CS422 Robotics and Automation, Programming Assignment 2

Maynooth University, Siyuan Zhan PhD

Due 10th Dec. Late submissions will not be accepted[*]

This entire assignment is written and consists of two significantly adapted problems from the textbook, Robot Modeling and Control by Spong, Hutchinson, and Vidyasagar (SHV). Please follow the extra clarifications and instructions on both questions.

Forward Kinematics of the PUMA 260 (20 pts)

1. **DH Function in MATLAB (5 pts)**
   Open the file *dh_starter.m*. Change the file name and the function definition (on the first line of the file) to your MU student number (1910xxxx) rather than the word starter. Update this file so that it calculates the four-by-four transformation matrix A for the set of DH parameters passed in by the user: a, alpha, d, and theta. The input angles have units of degrees, so use appropriate trigonometric functions. Test your function by calling it from the command line with various inputs before moving on to the next step.

2. **Animating the PUMA 260 (15 pts):**
   Open the file *starter.m* and put it in the same folder as the DH function you just wrote. Change the filename to your MU student number (1910xxxx) rather than the word starter, and enter your name as the value of the variable *student_name*.

   Update this script to animate the movement of the PUMA 260, using the origins of your seven frames (frame 0 to frame 6) as the points to be plotted. Ignore joint limits. All of your code should be between the two lines of asterisks. The first step is to calculate the robot's six A matrices (A1 through A6) given the current values of the joint variables; you should use the DH function you created in the previous step to do this. Note that the robot's measurements are already defined for you as **a** through **f**.

   Then calculate the position of the origin of each frame (o1 through o6) in the base frame; o0 is done for you. All seven origin locations should be put together in the variable *points_to_plot* in order, with the origin of frame 6 as the last column. Use the seven available motion modes to test your calculations, and fix everything that you notice is not correct. You are welcome to add more motion modes to test other trajectories as needed, but please don't modify the ones that are already defined. Instructions for turning in your MATLAB files appear at the end of this document.

[*]Some of the assignment materials were adapted from the open materials used in MEAM520 at the University of Pennsylvania, created by Dr Katherine J. Kuchenbecker

Inverse Kinematics (20 pts)

1. **Making the SCARA Draw a Vertical Circle (20 pts)**
Read Example 3.10 on pages 108–109 of the textbook. It derives the inverse kinematics of a SCARA manipulator with the goal of placing the end-effector at $\begin{bmatrix} o_x & o_y & o_z \end{bmatrix}^T$. The provided solution includes formulas for θ_2, θ_1, and d_3 (as well as θ_4, which we will be ignoring), but unfortunately it includes two mistakes. The corrected answers are as follows:

$$\cos\theta_2 = \frac{o_x^2 + o_y^2 - a_1^2 - a_2^2}{2a_1a_2} \tag{1}$$

$$\theta_2 = \text{ATAN2}\left(\frac{\pm\sqrt{1-\cos^2\theta_2}}{\cos\theta_2}\right) \tag{2}$$

$$\theta_1 = \text{ATAN2}\left(\frac{o_y}{o_x}\right) - \text{ATAN2}\left(\frac{a_2\sin\theta_2}{a_1 + a_2\cos\theta_2}\right) \tag{3}$$

$$d_3 = -o_z \tag{4}$$

Open the script called *scara_robot_circle_starter.m.* This script sets up an environment to animate the movement of a SCARA robot drawing a vertical circle over time. Rename the starter file to *scara_robot_circle_1910xxxx.m*, and put your name at the top of the file where it says student name = 'PUT YOUR NAME HERE';

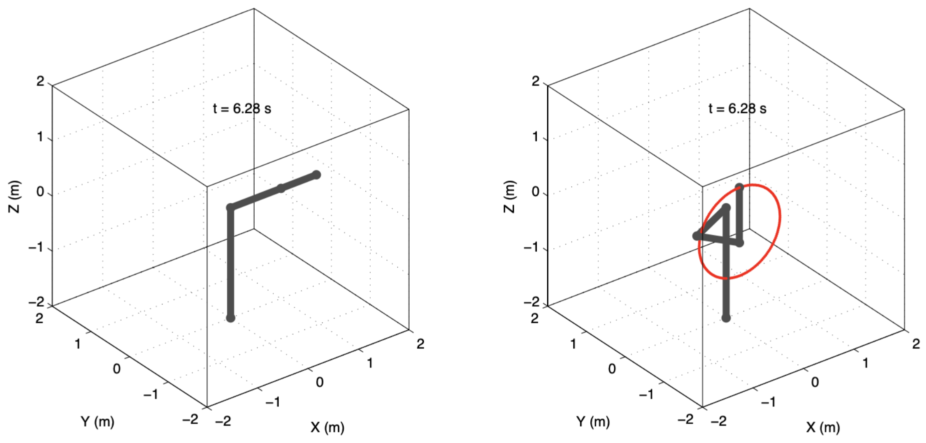

Figure 1: Left: SCARA Robot starting position. Right: SCARA Robot drawing a circle

Put the file *scara_robot_fk.p* in the same folder. As provided, the script should show a SCARA robot sitting still with all of its joint coordinates equal to zero, as shown below left.

Your job is to update this script so that the SCARA draws a circle in a plane parallel to the x-z plane, as shown above right. The desired *radius, y_offset*, *x_center*, and *z_center* are defined in the code (feel free to change these), along with the desired trajectory of the robot's tip. The provided starter code functions much as the code for Homework 2 did, except it provides you with ox, oy, and oz instead of the joint coordinates. You need to calculate *theta1* (in radians), *theta2* (in radians), and *d3* (in meters) from *a1*, *a2*, *ox*, *oy*, and *oz*. You should only need to update the code between the two lines of stars. Note that these angles are in radians, while the ones in the PUMA 260 problem are in degrees. Once you get this to work, spend some time playing with the circle parameters to improve your understanding of inverse kinematics. Submit your .m files.