Start-up Instruction & Function Statement:

For Point_and_Click method:

1. **roslaunch finalasm_moveit_config1 demo.launch**
   – Load the arm model, subscribe point cloud and interactive marker, initialize Rviz environment

2. **roslaunch freenect_launch freenect.launch**
   – Start up Kinect, visualize point cloud in Rviz

3. **rosrun tf static_transform_publisher 0.75 -0.3 0.25 3.14 0 0 base_link camera_link 10**
   – Build tf connection between camera_link and base_link (set camera_link as child link)
   # The parameter should adjust according to the robotic arm's position in real world, and user should keep adjusting until the arm model in Rviz totally overlaps the real arm in point cloud

4. **roslaunch rail_agile_nodes find_grasps.launch**
   – Start node of find_grasp and grasp_sampler

5. **roslaunch remote_manipulation_markers point_and_click.launch**
   – Start node of point_and_click

6. **roslaunch rail_agile_nodes click_action.launch**
   – Start node of ClickedImageClient and point_and_clicker, generate a rgb image window and collect information when user click on the image window
   # User can click on the desired place of the image window to start finding possible grasps here

7. To cycle the different final poses use the service:
   rosservice call /point_and_click/cycle_grasps true
   # Parameter TRUE is for moving down the list and FALSE is to move up the list

8. To see current selected pose use the service:
   rosservice call /point_and_click/execute_robotic_arm true
   # To make the code more clear, I remove the arm controlling part from this service, so this service can simply transfer the gained coordinate relative to the camera_rgb_optical_frame to controlling part

9. **roslaunch moveit_tutorials move_group_interface_tutorial_test.launch**
   – To activate the robotic arm in both Rviz and real world

System required: Ubuntu 16.04 & Kinetic, Opencv 2.4.10, Eigen 3.3.4, PCL 1.7.2, VTK 6.2
#PCL 1.7.2 and VTK 6.2 are **default** version in Kinetic

Haoran Li
haoranli@gatech.edu