Lab 2 Manual

Robotics & Automation
Dylan Losey, Virginia Tech

1 Introduction

Welcome to the second robotics lab! The GTAs are here to facilitate your learning and help with any technical challenges. Please be kind, professional, and respectful to the GTAs.

Objective. This lab has two objectives:

- Finding the orientation of the robot's end-effector
- Using rotation matrices and axis-angle representations

Figure 1: Geomagic Touch at your workstation.

2 Calibrating the Robot

Multiple groups use each workstation. Between sessions the robot needs to be re-calibrated. You must complete these steps at the start of *every* lab. On the plus side, you practiced these same steps during Lab 1.

Action 1

Perform the following checks:

- The computer licences are paired with specific Geomagic Touch robots. Confirm that your computer has the same number as your Geomagic Touch workstation.
- Make sure that the ethernet cable coming out of the Geomagic Touch is connected to the back of the computer, and the robot's charging cable is plugged in.

- Check that the computer is connected to the wifi "quanser_UVS" with password "UVS_wifi".

Hint. The computers are slow. Patience is a virtue. But if your connection to the robot is continually lagging throughout this lab, we recommend that you restart the computer.

Action 2

Now we are ready to initialize our robot arm. Open the Geomagic Touch Diagnostic Tool application shown in Figure 2. Then complete the following action items:

Figure 2: Application to check your connection to the Geomagic Touch.

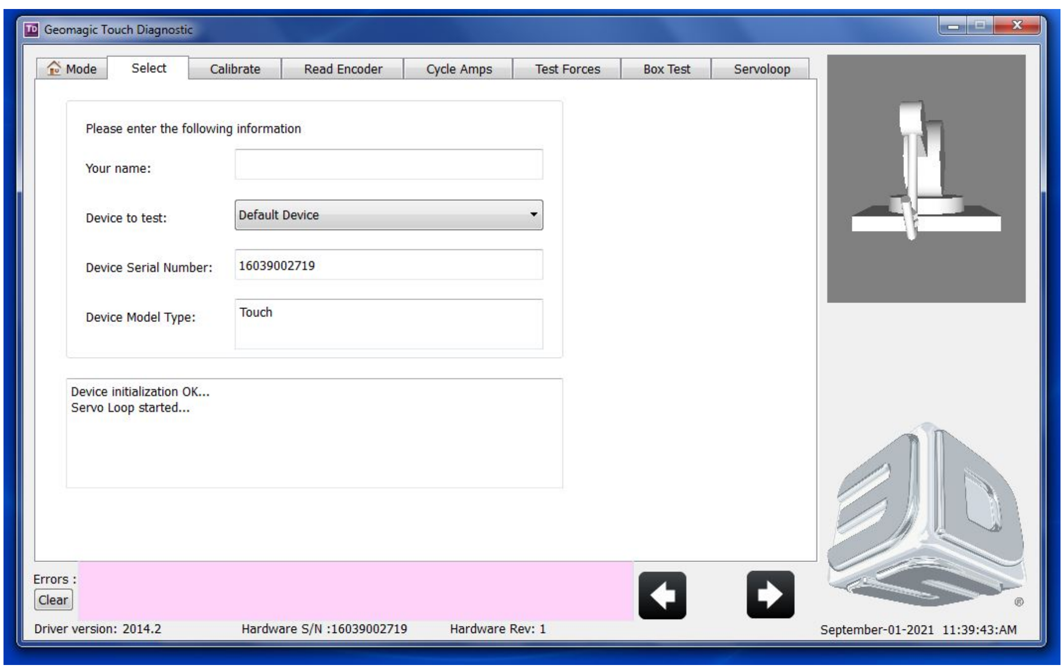

Figure 3: Geomagic Touch Diagnostic Tool. There is a 3D rendering of your robot arm in the top right. This should move in sync with your robot.

- When you open the application, you will see a tab at the top called **Select**. Click on this tab to see a small 3D model of your robot (shown in Figure 3). When you move the actual robot this 3D model should move as well. *If the 3D model is not moving when you move the robot ask a GTA for help.*

- Once you've ensured that the robot is connected, click on the **Calibrate** tab. Follow all the instructions on the screen to calibrate the robot (i.e., plug the pen into the holder for several seconds until the calibrate button turns green).
- Make sure to **close** the Diagnostic Tool after calibration. Otherwise it will conflict with Simulink in later steps.

Action 3

Let's open and run the initial Simulink model for this lab.

- In MATLAB, open the Rotations Simulink model:
  `..\Desktop\02 Rotations\Rotations_2015b_Start.mdl`
- You should see the Simulink diagram shown in Figure 4.

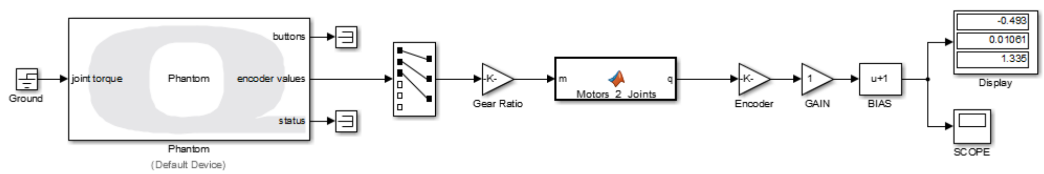

Figure 4: Initial Simulink diagram for the lab. This should look familiar — you developed this model during Lab 1.

- Build the simulation by going to the QUARC menu and clicking on **Build**. When the MATLAB Command Window shows that the model has been downloaded to the target, run the simulation by opening the QUARC menu and clicking on **Start**.
- Place the robot's end-effector on **position 2** of the baseboard. The positions on the **scope** for each revolute joint *in this lab* should read:

$$\theta_1 = 0.0 \quad \theta_2 = -0.3 \quad \theta_3 = 0.26 \tag{1}$$

  If your positions do not match, modify the **bias** block until the robot is correctly calibrated. Note: this won't be perfect, but try to get as close as possible. You might find that θ_3 is offset by $-\pi$ radians as compared to Lab 1.

3 Rotation Matrices

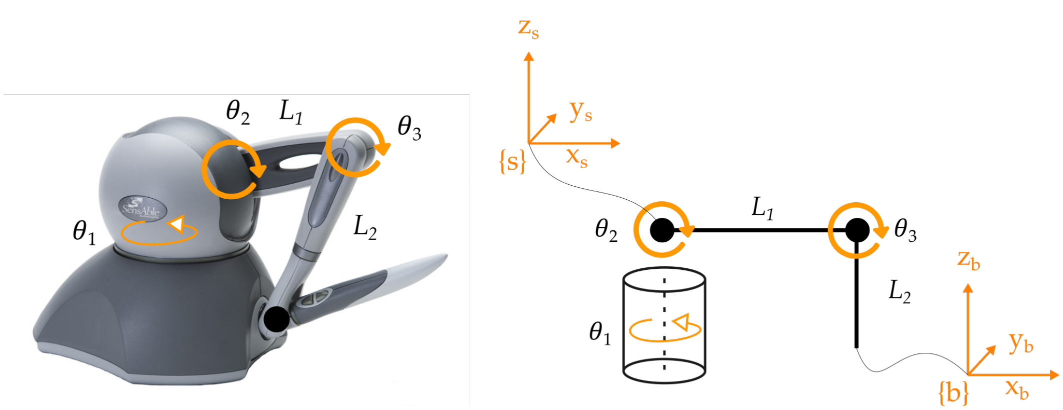

Figure 5: Joints of the Geomagic Touch. On right the robot is shown in its home position where $\theta_1 = \theta_2 = \theta_3 = 0$. Here $\{s\}$ is the fixed frame and $\{b\}$ is a coordinate frame at the robot's end-effector. Notice that $\{s\}$ and $\{b\}$ align in this home position.

Question 1

Given the home position of the robot shown in Figure 5, write an equation for R_{sb} as a function of θ_1, θ_2, and θ_3. Your function should provide the orientation of the robot's moving end-effector $\{b\}$ in the robot's fixed frame $\{s\}$.

Action 4

Implement your function in the Simulink model so that you can read R_{sb} in real time. How you do this is up to you — but your output must be a **scope** (or **display**) that shows R_{sb} as the robot moves. Here is one suggestion:

- In the Simulink model from Figure 4, insert a **MATLAB Function** block
- Write in a function that applies your answer to Question 1
- Connect the input of this block to θ, and connect the output to a **scope** or **display**

Question 2

If link 2 is in the home position from Figure 5, what is R_{sb}? Answer this question by moving the robot around while keeping link 2 downwards. Your answer should be a function of θ.

Question 3

Show your **display** for the rotation matrix R_{sb} to a GTA. As a team, demonstrate that R_{sb} approaches the identity matrix I when you move the robot to the home configuration from Figure 5.

4 Axis-Angle Representation

Action 5

Your Simulink model currently outputs the orientation of the robot's end-effector. What is the corresponding axis-angle representation? Update your Simulink model so that you **display** the three-dimensional axis and scalar angle in real time.

Question 4

While running the axis-angle Simulink model you designed, carefully move the robot as far as you can (*without pressing or pulling the robot too hard*). What is the maximum angle you can reach when rotating purely around z_s? What is the minimum angle when rotating around z_s? Separately looking at rotation about the x_s, y_s, and z_s axes, estimate the minimum and maximum rotations in radians.

Question 5

Imagine that someone needs to remotely control the robot over Zoom. If their perspective is not aligned with the robot (e.g., perhaps the camera is viewing the robot from an angle), what could you do to improve the user experience? Within your team, brainstorm two ways that you could align their perspective so that it is intuitive to remotely control the robot's orientation. Write down your solutions, and list any information you would need to implement those solutions.