RBE-500

Final Assignment Part 2

Group 9: Abhishek Jain, Paurvi Dixit, Surya Murugavel Ravishankar

Objective:

1. Change the joint types for all the joints to fixed except the last joint.
2. Write a position controller node for the last joint.
    1. This node will get the joint positions from Gazebo and will be able to send joint efforts.
    2. Write a PD controller for the last joint.
    3. Write a service that gets a reference position for the last joint, and makes it go there.
    4. Record the reference position and current position of the joint in a text file, and plot them via Matlab.

Implementation:

Que 1:

The objective is to change the joint type of all the joints barring the last one to fixed.

- This was completed by changing the joint type in the robot's URDF file.
- Also, It was necessary to make changes in the control **yaml file** and **launch file** as previously the controller was defined for 3 movable joints.

Que 2:

The objective is to write a position controller node for the last joint and plot the current and reference position data.

- File named **control_service.py** is the required kinematics node.
- Server **'control_server'** provides the service named **'control'** which has **'control'** as service type.
- The callback function of the server receives the desired reference position from the user for the last joint
- It publishes this reference position to a topic **"/custom_scara/joint3_position_controller/command"**
- This topic is part of **joint effort controller** which calculated the necessary effort for the joint to reach that reference position.
- A subscriber is also defined to subscribe to topic: **"/custom_scara/joint_states".**
- This topic provides the current joint state from Gazebo.
- The data from subscriber in combination with reference position is plotted using python plot to get the desired plots.
- The **X coordinate** represents the **time** (upto 10 sec.) and the **Y coordinate** represents the **current/reference position**.

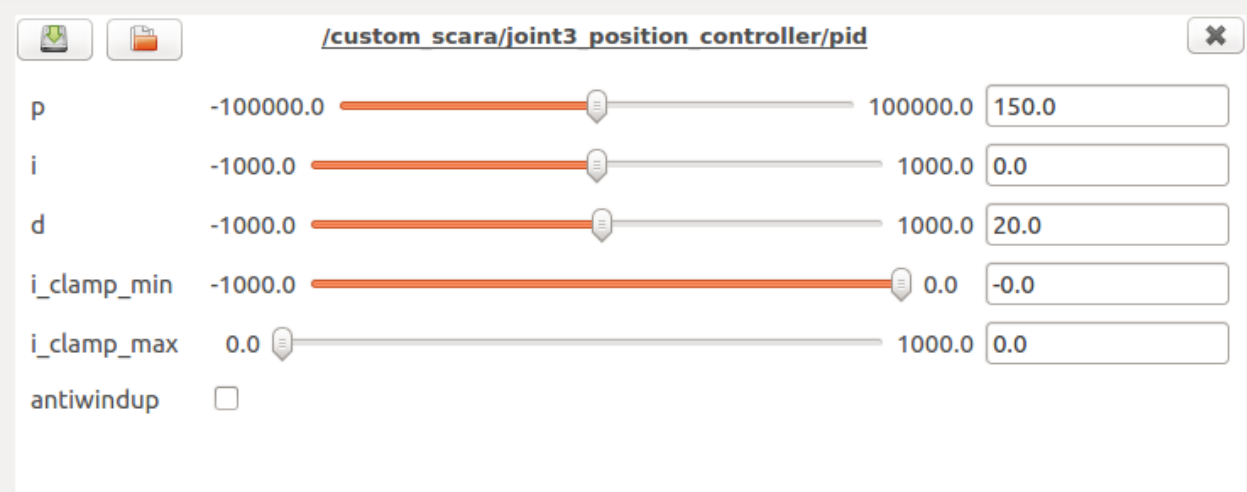
Below is the code snippet of **control_service.py**

```
def callback(data):
    global time_start
    time_start = rospy.get_rostime().secs
    time_start_n = rospy.get_rostime().nsecs

    global i
    global r
    global ref
    ref = data.q
    r = 1
    i = 1
    pub.publish(ref)
    return controlResponse()


def control_server():
    rospy.init_node('control_server')
    s = rospy.Service('control', control, callback)
    rospy.Subscriber('/custom_scara/joint_states', JointState, callback1)
    rospy.spin()
```

The PD controller was tuned with the following gains:



Below is the Service call command used to call the service.

```
killswitch@Legion:~/catkin_ws$ rosservice call /control -- '-3'
```

```
killswitch@Legion:~/catkin_ws$ rosservice call /control -- '-1'
```

The Plot results and corresponding simulations were as follows:.

1) **From position 0 to -3:**

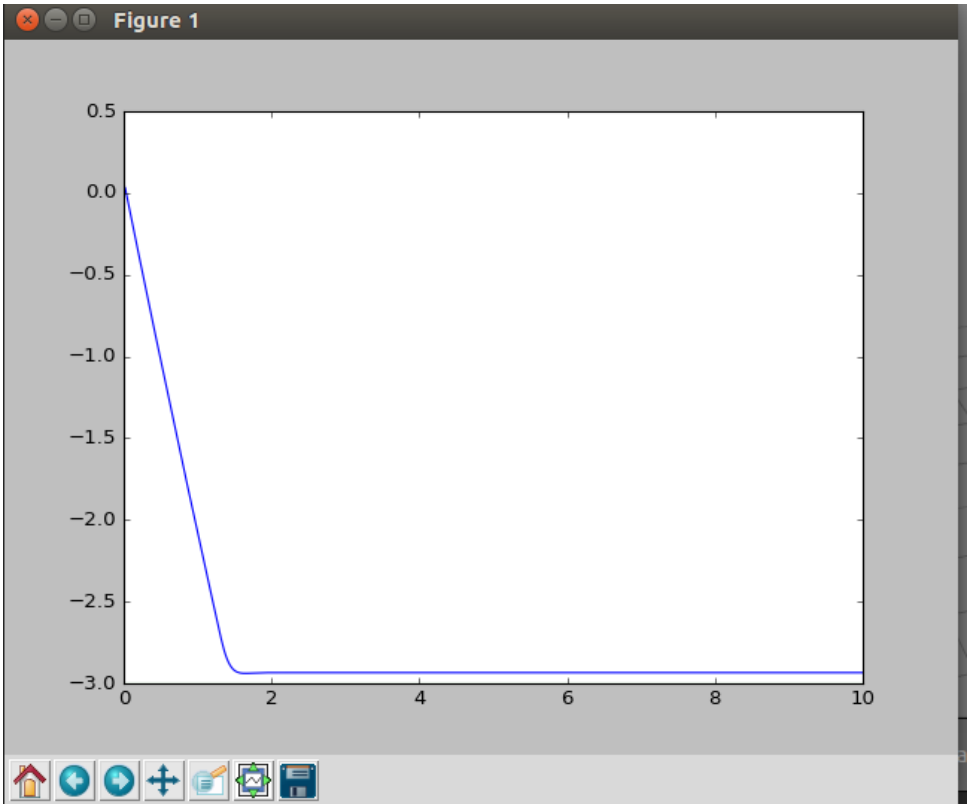


Fig: Plot of Current/Ref. Position wrt. Time

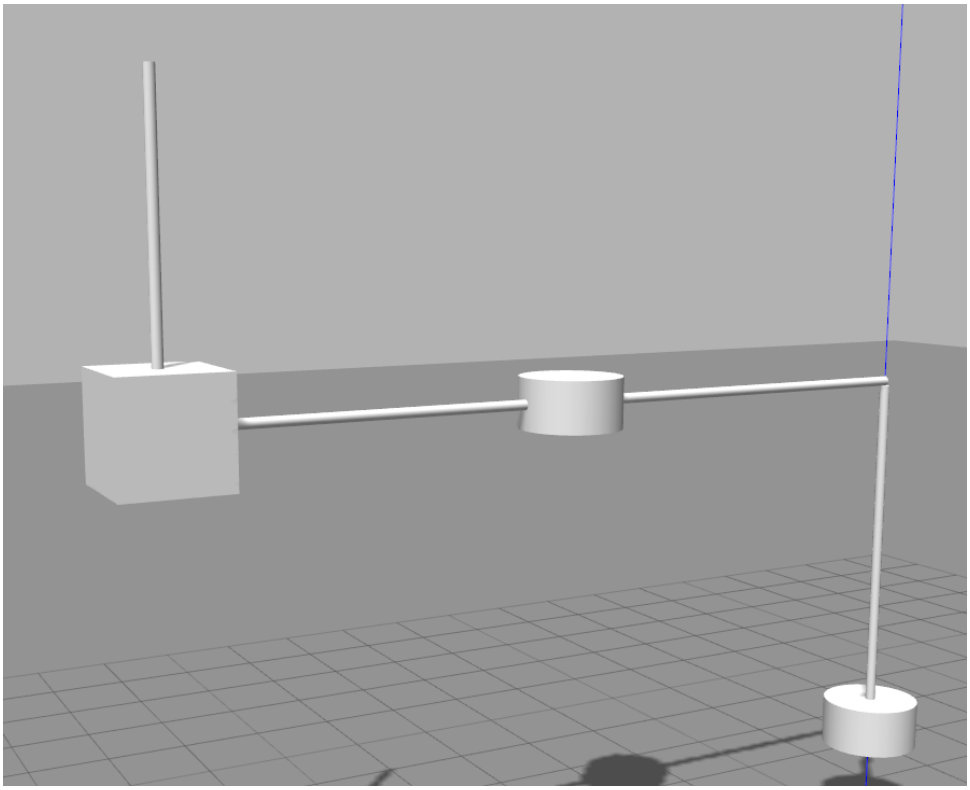


Fig: Gazebo Simulation

2) **From position -3 to -1:**

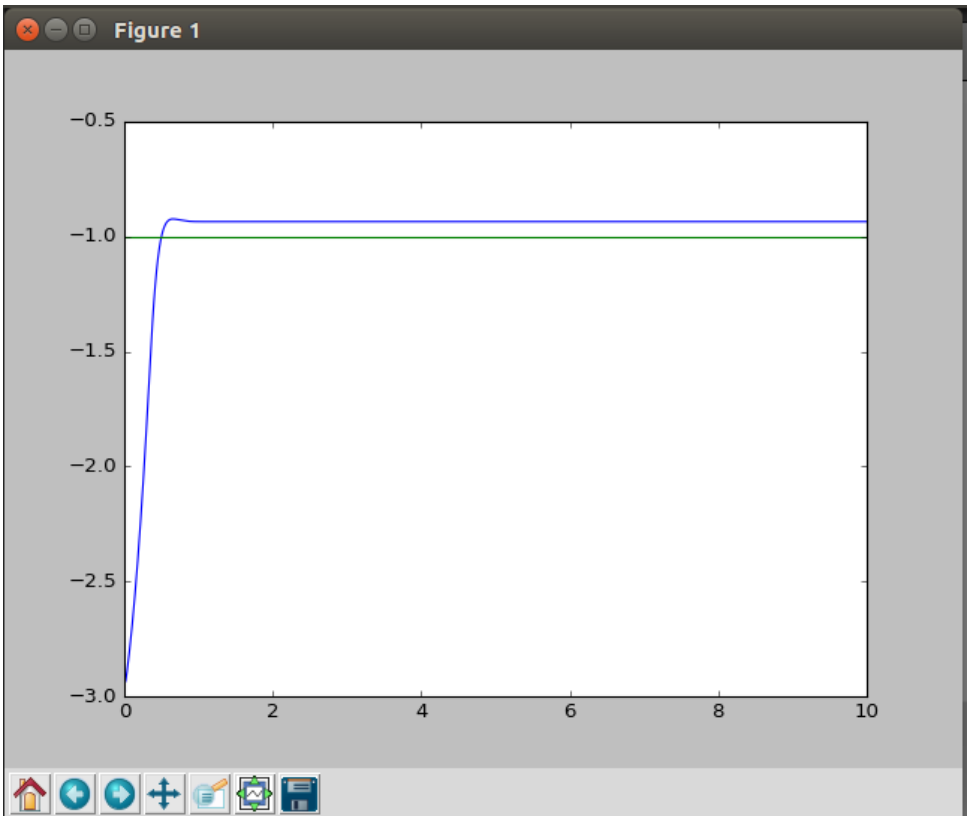


Fig: Plot of Current/Ref. Position wrt. Time

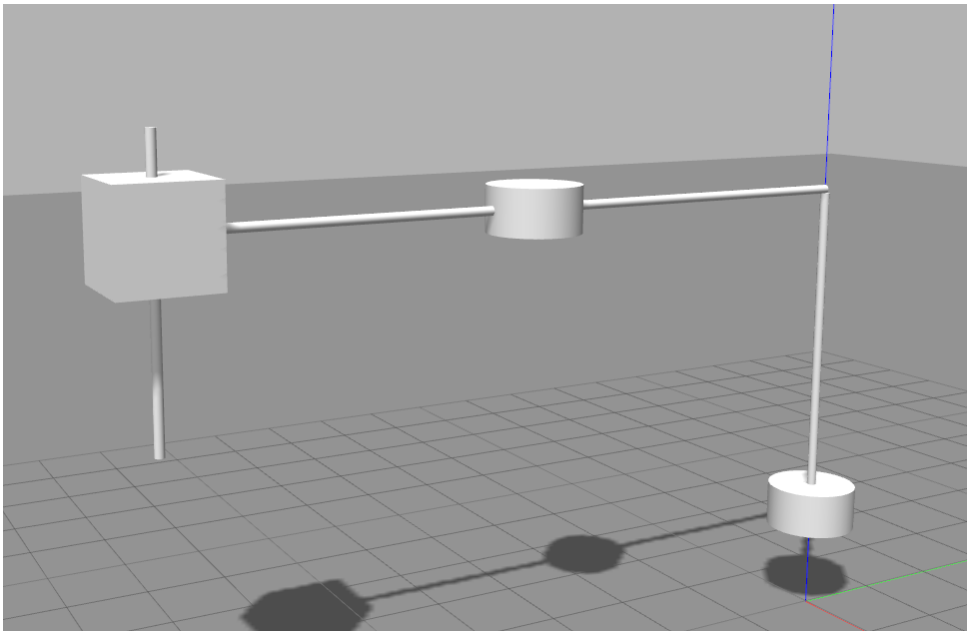


Fig: Gazebo Simulation