# Teleoperate the real UR5 by keyboard

# 1 [Task Summary]

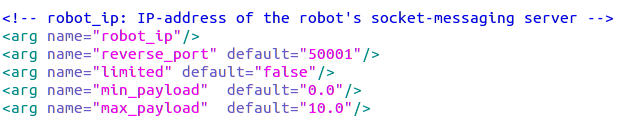
Here we got a UR5 model and configuration file. Besides, we have already been successful to teleoperate the UR5 on RVIZ platform. Now we expect we can control the real UR5 through keyboard input. We are going to build the connection between the real arm and PC through the ur\_driver package so that we can convert the messages from PC to the real servos.

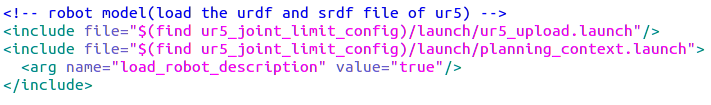
# 2 [Process]

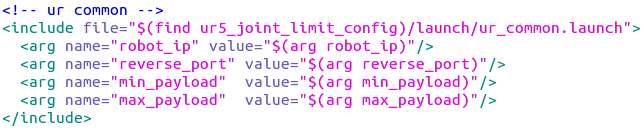
**Step 1:** Bring up the ur5\_bringup.launch file

# roslaunch ur5\_joint\_limit\_config ur5\_bringup.launch robot\_ip:=<ROBOT\_IP>

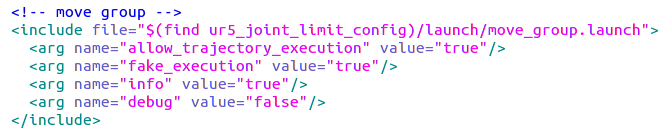
Let us take a look at this file.

At first, we receive a IP address of your real robot.

Then we upload the urdf and srdf file that was created by moveit\_setup\_assistant earlier before.



Now we include ur\_common.launch file which helps us to bring up the ur\_driver node. The ur\_driver node is the key to connect the real UR5 to our own PC.

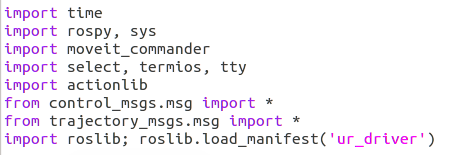


At last, we bring up the move\_group node for the usage of moveit function, including the solution of inverse kinematics.

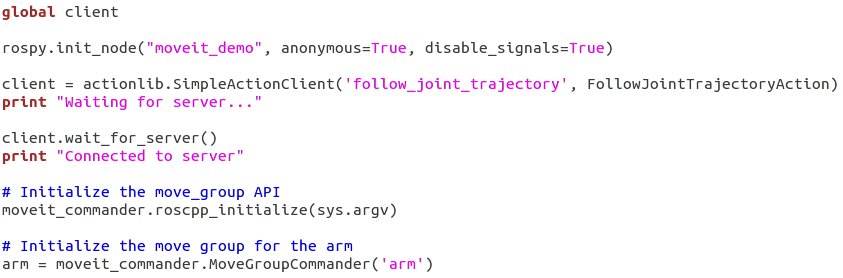
**Step 2:** Run the teleop.py

# rosrun ur5\_joint\_limit\_config teleop.py

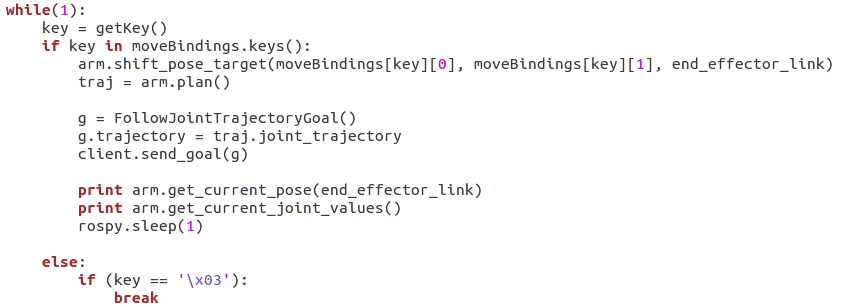
Let us divide the teleop.py into several parts.



Notice that we import **actionlib** and **roslib.load\_manifest(‘ur\_driver’)** for out next steps.



We initialize a client class here to send and receive messages from UR5. Besides, we also initialize the move\_group API and move group for the arm.



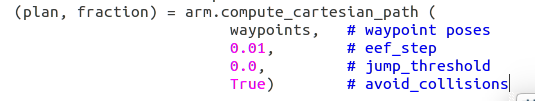
This part is the key to move the real arm. When we get a keyboard input, we will use **shift\_pose\_target()** function to set the target pose and the **plan()** function in the moveit group ‘arm’ to compute the trajectory of each joints. The solution will be put into traj, a **RobotJointTrajectory** class.

Then we create a **FollowJointTrajectoryGoal** class g to store the joint\_trajectory information and use **client.send\_goal(g)** to send it to UR5. After doing these, we will see UR5 move to the target pose.

**Step 2:** Run the teleop\_cartesian.py

# rosrun ur5\_joint\_limit\_config teleop\_cartesian.py

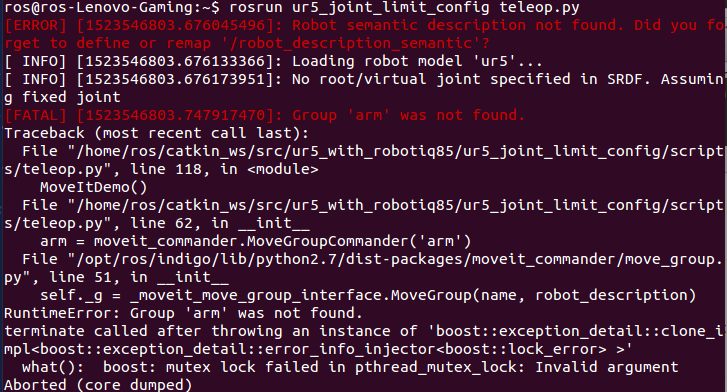
If you want your robot’s end effector go straight between 2 way points. You have to compute the cartesian path. The **arm.compute\_cartesian\_path()** function will help you to finish your assignment.



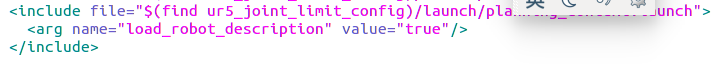
The usage of this function is quite simple. We only need to give the target poses we want the robot to reach to the function. As a result, it will return a robot trajectory if it find the solutions.

# 3 [Problems & Solutions]

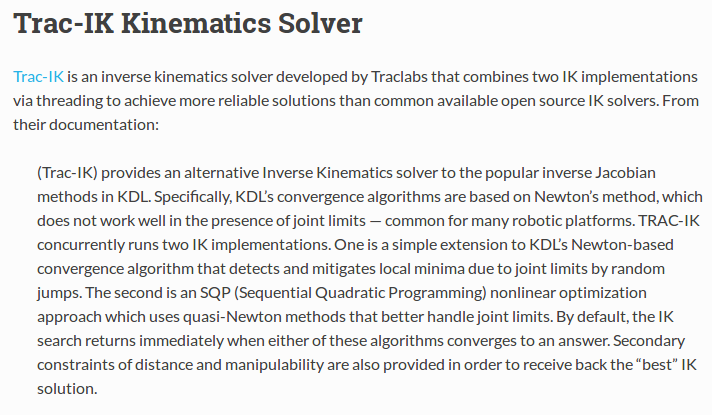
**3.1 Robot semantic description not found.**

****

**Solution:** This error means we haven’t uploaded the srdf file of your robot. You can simply include the planning\_context.launch file in the config folder.



**3.2 KDL system**

****