Project 1

NYU Tandon Schoool of Engineering

Nov 16, 2021

Student: Naveen Kumar University ID: N19068326, Net ID: nk3090
Instructor: Professor Ludovic Dominique Righetti
Course: Foundations of Robotics ROB-GY 6003

NYU

Contents

1. Introduction

In the project, we have been given 5 questions.

1. In question 1, we are supposed to write five functions namely vec_to_skew(w), twist_to_skew(V), exp_twist_bracket(V), inverseT(T) and getAdjoint(T).

   1.1. vec_to_skew(w) will convert a 3-dimensional array into a skew-symmetric matrix.

   1.2. twist_to_skew(V) will convert a 6-dimensional twist or screw array containing the first three elements as angular velocities in x, y, and z directions respectively, and the last three elements as linear velocities in x, y, and z directions respectively into a 4 by 4 matrix containing first 3 rows and 3 columns as a skew-symmetric matrix of angular elements of 6D array converted by function vec_to_skew(w), the elements of first 3 rows and last column contain the last 3 elements of 6d array and all the elements of the last row of this 4 by 4 matrix is zero.

   1.3. exp_twist_bracket(V) will return the exponent of the 6D array by first converting it to a 4 by 4 matrix using the function twist_to_skew(v). After that, it will take its exponent. Which returns a transformation matrix

   1.4. inverseT(T) will give the inverse of a transformation matrix.

   1.5. getAdjoint(T) will give the 6 by 6 Adjoint of a 4 by 4 matrix. In this 6 by 6 matrix, the first 3 rows, and 3 columns is the rotation matrix extracted from the first 3 rows and 3 columns from the 4 by 4 input matrix. The elements in the first 3 rows and last 3 columns will be zero matrices. The elements in the last three rows and first three rows will be the multiplication of skew-symmetric matrix of first three rows and last column of 4 by 4 matrix and rotation matrix, i.e. first three rows and three columns of the 4 by 4 input matrix. The elements of the last three rows and last three columns are the rotation matrix.

2. In question 2, we are supposed to perform forward kinematics by using joint angles of all the seven joints of the arm.
3. In question 3, we are supposed to find the Space Jacobian using the screw axes of each joint.
4. In question 4, we are supposed to plot the trajectories of the path of end-effector in xy position, yz position and xz position. We have to use forward_kinematics(V) function derived in Question 2 to figure out the position of end effector in X, Y and Z axes.

5. In question 5, we are supposed to plot the linear velocities of end-effector in Spatial frame, end-effector frame and end-effector frame orientation as similar as the spatial frame.

2. Working

2.1 Question 1

1. In vec_to_skew(w) function, 3 elements are given to the function, which converts into the skew-symmetric matrix like below.

$$\begin{bmatrix} 0 & -w(3) & w(2) \\ w(3) & 0 & -w(1) \\ -w(2) & w(1) & 0 \end{bmatrix}$$

The working of other functions is mentioned in the Introduction section of the report.

2.2 Question 2

In question, we are supposed to find the forward kinematics. In order to do this question, we have to find the screw (6 D Vector) of all 7 joints, as you can see in the jupyter notebook. In order to find the screw axes, I am declaring a 6 by 1 matrix as a zero matrix, First 3 elements can be found by looking at the diagram and checking the rotational axes of the joint with respect to the spatial frame. The last 3 elements can be found by cross multiplying the first 3 elements of 6D array with the position vector of the joint with respect to the spatial frame. We have to find a total of 7 screw vectors of all the joints. Now, we have to write an “M” matrix which is nothing but the transformation from spatial frame to end-effector frame at the initial position.
We now need the bracketed form of each screw which can be done by exp_twist_bracket(V) function.

$$T(\theta) = e^{[S_1]\theta_1} \cdots e^{[S_{n-1}]\theta_{n-1}} e^{[S_n]\theta_n} M$$

The above formula is used to calculate the forward kinematics. We have to multiply bracketed screw to the corresponding joint angle value and then we have to multiply the whole term to M to get the transformation between end-effector and spatial frame.

2.3 Question 3

In question 3, We have to find the space Jacobian ($J_s\theta$)

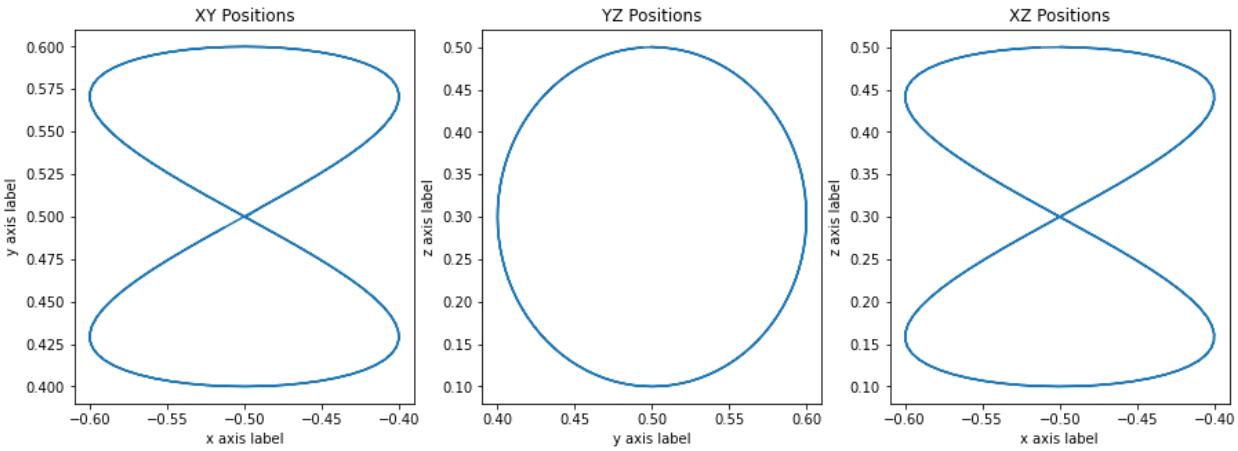
$$J_s\theta = [J_{s1} J_{s2}(\theta) \ldots J_{sn}(\theta)]$$

$$J_{s1} = S_1$$
$$J_{s2}(\theta) = Ad_{e^{[s1]\theta 1}} S_2$$
$$J_{sn}(\theta) = Ad_{e^{[s1]\theta 1} \ldots e^{[s(n-1)]\theta(n-1)}} S_n$$

For the first column of the space, Jacobian is the screw for joint 1, for the second column we have to multiply the screw of the second joint with the adjoint of the exponential of joint 1 and angle 1. Similarly, for others, we have to multiply the exponential of other screw axes and take their adjoint and multiply the term with its screw axes. You can see the program in which Js is calculated for each joint and arranged as a Jacobian matrix and we finally get a space jacobian.

2.4 Question 4

We have been provided with 200 configurations of 7 joint angles in the joint_treajectory.npy file. We have to store these values in a variable, in the program, it is stored in the joint_trajectory variable. Now, we have to iterate for loop 200 times, at each iteration we have to calculate the forward kinematics for each configuration. Then we have to store the values of x, y, and z position of end-effector from the last column of the transformation matrix. Now, we have to make three plots according to the question using matplotlib. We have to make xy, yz, and xz plots from the values obtained by calculations. When we look at the plots of XY, XZ, there is a similarity in the curve, and YZ curve is in an oval shape. As you can see in the diagrams below.



2.5 Question 5

In question 5, we are supposed to find out the velocities of the end-effector in the spatial frame, end-effector frame and the end-effector frame oriented as the spatial frame. In order to find the velocities in the Spatial frame, we need the space Jacobean and joint velocities. We have been provided with a file called joint_velocities.npy in which contains 200 instances for joint velocities of all 7 joints present. In order to solve this problem, we have to use for loop which iterates 200 times. At each iteration, it calculates the angular and linear velocities of the end effector by multiplying space jacobian with the joint velocities. The function to calculate space jacobian will be obtained from question 3. The space jacobian will be calculated from joint_trajectory variable containing values of the joint angle at a particular instance.
In the program, First, space Jacobian is calculated from get_space_jacobian function providing the joint angle values from joint_trajectory variable in for loop.
Then at a particular instant, the spatial twist is calculated as you can see in the program(Vels variable). From the spatial twist, the last three elements are the linear velocities in the spatial frame in x, y, and z directions respectively.
In the next step, we are storing these values for the plot.
Now, we have to calculate the linear velocities in the end-effector frame.
In order to that we have to multiply the body jacobian with joint velocities.

Body jacobian can be calculated by taking adjoint of the inverse of the transformation matrix by multiplying it with Space Jacobian. The formula of doing so is mentioned in the Modern Robotics book (5.22)

After calculating the body jacobian, Body jacobian will be multiplied by the angular velocities in order to calculate the linear velocities in the end-effector frame.

Now, we have to calculate the velocities in the frame at the end-effector but oriented as a spatial frame. From the situation, it is clear that the rotation between both the frame end-effector and the spatial frame is an identity matrix. We have to replace this Identity matrix in the matrix obtained by the forward_kinematics function.
After replacing the rotation matrix part of the transformation matrix, we have to take the inverse of that matrix. After taking the inverse, we have to take the Adjoint of that matrix and multiply it with space jacobian to get the Jacobian for this situation.
Now, we have to multiply this jacobian with the joint velocities, so we can get the linear velocities in the end-effector frame oriented as a spatial frame.
These velocities are stored in the respective variables so that we can plot them.
In the next pages, the plots of these velocities are shown.

I am answering the third part of question 5 here in this report.
When we look at the plots we will find that in spatial frame plots of x and z are following the same zigzag pattern. In plots at position 0 plots starts going up and stop at some position then both of the plots go down till 25 and then come up to achieve peak at 50 position. Then again the curve goes down and achieves a down value of 75 and goes up. When we compare the x and z plots with y plot we will see that, the points at which x and z achieve upward peak and come down, y achieve downward peak and goes up. We observe that, the pattern of the body frame velocities are opposite to the zig-zag pattern of spatial velocities.
When we look at the linear velocities in end-effector frame with orientation as same as spatial frame in X, Y and Z we will see that we are observing a sine wave like curve. And this frame seems more **intuitive** to me. Because these velocities are more regular, the peak values are approximately similar.

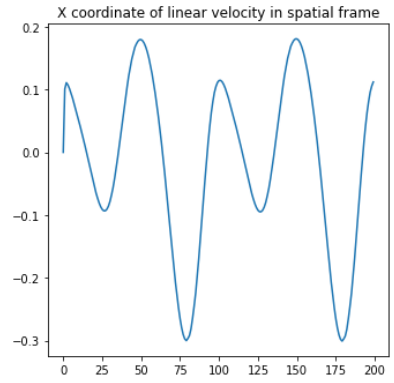
When we look at the YZ position curve which is like an ellipse. When we look at the equation of the ellipse.

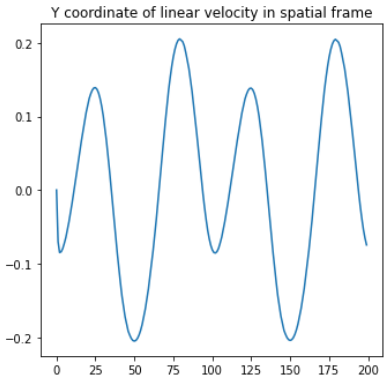
$$\frac{(y-h)^2}{b^2} + \frac{(z-k)^2}{a^2} = 1$$

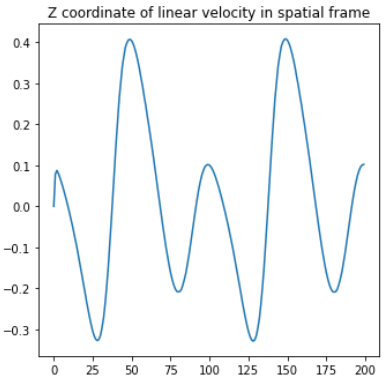
When we differentiate this equation with respect to time, we will find that

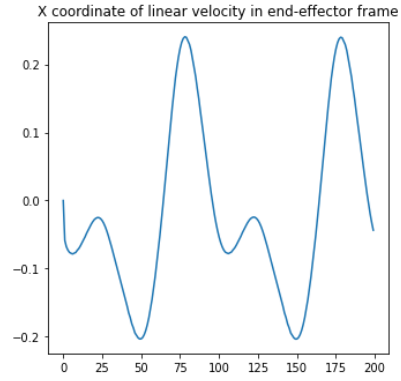
$$\frac{(y-h)}{b^2}\frac{dy}{dt} = -\frac{(z-k)}{a^2}\frac{dz}{dt}$$

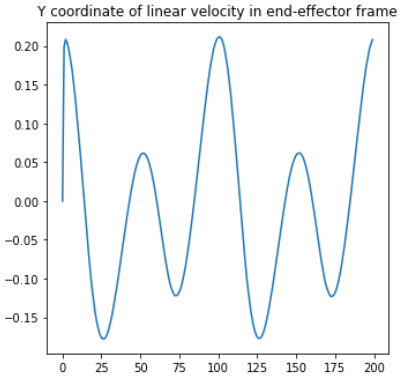
When we see the equation above we will find that the velocities in the Y and Z direction are opposite to each other and the same is happening with the spatial velocity plots.

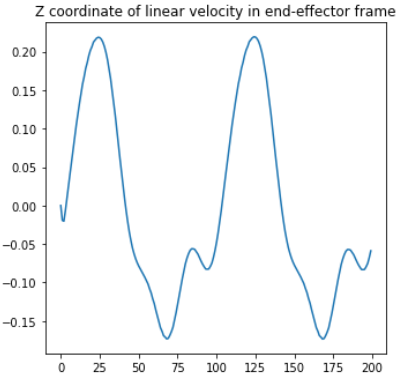

X coordinate of linear velocity in spatial frame

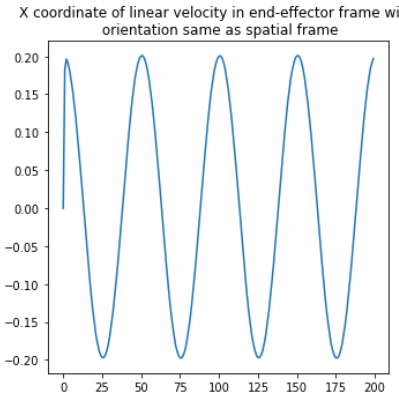

Y coordinate of linear velocity in spatial frame

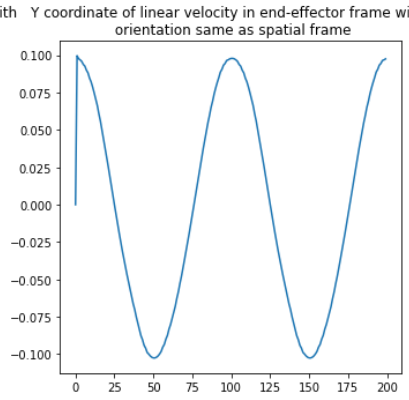

Z coordinate of linear velocity in spatial frame

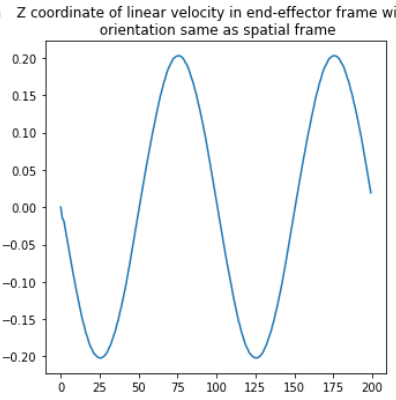

X coordinate of linear velocity in end-effector frame

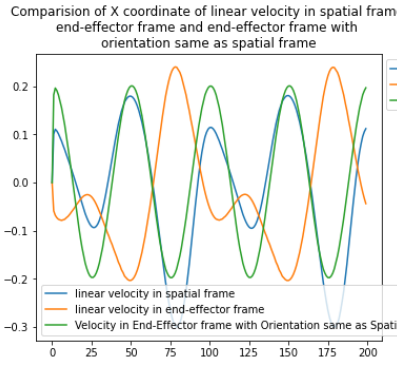

Y coordinate of linear velocity in end-effector frame

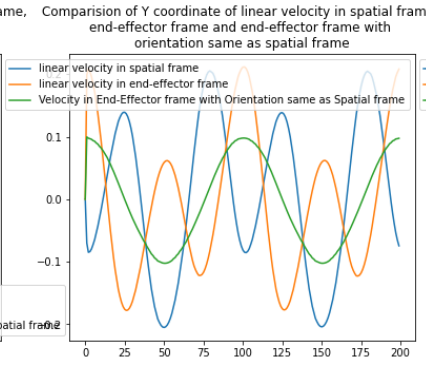

Z coordinate of linear velocity in end-effector frame

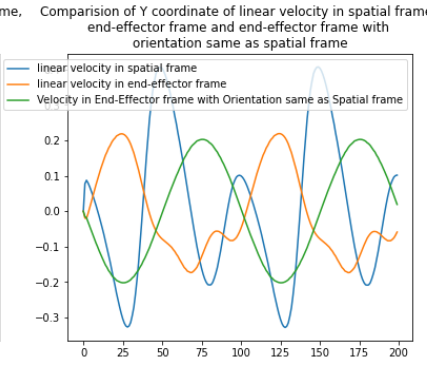

X coordinate of linear velocity in end-effector frame with orientation same as spatial frame


Y coordinate of linear velocity in end-effector frame with orientation same as spatial frame


Z coordinate of linear velocity in end-effector frame with orientation same as spatial frame


Comparision of X coordinate of linear velocity in spatial frame, end-effector frame and end-effector frame with orientation same as spatial frame
linear velocity in spatial frame
linear velocity in end-effector frame
Velocity in End-Effector frame with Orientation same as Spatial frame


Comparision of Y coordinate of linear velocity in spatial frame, end-effector frame and end-effector frame with orientation same as spatial frame
linear velocity in spatial frame
linear velocity in end-effector frame
Velocity in End-Effector frame with Orientation same as Spatial frame


Comparision of Y coordinate of linear velocity in spatial frame, end-effector frame and end-effector frame with orientation same as spatial frame
linear velocity in spatial frame
linear velocity in end-effector frame
Velocity in End-Effector frame with Orientation same as Spatial frame