Project 5 FAQ

Bookmark this page

- **How do run this project in my own Ubuntu machine?**

  1. Launch Project 5, then in Vocareum click Actions>Download Starter code. This will download all the files you need to make the project run locally in your computer.
  2. Install the needed ROS package(s). Run the following lines on your terminal:
     ```
     sudo apt-get update
     sudo apt-get install python-wstool ros-kinetic-moveit*
     ```
     Replace kinetic with the ROS version that you are running on your local machine.
  3. **IGNORE** all the files other than the catkin_ws folder and kuka_lwr_arm.urdf file. Put the catkin_ws and the kuka_lwr_arm.urdf file in your home directory.
  4. The downloaded files are structured as a catkin workspace. You can either use this structure directly (as downloaded) and build the workspace using the "catkin_make" command or use whatever catkin workspace you already had, and just copy the packages inside your own src folder and run the catkin_make command. If you are having troubles with this, you should review the first ROS tutorial "Installing and configuring your ROS Environment".
  5. Once you have a catkin workspace with the packages inside the src folder, you are ready to work on your project without having to make any changes in any of the files. Navigate to the catkin workspace folder and build the workspace using the command "catkin_make".
  6. NOTE: You can source both your ROS distribution and your catkin workspace automatically everytime you open up a terminal automatically by editing the ~/.bashrc file in your home directory. For example if your ROS distribution is Kinetic, and your catkin workspace is called "project5_ws" (and is located in your home directory) then you can add the following at the end of your .bashrc file:
     ```
     source /opt/ros/kinetic/setup.bash
     echo "ROS Kinetic was sourced"
     source ~/project5_ws/devel/setup.bash
     echo "project5_ws workspace was sourced"
     ```
     This way every time you open up a terminal, you will already have your workspace sourced, such that ROS will have knowledge of the packages there.
  7. **Before moving forward, if you haven't followed the instructions on step 6, you will need to source ROS and the catkin workspace every time you open a new terminal.** To run the project, first open up a terminal and run `"roslaunch motion_planning mp.launch". In the second terminal, run "rosrun motion_planning marker_control.py". Note that you do NOT have to run "roscore" as roslaunch includes all the necessary packages.`
  8. On another 2 separate terminals, `run "rosrun motion_planning motion_planning.py"` (this is what you need to edit to complete the project), and as always `"rosrun rviz rviz"` to visualize the robot.
  9. On rviz, you will need to add a RobotModel, InteractiveMarker, and Marker. When you add the InteractiveMarker, click on "InteractiveMarker" to expand it, and select /control_markers/update as the update topic. You shouldn't need to do anything for the RobotModel and the Marker.

- **Pyassimp error with kinetic**

Edit "/usr/lib/python2.7/dist-packages/pyassimp/core.py" as follow :

```
-    load, load_mem, release, dll = helper.search_library()
+    load_mem, release, dll = helper.search_library()
```

(You should navigate to this path and use sudo gedit core.py to edit the file. Once its open find the mentioned line (-) and replaced it with the one marked (+))

If it doesn't work, try to update the pyassimp module to latest 3.3 version.

```
sudo pip -H uninstall pyassimp

sudo pip -H install pyassimp
```

- **XML error running roslaunch locally on kinetic ubuntu**

  You have to edit the file lwr_robot/lwr_defs/defs/util_defs.xml and ensure matching parantheses in the marcro definitions as following :

```
<?xml version="1.0"?>
<robot xmlns:sensor="http://playerstage.sourceforge.net/gazebo/xmlschema/#sensor"
       xmlns:controller="http://playerstage.sourceforge.net/gazebo/xmlschema/#controller"
       xmlns:interface="http://playerstage.sourceforge.net/gazebo/xmlschema/#interface">

  <property name="M_PI" value="3.1415926535897931" />

  <!--
     Little helper macro to define the inertia matrix needed
     for links.
     -->
  <macro name="cuboid_inertia_def" params="width height length mass">
    <inertia ixx="${(mass * (height * height + length * length) / 12)}"
                 iyy="${(mass * (width * width + length * length) / 12)}"
             izz="${(mass * (width * width + height * height) / 12)}"
             ixy="0" iyz="0" ixz="0"/>
  </macro>

  <!-- length is along the y-axis! -->
  <macro name="cylinder_inertia_def" params="radius length mass">
    <inertia ixx="${(mass * (3 * radius * radius + length * length) / 12)}"
                 iyy="${(mass * radius* radius / 2)}"
             izz="${(mass * (3 * radius * radius + length * length) / 12)}"
             ixy="0" iyz="0" ixz="0"/>
  </macro>

</robot>
```