matrix, and we get:

$$J = \begin{bmatrix} 0 & 409.575 & 263.525 & 76.2 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$



(a) $q_3 = \pi/2$, $q_2 = q_4 = 0$ (b) $q_3 = -\pi/2$, $q_2 = q_4 = 0$

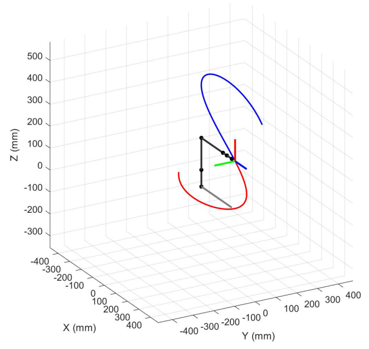
Figure 2: Singularity caused by Joint 3

4.4 Constant Velocities

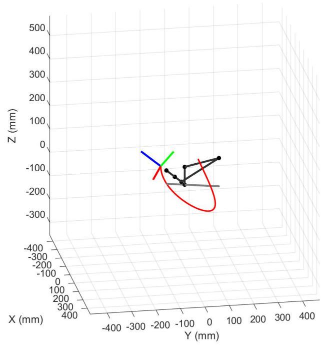
We set the joint velocity input so that each joint rotates in the same direction with a constant angular rate. We tested both positive angular rate and negative angular rate:

$$q_1 = \begin{bmatrix} 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 0 \end{bmatrix}$$
$$q_2 = \begin{bmatrix} -0.5 & -0.5 & -0.5 & -0.5 & -0.5 & 0 \end{bmatrix}$$

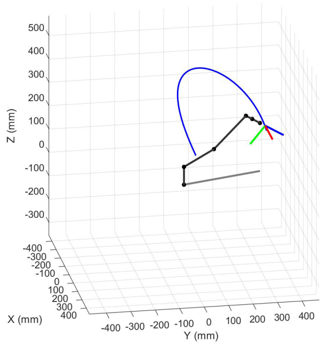
Figure 3(a) shows the trajectory the end effector traced before one of the joint hitting its limit. The red trace correspond to executing q_1 and the blue trace correspond to executing q_2. However, if we ignore the joint limits, the end effector will trace a loop as shown in Figure 4.



(a) Possible trajectories



(b) Positive Constant $\dot{q}$



(c) Negative Constant $\dot{q}$

Figure 3: End effector trajectory when all joints are moving at constant velocity.

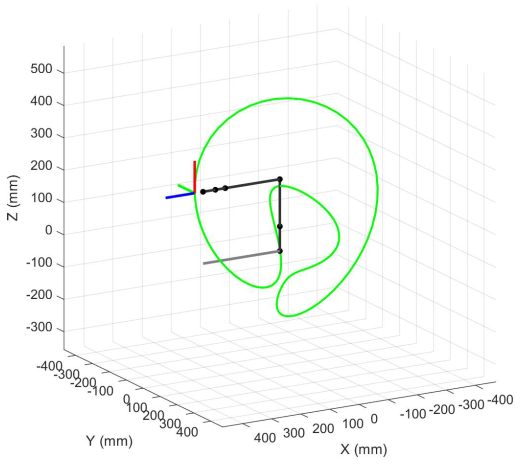
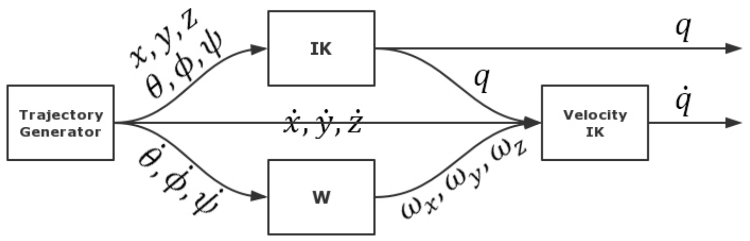


Figure 4: Caption

4.5 Trajectory Tracking

We further evaluate our method by controlling the end effector to track a desired trajectory in workspace. The trajectory is constructed using the method described in Section 3.3. We use Euler angle to represent the orientation of the end effector and impose the physical constraint on $\theta(x, y)$.

To obtain position and velocity reference in configuration space from a trajectory in workspace, we follow the process illustrated in Figure 5. First calculate the inverse kinematics to obtain joint angle $q(t)$, then using inverse velocity kinematics to obtain joint angular velocity $\dot{q}(t)$.


Trajectory Generator
IK
W
Velocity IK
x, y, z
θ, ϕ, ψ
$\dot{x}, \dot{y}, \dot{z}$
$\dot{\theta}, \dot{\phi}, \dot{\psi}$
q
q
$\dot{q}$
$\omega_x, \omega_y, \omega_z$
Figure 5: Trajectory tracking workflow.

We tested our method with a linear trajectory and a circular trajectory, with and without orientation control. You can view a video demo of our experiment here: `https://youtu.be/YnN1f3wJ2f4` in which the robot tracks the reference input $q(t)$ exactly.

1. Trace a Straight Line

   The trajectory is a diagonal line in the yz plane.

$$\begin{cases} x(t) = 300 \\ y(t) = -150 + 150t \\ z(t) = -100 + 200t \\ \phi(t) = 0 \\ \psi(t) = 0 \end{cases} \qquad \begin{cases} \dot{x}(t) = 0 \\ \dot{y}(t) = 150 \\ \dot{z}(t) = 200 \\ \dot{\phi}(t) = 0 \\ \dot{\psi}(t) = 0 \end{cases}$$

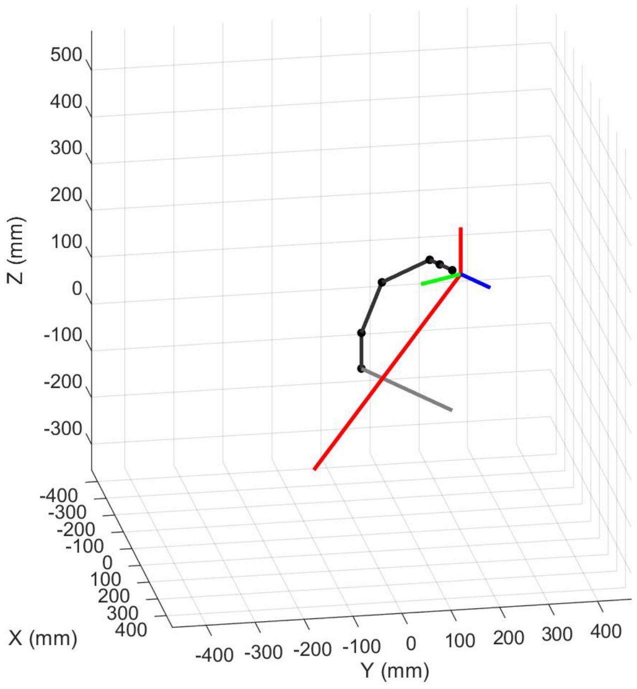
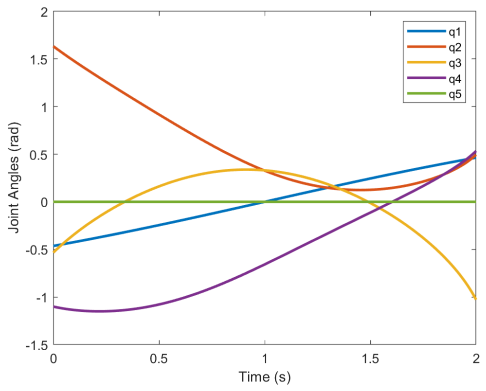
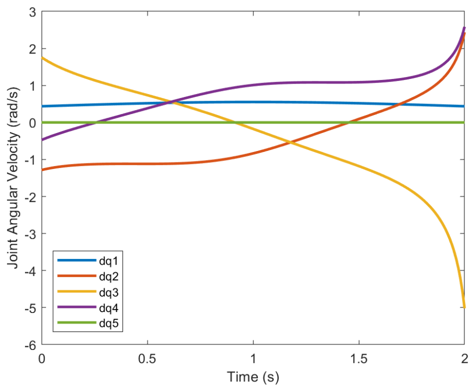


Figure 6: End effector trajectory when tracing a line.



(a) Joint angle plot



(b) Joint angular velocity plot

Figure 7: Result of inverse kinematics and inverse velocity kinematics when tracing a line.

2. Trace a Circle

   Similarly we can control the end effector to track a circle with radius 100mm in the yz plane.

$$\begin{cases} x(t) = 300 \\ y(t) = 100\sin 2t \\ z(t) = 100\cos 2t \\ \phi(t) = 0 \\ \psi(t) = 0 \end{cases} \qquad \begin{cases} \dot{x}(t) = 0 \\ \dot{y}(t) = 200\cos 2t \\ \dot{z}(t) = -200\sin 2t \\ \dot{\phi}(t) = 0 \\ \dot{\psi}(t) = 0 \end{cases}$$

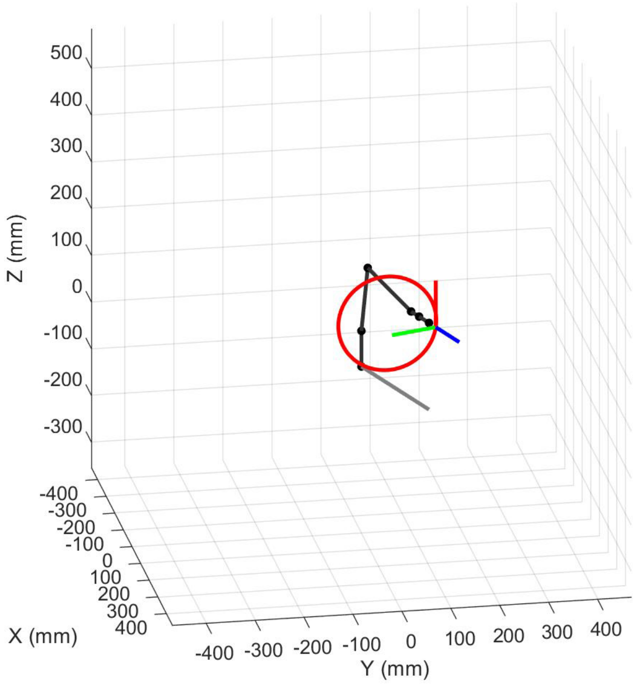
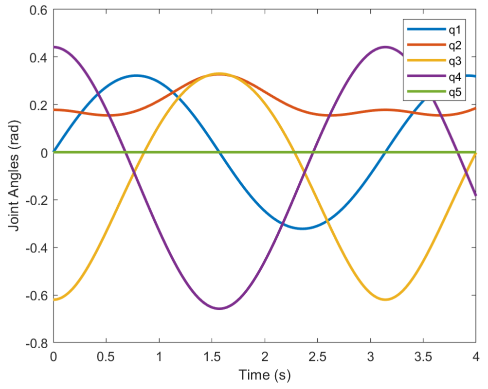
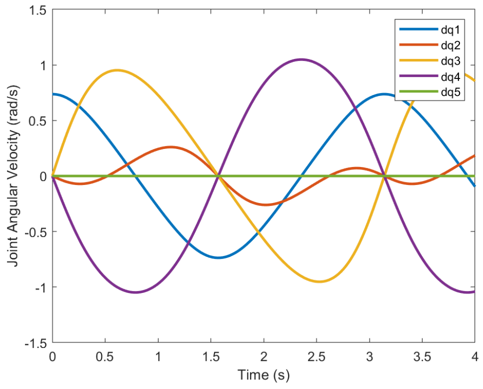


Figure 8: End effector trajectory when tracing a circle.



(a) Joint angle plot



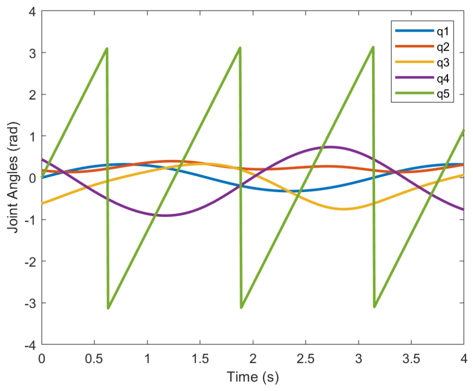
(b) Joint angular velocity plot

Figure 9: Result of inverse kinematics and inverse velocity kinematics when tracing a circle.

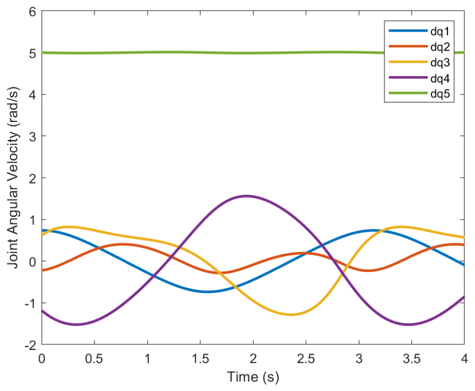
3. Control the orientation

   Now we make the Euler angles ϕ and ψ time-variant as well. Note we need to convert $[\dot{\theta}, \dot{\phi}, \dot{\psi}]^T$ to $[\omega_x, \omega_y, \omega_z]^T$ by left multiply $\mathbf{W}$ in the Equation 8 before calculating the inverse velocity kinematics.

$$\begin{cases} x(t) = 300 \\ y(t) = 100 \sin 2t \\ z(t) = 100 \cos 2t \\ \phi(t) = \frac{\pi}{8} \sin 2t \\ \psi(t) = 5t \end{cases} \qquad \begin{cases} \dot{x}(t) = 0 \\ \dot{y}(t) = 200 \cos 2t \\ \dot{z}(t) = -200 \sin 2t \\ \dot{\phi}(t) = \frac{\pi}{4} \cos 2t \\ \dot{\psi}(t) = 5 \end{cases}$$



(a) Joint angle plot



(b) Joint angular velocity plot

Figure 10: Circle tracing with orientation control.

5 Analysis

In conclusion, our velocity FK and IK solutions make sense, because

1. when joint velocities are picked arbitrarily, both FK and IK solutions are consistent with each other. As shown in Section 4.1, we run three trials, and the input of velocity FK matches with the output of velocity IK.
2. the geometric approach and analytical approach (Jacobian) to solve the velocity FK at 0 position match. In Section 4.2, Figure 1 show that the end effector has certain velocity components when one joint moves from 0 position. Table 2's results justify the simulations in Figure 1.
3. both the Jacobian matrix and the simulation reflect the singularities. In Section 4.3, the results from rank test of Jacobian matrix agree with that of simulation.

Besides the validity of the velocity FK and IK solutions, we also find that velocity control is good at path smoothing. For example, if we use position control to trace a circle without specifying the initial and end velocity of the end effector at the way points, the end effector would receive impulse signal to move. In another word, the end effector would accelerate for a very short period of time, reach the way point, and decelerate to 0. As a result, the end effector is easily overshooting the way point and the movement seem not smooth. However, if we use velocity control to implement this, we can assign a constant joint velocity, so the movement of the end effector is very smooth, and the overshoot problem is reduced. Moreover, if we find overshooting occurs for the velocity control, we can simply reduce the value of constant velocity.

Another advantage velocity control has is its ability to avoid singularities. For robotic arm, singularity occurs at certain configurations, and it's more difficult for us to predict the singularity by using position control. In comparison, if we use velocity control, we can simply check the rank of Jacobian matrix. Moreover, to avoid singularities, we can use pseudoinverse Jacobian, regardless of whether the rank of J is higher or lower than 6 DOF. As discussed in method and evaluation section, pseudoinverse Jacobian always guarantees a solution to velocity IK and it gives us a least square error solution.

On the other hand, velocity control is bad at real-time planning or planning for very complicated path. Velocity control definitely adds to the complexity of computing. In velocity IK, the inverse/pseudoinverse of Jacobian matrix could be difficult to compute when the Jacobian matrix grows very complicated.

Therefore, we would prefer to use velocity control if we want smooth path and want an accurate open loop control of end effector position. What's more, when we want the end effector to move in constant velocity, velocity control is definitely easier. We also prefer to use velocity control when we know the end effector is likely to hit its singularities during a path.