

Week 8 Lab – Manipulation Challenge

This lab sheet will explore the fundamentals you will need next week to create your submission for the third challenge, which is about grasping. **Your end goal after 2 weeks will be to get an OpenManipulator robot to successfully pick objects and sort them into boxes based on their size**

Exercise 1: Simulating the OpenManipulator

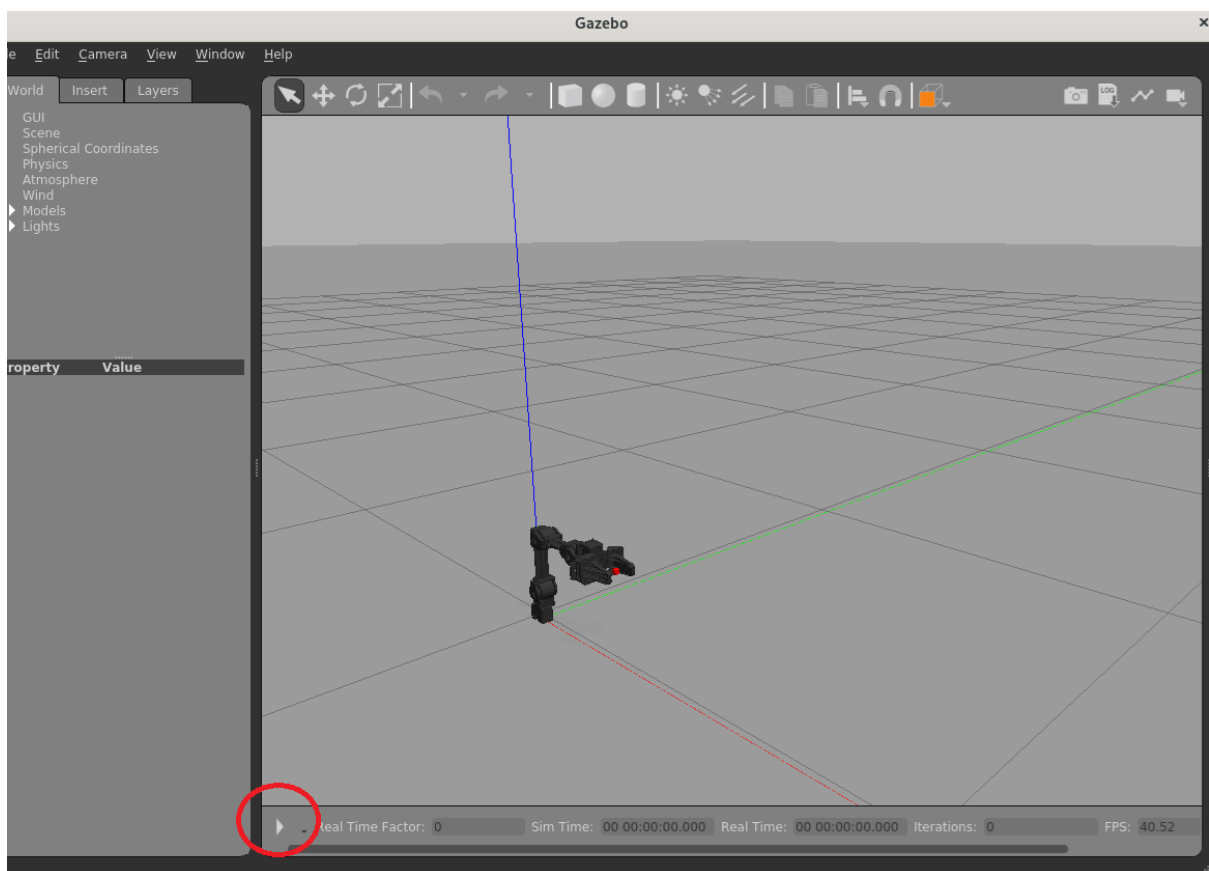
Source the OpenManipulator workspace: In your `.bashrc`, source the following workspace to get access to the necessary packages:

```
/vol/vssp/signsrc/openmanipulator_ws/devel/setup.bash
```

First lets load the robot controller. In a new terminal roslaunch the **open_manipulator_controller.launch** file from the package **open_manipulator_controller**. By default it will try to use the real robot. Add the argument **use_platform:=false** to the end of your launch command to make it simulate the arm.

In order to see what we are doing we should also launch a gazebo visualisation. To do so launch the **open_manipulator_gazebo.launch** file from the package **open_manipulator_gazebo**.

You should now see the arm in gazebo as below. Make sure to tick the “play” button in the bottom left corner to start updating the arm in gazebo.



Week 8 Lab – Manipulation Challenge

Explore the model in gazebo. If you expand “**open_manipulator**” in the top left you can see the list of links and joints associated with the model.

In the terminal, list the active rostopics to see what the controller is listening to (generally the topics ending in command) and what it is publishing. You can also try viewing the **rqt_graph**.

Now try to echo the `/joint_states` topic. How do you think the different values in these messages correspond to the joints in Gazebo? (Note, if no messages are echoed, you may have forgotten to click the “play” button in gazebo)

Exercise 2: Teleoperation

In a new terminal, launch the **open_manipulator_teleop_keyboard.launch** file in the **open_manipulator_teleop** package.

Try increasing and decreasing the joints directly using (y,h,u,j,l,k,o,l,g,f). Did the various joints match the behaviour you expected based on inspecting the model?

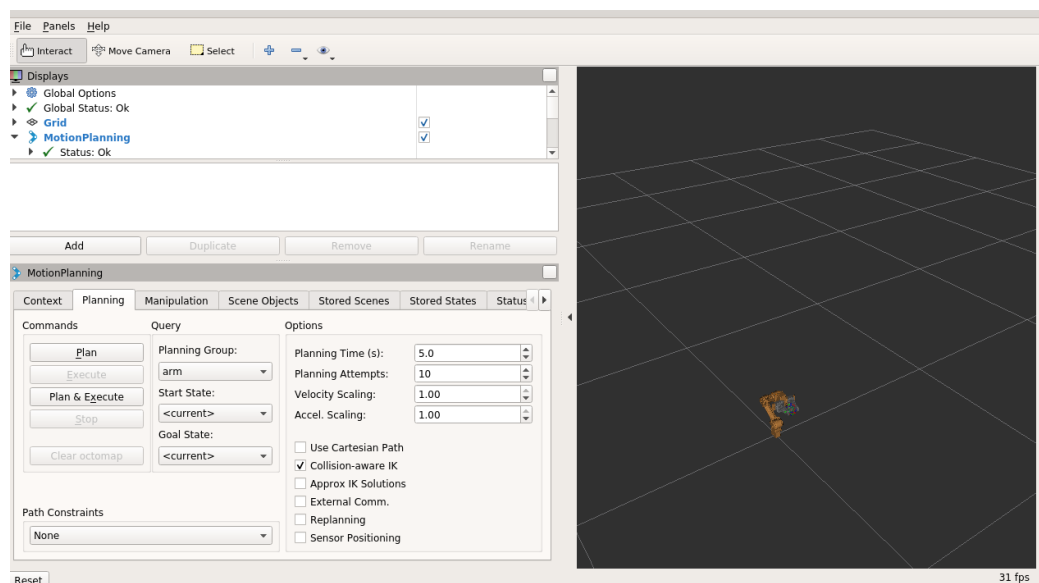
Next try teleoperating using the “task space” controls (w,s,a,d,z,x).

How does the task space control differ from the joint control?
When might you want to use one or the other?

Exercise 3: Moveit

We will now try a more advanced approach to controlling the arm. First close the gazebo and keyboard teleoperation nodes (but keep the controller node running). Next launch **joint_trajectory_controller.launch** from the **open_manipulator_controllers** package, and pass the argument “sim” as true.

Once again you should see Gazebo open up, but you should also see an RVIZ window with moveit controls. (Note, if RVIZ takes a long time to appear, try pressing the “play” button in gazebo).



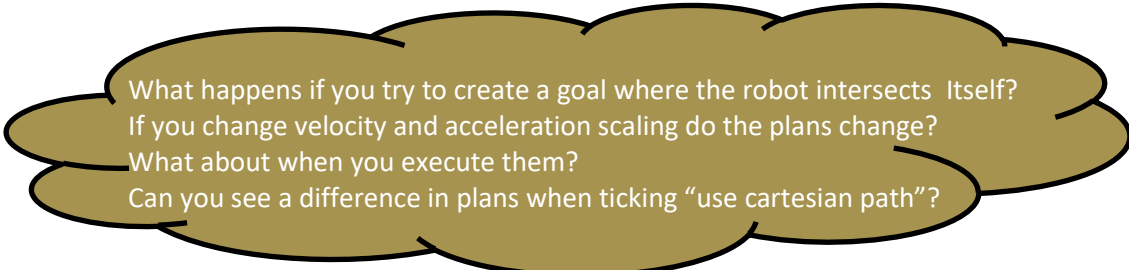
Week 8 Lab – Manipulation Challenge

Try dragging the arrows at the end of the gripper to change the target goal position, and drag the wheels to change the target orientation. Then click “plan and execute” to move the arm into the target pose.

If you receive a lot of failed executions with the message “GOAL_TOLERANCE_VIOLATED”, you can try reducing the velocity and acceleration scaling. However, you will notice that the robot is roughly in the right place regardless.

For more detailed complex movements, such as turning the arm side-to-side you will need to use the “joints” tab to control some of the joints individually. This will also allow you to open and close the gripper when the “gripper” group is selected.

You can inspect the planned path by clicking “panels” at the top of rviz, then “motion planning – trajectory slider”. Try clicking pause and dragging the slider to see the individual poses on the path.



What happens if you try to create a goal where the robot intersects itself?
If you change velocity and acceleration scaling do the plans change?
What about when you execute them?
Can you see a difference in plans when ticking “use cartesian path”?