Lab 1: Forward Kinematics and Planar Inverse Kinematics

Pre Lab Due Date: 09/16/2021 @ 11:59pm

Code + Report Due Date: 09/28/2021 @ 11:59pm

This assignment consists of both a written Prelab (due roughly one week from assignment release) and the main portion of the lab (code and report). You will submit all parts of the assignment through Gradescope. Late submissions will be accepted per the standard late submission policy, but they will be penalized by 25% for each partial or full day late. After the late deadline, no further assignments may be submitted; post a private message on Piazza to request an extension if you need one due to a special situation such as illness. The Prelab is worth 5 points, and the report and code are each worth 20 points. You may talk with other students about this assignment, ask the teaching team questions, use a calculator and other tools, and consult outside sources such as the Internet. **If you make use of outside sources, you are expected to cite them in your report.** When you get stuck, post a question on Piazza or go to office hours!

Note: this (like all future labs) is a partner assignment! You will complete and submit all Prelabs individually, but submitting the main lab individually is not allowed.

Please fill out **this form** to either submit the name of your partner (only one partner needs to submit) or ask to be matched up. This form is due the same day as the Prelab, so you are ready to hit the ground running on the main lab.

1 Prelab

The goal of the prelab is to connect ideas you are learning in class with the actual tasks you will work on in the projects. These assignments are "pen and paper" assignments in that you should not need to write any code, but if you would like to use digital tools to help explain your answers or improve your own understanding of the problems you are welcome to do so. In the last few classes, we have studied rotations, coordinate transformations, and homogeneous transforms. These tools will be critical the rest of the semester and are repeatedly used in every type of robotics. It is important that you are comfortable with these ideas.

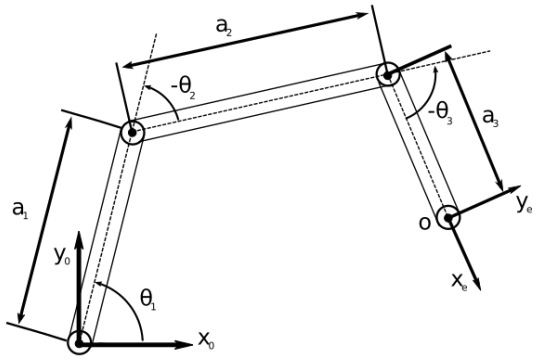

Figure 1: This is a planar robot arm with three revolute joints. Link lengths are specified by the variable a_i, and joint angles are represented by θ_i. The base frame is Frame $o_o x_o y_o$ and the end effector has Frame $o_e x_e y_e$. Note that some of the angles are labeled as negative because their value is negative in the configuration in which the arm is drawn, i.e., all angles have a positive sense of counterclockwise rotation in the plane.

We are going to consider the planar robot arm shown in Figure 1.

1. Draw the dexterous workspace for the robot arm.
2. Draw the reachable workspace for the robot arm.
3. Using what we learned in class, derive the forward kinematics for the robot arm which will result in coordinate frames for each joint and a transformation matrix that describes the end effector pose in base frame coordinates. Note that you can use any convention you desire, but we want you to use the base frame and end effector frame we have specified in Figure 1.

4. Write 3 - 5 sentences about the process you used to get the transformation matrix which can describe the end effector in base frame coordinates.

5. Select few example configurations and using your transformation compute the end effector pose in base frame coordinates. Do your results make sense? Write down one of your configurations and the resulting end effector pose, along with a 1 sentence reason for why this is the right answer.

   (**Hint**: Pick configurations that make it easy for you to check, e.g when $\theta = [0, 0, 0]$.)

6. Now that you have considered this simplified arm, we would like you to think about the Panda Robot arm you used in Lab0 and that we will continue to use the rest of the semester. Draw out a schematic of the Panda arm using the joint notation introduced in lecture. Write down 2 differences you observe between the Panda arm and the robot arm shown in Figure 1.

Submit the prelab on Gradescope (Due 09/16/2021 @ 11:59pm). You must submit your prelab assignment individually, and your writeup should be clear and concise.

2 Simulation Setup

To update your own private fork of the `meam520_labs` repo for Lab1 please do the following:

```
$ cd ~/meam520_ws/src/meam520_labs/
$ git pull upstream master
```

This is going to look at the public TA repository and grab the code stubs and launch files that we've added since the last lab. Occasionally when running this command, you might encounter a merge conflict. We don't expect that to occur, but if it does, don't hesitate to reach out to the TAs for help.

The algorithms you implement in labs will be stored in the `lib` folder:

```
$ cd ~/meam520_ws/src/meam520_labs/lib
```

since you are building a library of functionality which can be reused in future labs.

3 Forward Kinematics

In this part of the lab, you will implement the forward kinematics of the Panda robot arm. Your task is to write a function which takes in the joint angles of the robot, and spits out the pose of the end effector frame and the position of each joint, all relative to the world frame.

3.1 Definitions, Dimensions, and Conventions

In Figure 2, you can see a schematic of the Panda arm in a given configuration, including dimensional information. Additionally, you can see the assignment we have chosen for the end effector frame and the world frame. The end effector frame follows the convention in your textbook, with the origin centered between the gripper fingers, the z axis pointing in the *approach direction* and the y axis pointing along the *sliding direction* of the gripper. The world frame lies on the axis of the first joint, on the bottom of the robot where it mounts to its surroundings.

These are the only coordinate frames that we have adopted as a standardized convention for the entire class. As you compute the forward kinematics, you will need to assign intermediate coordinate frames to the various links of the robot. How you assign these coordinate frames is left completely up to you, and will not affect your results as long as you correctly compute the relative transformations between each pair of frames.

Likewise, you are also asked to compute the position of each joint. The notion of the position of a joint is not unique, in the sense that any point lying along the joint's axis of rotation could be considered the position of that joint, since it is fixed relative to both the *proximal link* (closer to base) and *distal link* (closer to gripper) of that joint. However, we have adopted canonical positions for the joints which are unambiguous due to the mechanical design of the robot. Namely, for each joint there exists a clear "cut plane" orthogonal to the joint's axis which separates the two links (as far as can be seen on the outer surface of the robot). These positions are shown clearly in Figure 2, and will be considered the canonical joint positions for all of MEAM520.

Finally, it's worth mentioning that there are several popular conventions for assigning intermediate frames to the links of the robot, such as the Denavit-Hartenberg (DH) convention. It should be noted that in general, **the origins of the intermediate frames under some chosen frame convention (e.g., DH) are not necessarily the same as the joint positions, even if this occurs some of the time!**

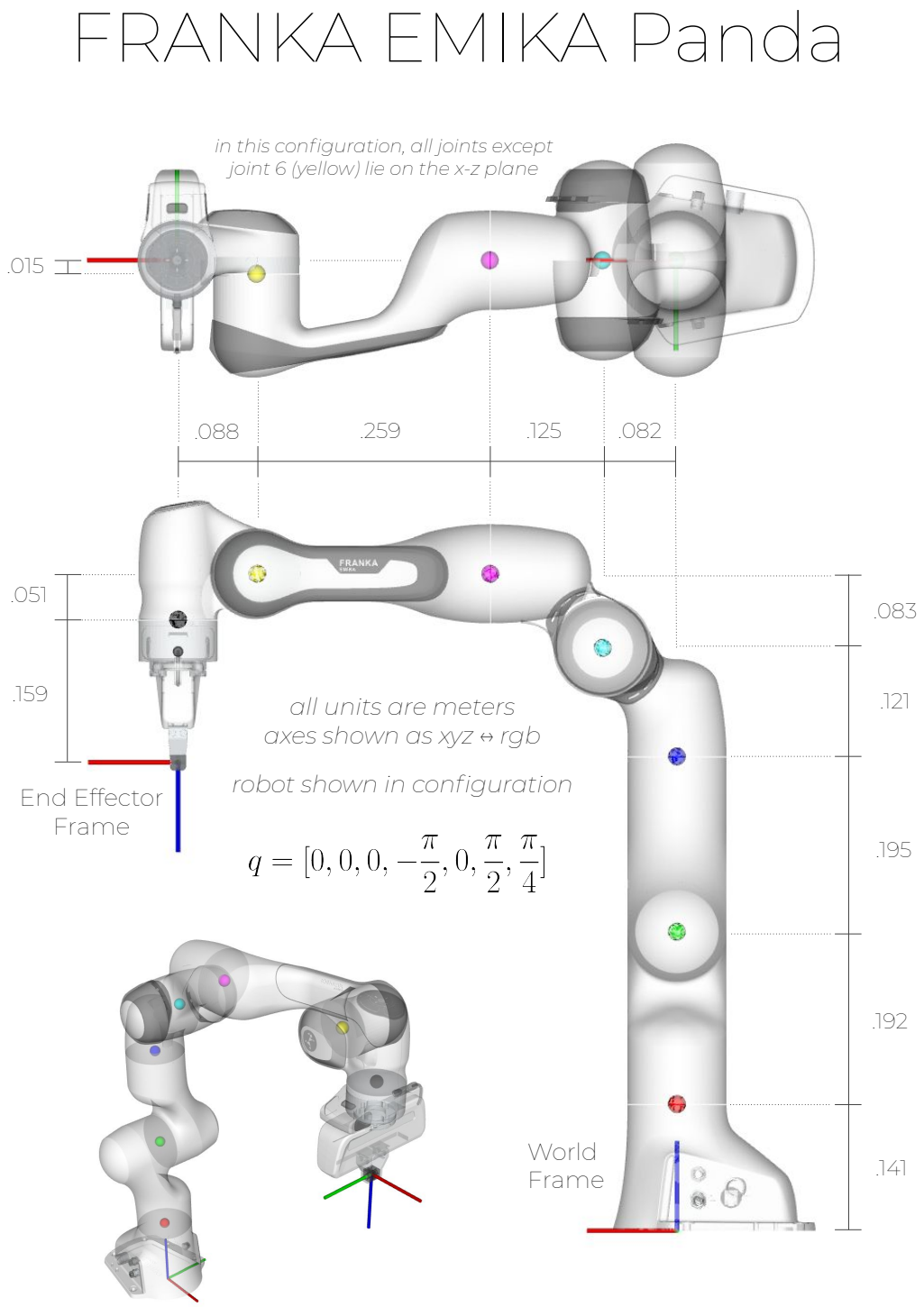

Figure 2: This schematic provides all the dimensional information necessary to formulate the forward kinematics of the Panda arm. The position of each joint is shown as a colorful dot *(in order: red, green, blue, cyan, magenta, yellow, black)* along with the end effector and world frames we have assigned. If the axis of rotation and positive sense of rotation are unclear for any joint, try playing around in simulation to resolve any ambiguities.

3.2 Formulation

It's recommended to start by planning out your strategy for the forward kinematics on paper first. Your solution should contain the following key components:

1. Assign coordinate frames for each link of the robot, according to some convention (either established or of your own creation)
2. Compute the relative transformation between each consecutive pair of coordinate frames as a function of the robot's configuration q
3. Determine how to compose these transformations in order to compute the transformation between any given link and the world frame
4. Compute the full pose of the end effector, and the position of each joint, using these transformations and any other dimensional information you may require.

Note: You will want to include your work in this section in your report (see below for details).

3.3 Implementation

You will need to implement your solution in `calculateFK.py`. Open the `lib` folder using your preferred text editor, for example:

```
atom ~/meam520_ws/src/meam520_labs/lib/
```

and implement the `FK` class, particularly the `forward` method. You are free to define other class methods to modularize your solution. *Note: if you are having problems with Atom, or want to use more advanced debugging features, you are free to install and use any editor of your choice - we will not need any features specific to Atom, it was just a suggestion.*

3.4 Testing

We recommend that you test your solution separately from the simulator first. Examine the bottom of `calculateFK.py` to find a *very minimal* example of a test. To see the output from this simple test, run this script from the terminal, i.e.,

```
cd ~/meam520_ws/src/meam520_labs/lib/
python calculateFK.py
```

You should perform more extensive testing yourself by adding to the tests done in `calculateFK.py`.

Once you are ready, you can test your solution using the simulator. First launch the simulator:

```
roslaunch meam520_labs lab1.launch
```

This week, we are running Gazebo *headless*, meaning that although the simulation is running in the background, the visualization and GUI will not appear. Instead, we will visualize the robot in RViz, another robotics development tool, which allows us to annotate the simulation with geometric markers. Try running

```
cd ~/meam520_ws/src/meam520_labs/labs/lab1
python visualize.py FK
```

This script will use your FK implementation to draw the positions of each joint and the end effector frame, and will provide a more geometric means of verifying the correctness of your solution. Press enter to click through several different configurations and visualize the results. Note: you do not need to relaunch the simulator between runs of `visualize.py` unless you kill the script early.

If your implementation is correct, it should look like the joint points and end effector frame are glued to the robot as it moves, although there may be some minor jitter, and the visualization should also match the bottom left of Figure 2. You should define additional elements in the `configurations` list in the `visualize.py` script to more extensively test your solution. In the "Views" panel on the right of RViz, we've also predefined several different views of the robot. One is a perspective view, and the rest are parallel projections which provide you an undistorted view of the robot from each of six axis-aligned directions. These may be helpful to you in debugging your solution.

4 Planar Inverse Kinematics

In this part of the lab, you will implement IK for targets in the world's x-z plane. The targets include a point relative to the world frame and an angle about the world's y axis. The output will be a set of joints that result in the end effector reaching the target. This simpler inverse kinematics problem serves as a preview of the next lab, where you will implement an inverse kinematics algorithm for the full 3D pose of the end effector.

4.1 Abstract and Physical Robots

For this lab you are doing inverse kinematics in the plane, which means we will be thinking of the 7-DOF Panda robot as a much simpler 3-DOF RRR robot (3 planar revolute joints) - in fact, the same robot you considered in the Prelab! We will refer to the Panda robot as the physical robot and the RRR robot as the abstract robot. This use of abstract models is very common in robotics when the physical robot is very complicated. The general procedure is to first convert your goal/target from the physical space to the abstract space (`physical_to_abstract_target`), then plan in the abstract space (`rrr_abstract_ik`), before converting the abstract joints to the physical joints (`physical_to_abstract_joints`).

4.2 Abstract Robot Geometry and Conventions

The abstract robot consists of 3 revolute joints (positive counter clockwise). The zero configuration of the robot is the arm fully extended pointing in the x direction. The target in the abstract space is given as the end effector position relative to the abstract robot's base and an angle of rotation about the abstract robot's z axis. For the link lengths a_i, see the provided code in `rrr_abstract_ik`.

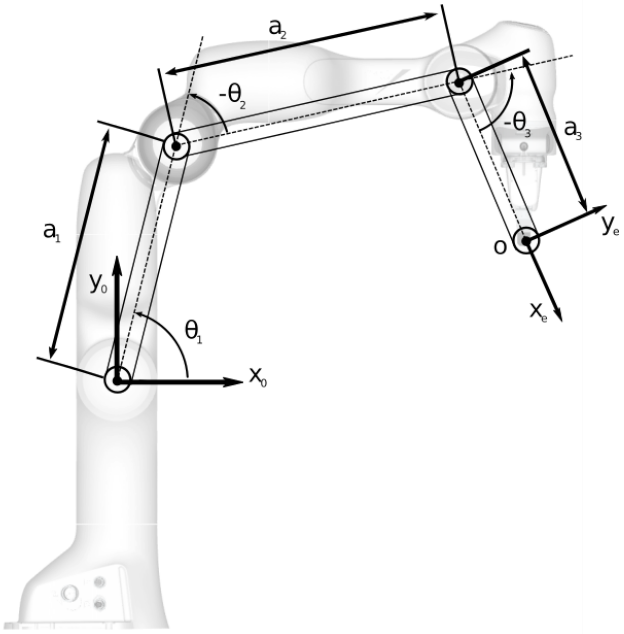

Figure 3: This schematic shows how the RRR robot is embedded in the full Panda robot. It also includes the conventions of the RRR robot.

4.3 Implementation

Your job is to implement `rrr_abstract_ik` in `lib/PlanarIK`. We have provided the relevant code to convert between the physical and abstract robot. `rrr_abstract_ik` takes in a dictionary with an abstract target and outputs a set of abstract joint angles. Since the abstract IK problem has 2 solutions, you should return both possible solutions.

Hint: *can you break the IK problem down into two simpler, decoupled subproblems?*

4.4 Testing

Remember that it is good practice to first test your code manually using your own test scripts. Once you are ready, then relaunch the simulator:

```
roslaunch meam520_labs lab1.launch
```

and try running

```
python visualize.py IK
```

This script will use your IK implementation to attempt to reach 4 provided targets one by one. Press enter to cycle through the different targets. If your implementation is correct, the robot will be able to exactly hit all the targets. You should define additional targets in the `targets` list in this script to more extensively test your solution.

Remember that using the predefined RViz views may be helpful in verifying that your solution is correct.

5 Report

You will submit a written report describing your work in this lab. **The report must be no more than 8 pages.** You are free to use whatever typesetting workflow you are comfortable with, but we highly recommend using LaTeX. Handwritten submissions will not be accepted. If you need to sketch some diagrams by hand, this is okay. Your report should address all of the tasks listed in this section.

5.1 Methods

Define the tasks your solutions perform, for both forward kinematics and planar inverse kinematics. Describe the approach you followed to solve each problem. Your exposition should cover each of the four numbered components in Section 3.2, as well as how you solved the IK problem for the RRR robot. Please include any diagrams that are necessary to clearly explain your methods.

5.2 Evaluation

You should extensively test your solutions to ensure their correctness. In this section, document your testing process: how can you be confident that your solutions are free of bugs? Additionally, you should include screenshots of the robot in three ***new*** configurations of your choosing, as well for three ***new*** planar inverse kinematics targets.

5.3 Analysis

In this section, you will analyze the performance of your solutions and consider some extensions. You should comment on each of the below tasks in your report.

5.3.1 Gravity

The simulator has been running with gravity turned off for this lab. Try re-enabling gravity by editing the following file:

```
atom ~/meam520_ws/src/meam520_labs/ros/meam520_labs/worlds/lab1.world
```

and temporarily deleting the line that says

```
<gravity>0 0 0</gravity>
```

to default back to Earth gravity. Try rerunning some of your IK and FK tests. Identify and try to account for any differences in the outcome.

5.3.2 Reachable Workspace

In class and in the Prelab, we discussed the *reachable workspace* of a manipulator. How can you use your solutions to this lab to generate a visual representation of the Panda's reachable workspace?

- Edit the file `workspace.py` in your `lab1` folder to create plot(s) of the reachable workspace. The exact methodology you use is left up to you to determine.
- Consider what assumptions you may have made in generating such a graphic. Can the robot truly locate its end effector at any point within your plot, or are there some points which might still be impossible? Why? Comment on this in your report.

5.3.3 Extending Inverse Kinematics to 3D

In this lab you implemented IK in 2D. Please discuss some of the challenges of implementing 3D IK on the Panda arm and how one might approach it. Some things to consider:

- Does the Panda robot have a spherical wrist?
- In class you learned about kinematic decoupling, and probably used it in your planar solution. Will this work on the full Panda robot? Why or why not?
- What challenges will the 7-dof nature of the arm pose, compared to a more standard 6-dof arm?

6 Submission via Gradescope

- **Group Submission:** Each group should only submit once to Gradescope per Gradescope assignment. While going through the submission please ensure that you **add your group members to the submission**. Failure to do so could impact the grades of all parties involved.
- Please submit `calculateFK.py`, `PlanarIK.py`, `workspace.py`, and any other files needed to run your code to the Lab1 Code assignment on Gradescope. We will not run autograder tests on `workspace.py`, but you are still required to submit it. Once submitted, we will run automated tests to grade your code. You may submit as many times as you want, but any attempts to try to figure out what test cases we are running will result in your code getting a 0.
- Please submit your written report as a PDF to the Lab1 Report assignment on Gradescope.

IMPORTANT, please submit the correct thing to each assignment. Don't submit the code to the report assignment. Don't submit the report to the code assignment.