Milestone: Welding Robot (Wire-Loop Game)

Robot Kinematics and Dynamics

Prof. Howie Choset

Contents

1 Overview

In your capstone project, you are asked to adapt the 2D Robot.m class you wrote before to accommodate a 3D robot.

- Start by creating a copy of Robot.m file and name the file "Robot3D.m"
- Use a matrix of DH parameters as the argument for the class constructor, rather than link lengths. You do not need to worry about mass information, but if you end up adding gravity compensation torques to improve performance later you will also want the center of mass and actual mass for each link and joint as well as for the end effector. For this milestone assignment, you should have DH parameters as the ONLY argument.
- Your Robot class should be able to adapt to a general n-link robot. (You can use the information from the DH parameters to initialize other variables like degrees of freedom.
- Modify the forward_kinematics method of the Robot class to return the frames, given the joint angles as argument.
- Modify the end_effector method of the Robot class to return the end effector description (x, y, z, roll, pitch, yaw), given the joint angles as argument
- Modify the jacobian method to calculate the Jacobian for the manipulator arm

2 Variables

You can try and find the symbolic DH parameter for this robot. Here, you are provided with DH parameters that comes with the user manual. We have also provided you with a test file named validate_robot_class.m in the milestone_handout folder. Copy your Robot3D.m file inside this milestone_handout folder. Assume that the robot is weightless and only forces acting on the robot are at the end effector.

The sample_ground_truth.mat file provided to you contains the following variables:

- dhp: DH parameter for shown 6 dof arm
- theta: joint angle trajectory with dimension as (6), where 6 is the number of joints and n is the number of points on trajectory
- pose: this variable contains the truth value for the end effector position for the given joint angles. Each column represents (x, y, z, roll, pitch, yaw)
- torques: joint torques trajectory with dimension as $6 \times n$, where 6 is the number of joints and n is the number of points on trajectory
- forces: this variable contains the truth value for the end effector forces (forces applied on the end effector) at the given joint angles. Each column represents (Fx, Fy, Fz, Mx, My, Mz).

3 Validation

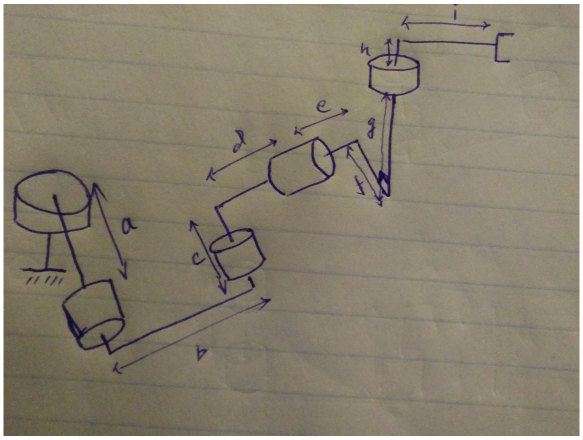
Update the validate_robot_class.m file to:

- Create an instance of robot class using DH parameter
- Calculate the end effector pose (x, y, z, roll, pitch, yaw)
- Calculate the forces applied on the end effector (Fx, Fy, Fz, Mx, My, Mz) for each joint angle.

Run the validate_robot_class function without passing any arguments. You will get two plots comparing your values of end effector position and forces with ground truth.

4 Application to capstone robot

Once you have verified that your robot3d class works on the sample robot, we want you test out all the functions on the robot in the REL. The schematic and geometry of the robot is shown below:



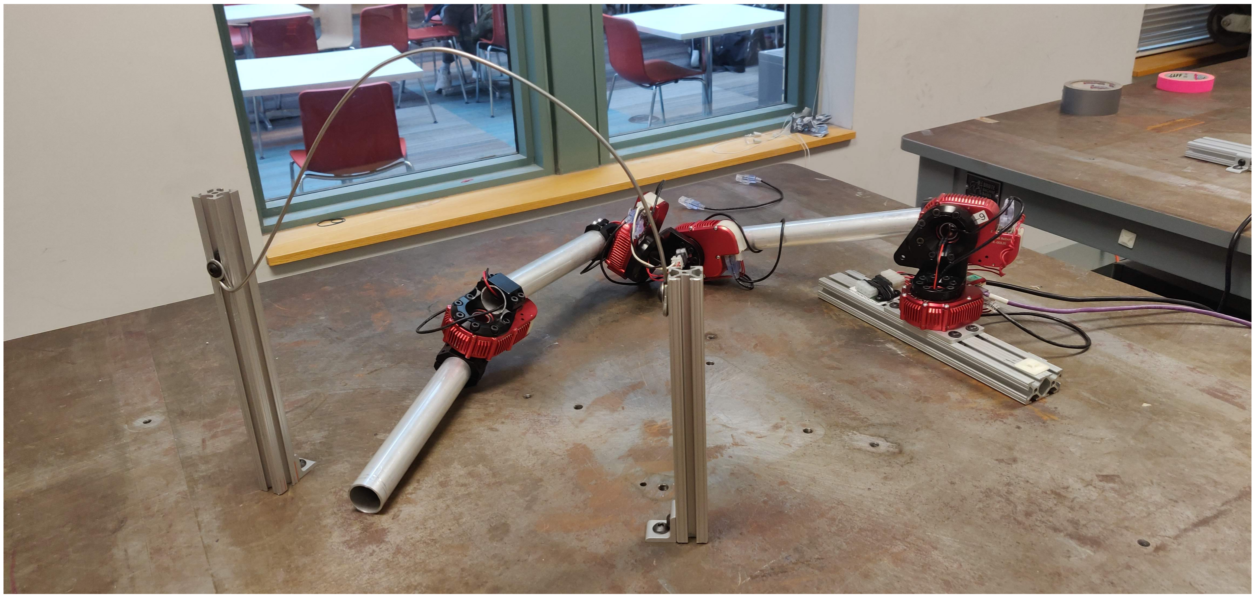
Parameter	Value (mm)
a	53.34
b	317.5
c	53.34
d	73.66
e	15.24
f	254
g	50.8
h	15.24
i	241.3

Table 1: System parameters for Welding Robot

Find the dh parameters and compute the jacobian of the robot. We recommend you measure the link lengths again to make sure your kinematics are correct. (Note: watch piazza for changes in the forward kinematics.)

5 Submission Checklist

- ☐ Coding Questions
  - ☐ Run create_submission.m in Matlab
  - ☐ Upload handin-6.tar.gz to Canvas
  - ☐ Upload a video of the robot executing the path constructed with steel wire in the workspace of the robot. You can reuse the previously written teach and repeat code to accomplish this task. A picture of the setup is shown below.



Note: the main goal for the last checklist item is to get started on the capstone early. You will be graded only on the video of the robot following the path. It is up to you to figure out how you want to do that. Eventually, we will only be providing the trajectory of 'welding' in x,y, and z space. You will have to compute the orientations yourself.