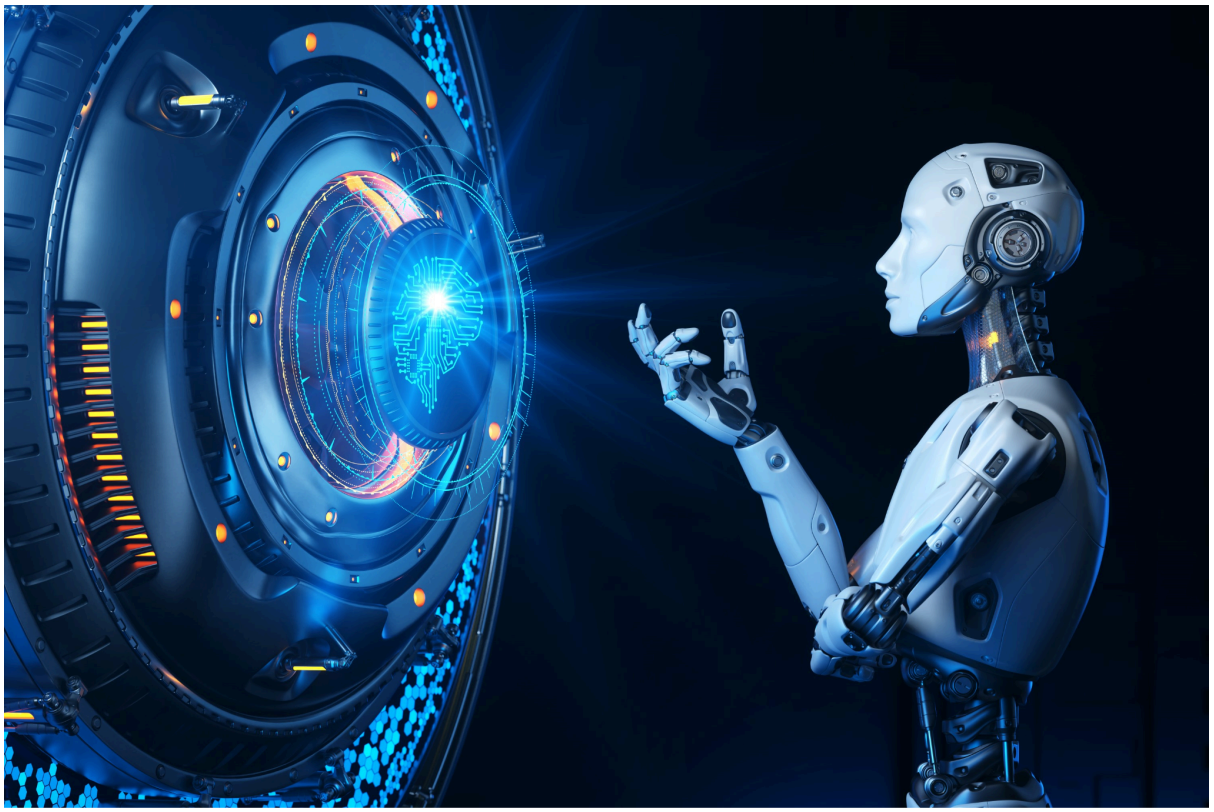
Instituto Tecnológico y de Estudios Superiores de Monterrey

Campus Monterrey

Reporte de Mini Reto



Hecho por:
José Miguel Flores González | A01383155

11 de Abril del 2024

Ecuaciones:

$$x2_{dot} = \frac{1}{J}(\tau - mga\,cos(x1) - kx2$$

$$J = \frac{4}{3}ma^2$$

Repositorio:

https://github.com/DevasNAI/Electro-Horchatas-Autonomous-Challenge/tree/Mike's-98-Toyota-Corolla

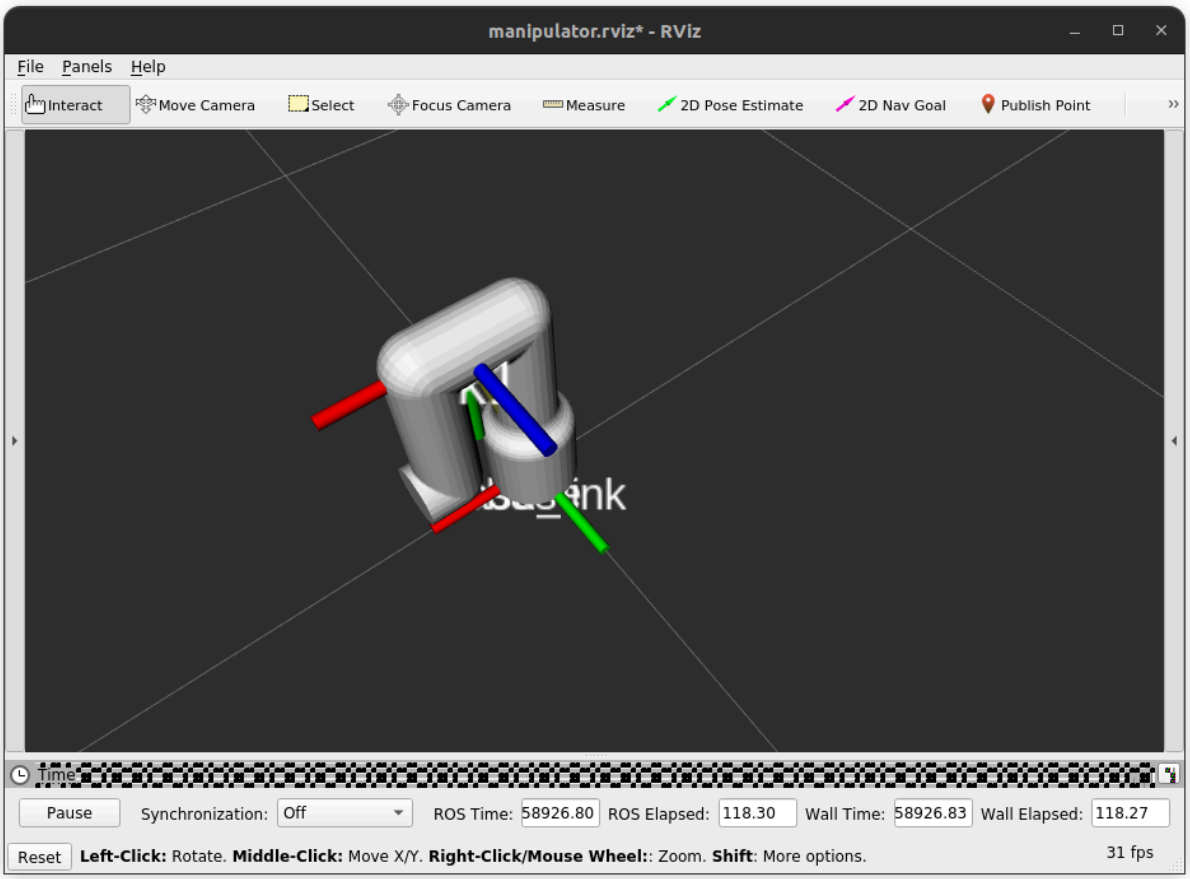
Código de Python:

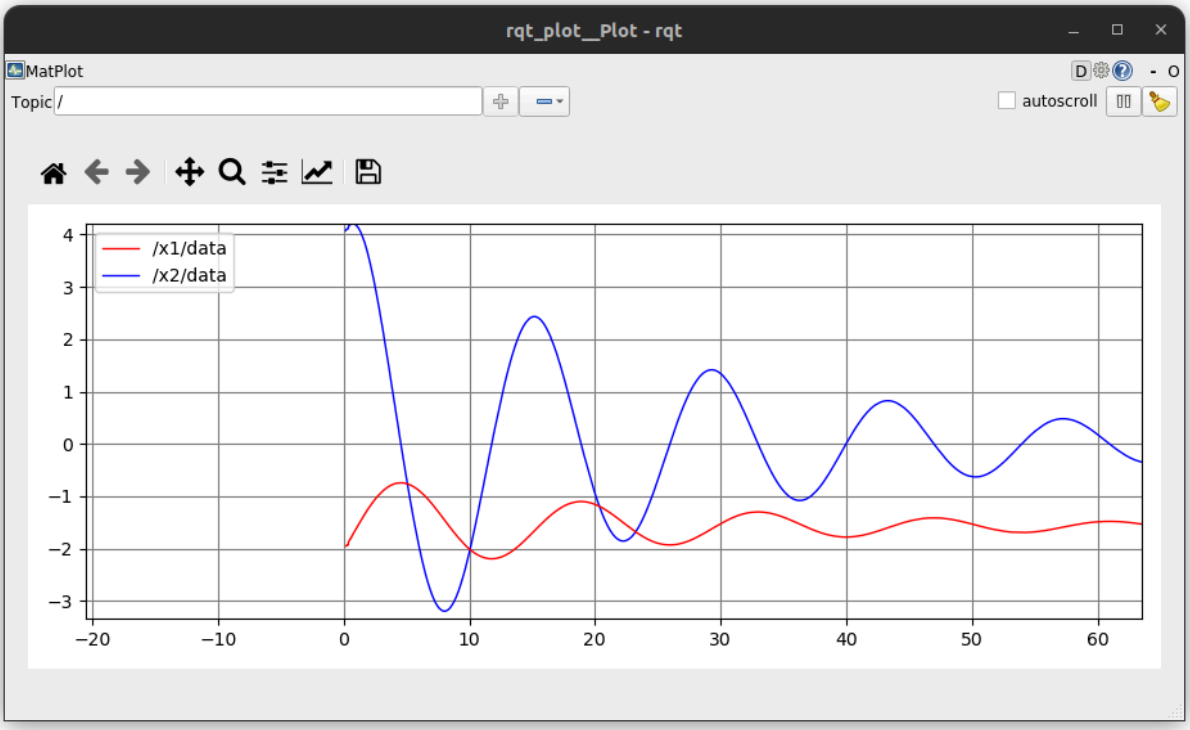
```python
#!/usr/bin/env python
import rospy
import numpy as np
import math
from std_msgs.msg import Float32
from sensor_msgs.msg import JointState

#Declare Variables to be used
k = 0.1
m = 0.75
l = 0.36
g = 9.8
Tau = 0.0
x1 = 0.0
x2 = 0.0
x2_dot = 0.0
dt = 0.001
J = (4/3) * m * pow(l,2)

# Setup Variables to be used
joints = JointState()

# Declare the input Message

# Declare the  process output message
def init_joints():
       joints.header.frame_id = "link1"
       joints.header.stamp = rospy.Time.now()
       joints.name.append("joint2")
       joints.position.append(0.0)
```

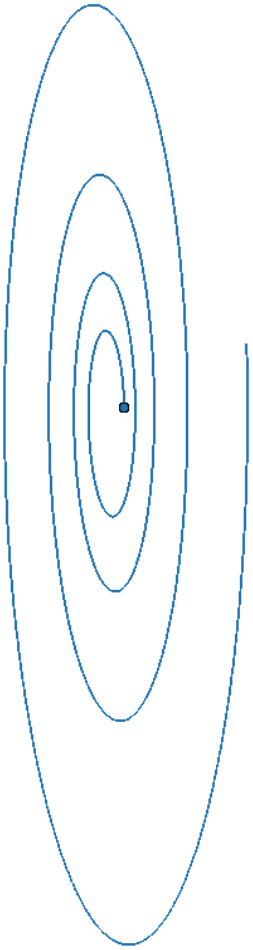
```
        joints.velocity.append(0.0)


#Define the callback functions
def callback(data):
        global Tau
        Tau = msg.data
        print(Tau)


  #wrap to pi function
def wrap_to_Pi(theta):
        result = np.fmod((theta + np.pi),(2 * np.pi))
        if(result < 0):
        result += 2 * np.pi
        return result - np.pi


if __name__=='__main__':
        #Initialise and Setup node
        rospy.init_node("joint_state_publisher")

        #Get Parameters

        # Configure the Node
        loop_rate = rospy.Rate(rospy.get_param("~node_rate",100))

        # Init Joints
        init_joints()

        # Setup the Subscribers
        rospy.Subscriber("/tau", Float32, callback)

        #Setup de publishers
        pub = rospy.Publisher('/joint_states', JointState, queue_size=1)
        pub_1 = rospy.Publisher('/x1', Float32, queue_size=1)
        pub_2 = rospy.Publisher('/x2', Float32, queue_size=1)

        print("The SLM sim is Running")
        try:
        #Run the node (YOUR CODE HERE)
        while not rospy.is_shutdown():
                #WRITE YOUR CODE HERE
                t = rospy.Time.now().to_sec()

                x2_dot = (1/J) * (-(m*g*l*math.cos(x1))-k*x2+Tau)
                x2 += x2_dot * dt
                x1 += x2 * dt
```

```
            joints.header.stamp = rospy.Time.now()
            joints.position = [x1]
            pub_1.publish(x1)
            joints.velocity = [x2]
            pub_2.publish(x2)

            # Publish
            pub.publish(joints)


            #Wait and repeat
            loop_rate.sleep()

    except rospy.ROSInterruptException:
    pass #Initialise and Setup node
```

Imágenes de las gráficas:


manipulator.rviz* - RViz
File Panels Help
Interact
Move Camera
Select
Focus Camera
Measure
2D Pose Estimate
2D Nav Goal
Publish Point
Time
Pause
Synchronization:
Off
ROS Time:
58926.80
ROS Elapsed:
118.30
Wall Time:
58926.83
Wall Elapsed:
118.27
Reset
Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click/Mouse Wheel:: Zoom. Shift: More options.
31 fps


rqt_plot__Plot - rqt
MatPlot
Topic /
autoscroll
/x1/data
/x2/data
4
3
2
1
0
−1
−2
−3
−20
−10
0
10
20
30
40
50
60



Video evidencia:
https://drive.google.com/file/d/1gaeAmuH1HzYy0F8OKZt85evUmWTajAuf/view?usp=sharing