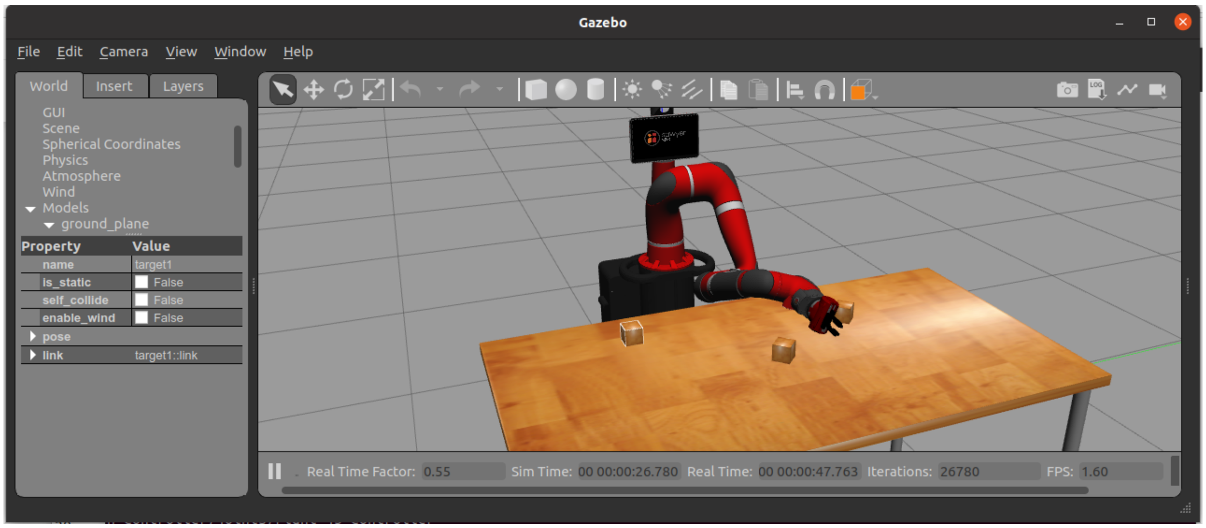
ECE 555 Computer Control of Robotics Course Project

In this project involves computer control of the sawyer robot for a continous pick and place action using the ROS/Gazebo Development environment. The course project consist of the following:



Part 1: Object Pose Detections [20 pts]

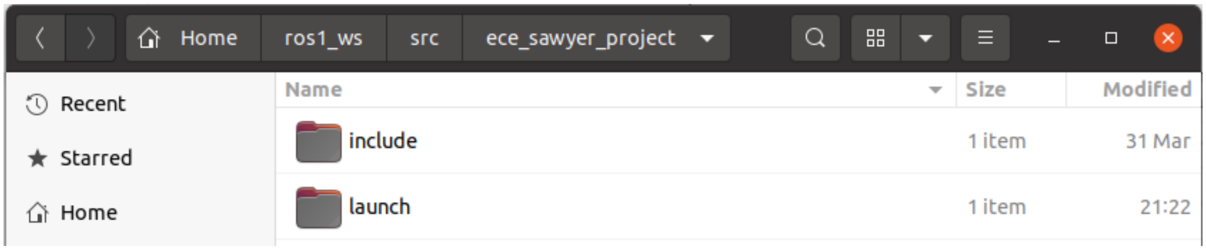
- A. Pose of the three targets with the repect to the robot base frame [10 pts]
- B. The joint variables to grasp the object at each location, ie. inverse kinematics [5 pts]
- C. The Jacobian matrix at each location using the joint variables from Part B [5 pts]

An example ROS Package, ece_sawyer_project, has been created to assist the project. This package is available to download using the following:

In [ ]:

```
git clone https://livingston_robotics@bitbucket.org/livingston_ai/ece_sawyer_
```

This resposity contains the following contents



worldfile [sawyer_eb2036.world]

In [ ]:

```
<?xml version="1.0" ?>
<sdf version="1.6">
  <world name="default">
    <include>
      <uri>model://ground_plane</uri>
    </include>

    <include>
      <uri>model://sun</uri>
    </include>

    <include>
      <uri>model://table</uri>
      <name>table</name>
      <pose>1 0 0 0 0 1.57</pose>
    </include>

    <include>
      <uri>model://wood_cube_5cm</uri>
      <name>target1</name>
      <pose>0.75 -.35 5 0 0 0</pose>
    </include>

    <include>
      <uri>model://wood_cube_5cm</uri>
      <name>target3</name>
      <pose>0.85 0.25 5 0 0 0.785</pose>
    </include>

    <include>
      <uri>model://wood_cube_5cm</uri>
      <name>target2</name>
      <pose>1 0 5 0 0 -0.785</pose>
    </include>

    <physics type="ode">
      <real_time_update_rate>1000.0</real_time_update_rate>
    </physics>

    <gravity>
      0.0 0.0 -9.81
    </gravity>

    <gui fullscreen='0'>
      <camera name='user_camera'>
        <pose frame=''>2.69836 -0.874828 2.04939 0 0.399643 2.75619</pose>
        <view_controller>orbit</view_controller>
        <projection_type>perspective</projection_type>
      </camera>
    </gui>

  </world>
</sdf>
```

The repository also contains example source code for motion planning of the sawyer robot. Part 2 involves writing a software algorithm to manipulate the sawyer robot. You may use the sample

Part 2: Computer Control of Robot System [50 pts]

Write a computer algorithm the performs the following task. (Example video available)

- 1. Pick up and discard target 2 from the scene
- 1. Pick up target from location 1 and place location 2
- 1. Pick up target from location 3 and place in location 1
- 1. Pick up target from location 2 and place in location 3
- Repeat steps 2 - 4

The program should be demostrated using the following actions

Bring up the simulator

In [ ]:
```
./intera.sh sim
```

In [ ]:
```
roslaunch ece_sawyer_project sawyer_world.launch
```

Enable the sawyer robot

In [ ]:
```
./intera.sh sim
```

In [ ]:
```
rosrun intera_interface enable_robot.py -e
```

Let's start by lanch the trajectory action server of the robot with the following command

In [ ]:
```
rosrun intera_interface joint_trajectory_action_server.py
```

Then run the moveit package again with the controller.

In [ ]:
```
./intera.sh sim
```

In [ ]:
```
roslaunch sawyer_moveit_config sawyer_moveit.launch electric_gripper:=true
```

Execute motion planning algorithm

In [ ]:
```
./intera.sh sim
```

In [ ]:
```
rosrun ece_sawyer_project sawyer_pick_and_place
```

Try to plan the trajectory and execute again, you should see the robot move while executing the trajectory.

- Documentation and Demo -

Documentation [20 pts]

Documentation of the algorith from Part 2 and the robot analysis from Part 1 should be submitted in the IEEE Conference Paper format no later than the end of the schedule final examination. Typicaly 3- 5 pages with refs.

Robot Demostration [10 pts]

The project must be demostrated to the instructor or TA. Groups may use the simulated or real environment for demostration. Other options include:

- Virtual Demos
- Recorded Demos
- Remote Lab access (pending)

In [ ]:

```

```