CS 309 Robot Learning (FRI I) Homework #2

(Spring 2024) Due: **2/13**

Please complete the questions below.
Submit a zip file with your written answers (pdf), code (python scripts), and videos on Gradescope. You may optionally complete this assignment in pairs if you so choose. Clearly indicate your partner on your written answers. You will be graded on your submitted material and with an interview with a peer mentor to assess your understanding of what you submitted.

Be sure to include the following attestation in your written answers.

> Attestation: By submitting this assignment I affirm that these exercises were completed in accordance with the University's academic integrity policies and the guidelines put forth in the course syllabus.

Preamble: Setup ROS
This assignment requires that you program using the Robot Operating System (ROS). For your convenience, the `cereal` Ubuntu machines (kix, fruity-pebbles, etc.) in the basement North computer lab (GDC 1.310) have all dependencies installed. If you have a machine running **Ubuntu 20.04**, you may install ROS **noetic** and the requisite packages — see the course website for setup instructions. Peer mentors can assist with setup. Do not attempt to run ROS in a virtual machine, or on Mac or Windows directly. Discuss with the peer mentors if you wish to dual boot your machine (and be sure to create a backup first).

First, set up `git` and add your ssh key. See the instructions on the course website here. The department machines already have git installed.

Let's prepare for the homework. Clone the code for these exercises in your home directory:

```
cd
git clone git@github.com:UT-Austin-RobIn/robot_learning_ws.git
cd robot_learning_ws
git submodule init
git submodule update
```

Build your `catkin` workspace. This will compile the code included in the `src` directory.

```
source /opt/ros/noetic/setup.bash
cd ~/robot_learning_ws
catkin_make
```

Finally, whenever you open a new terminal, source your workspace's `setup.bash` file. This allows packages inside the workspace to be found by ROS.

```
source devel/setup.bash
```

You can add that line to your .bashrc file

```
echo "source ~/robot_learning_ws/devel/setup.bash" >> ~/.bashrc
```

so that the workspace is ready to go automatically.

Problem 1: `rosbag`
Consult the `rosbag` commandline tools documentation here, and `rostopic` documentation here.

a. Start a new ROS master.

b. (3 points) Included in the `src/hw2` directory is the rosbag file `hw2.bag`. In a new terminal, play the bagfile in a loop. Include the command you used in your written answers.

c. (3 points) Open a second terminal. Use `rostopic` to identify the list of topics available. Include your command and the result.

d. (3 points) Print one message from the `/robot/joint_states` topic.

e. (3 points) What type of message is published on the topic `/robot/joint_states`?

Problem 2: `callback`
Implement a simple callback function. Play the bagfile from the previous question to test your implementation. To run your code:

```
cd ~/robot_learning_ws/src/hw2
python3 listener.py
```

Note that on GDC lab machines, saying "python3" is required — on these machines, "python" refers to "python2.7". On your personal machine, "python" might work.

a. (3 points) Edit the file `src/hw2/listener.py`.

b. (3 points) Initialize the node.

c. (3 points) Subscribe to `/robot/joint_states` topic.

d. (3 points) Implement the callback function. Print the message header and convert the joint states message to a numpy array and print it.

Problem 3: `MoveIt!`
Let's move the robot!

a. Kill your rosbag and rosmaster from the previous questions.

b. In one terminal, launch the robot:

```
roslaunch sawyer_moveit_config sim_sawyer_moveit.launch electric_gripper:=true
```

c. Complete the missing function in the file `src/hw2/moveit.py`. In particular,

   (a) (3 points) Compute a new Cartesian pose goal that is 5cm in the x direction in the gripper local frame.

   (b) (3 points) Form the `Pose` message and call the planner.

   (c) (3 points) Start a second shell, and run the code like so.

```
cd ~/robot_learning_ws
source devel/setup.bash
cd src/hw2/
python3 moveit.py
```

   Record a video of the simulated robot moving to the new goal pose. On Linux machines, this can be done easily with `SimpleScreenRecorder`.