Problem 1: What message type did you choose for these messages? Include your updated code in your submission.

Message type chosen: Float64

Code:

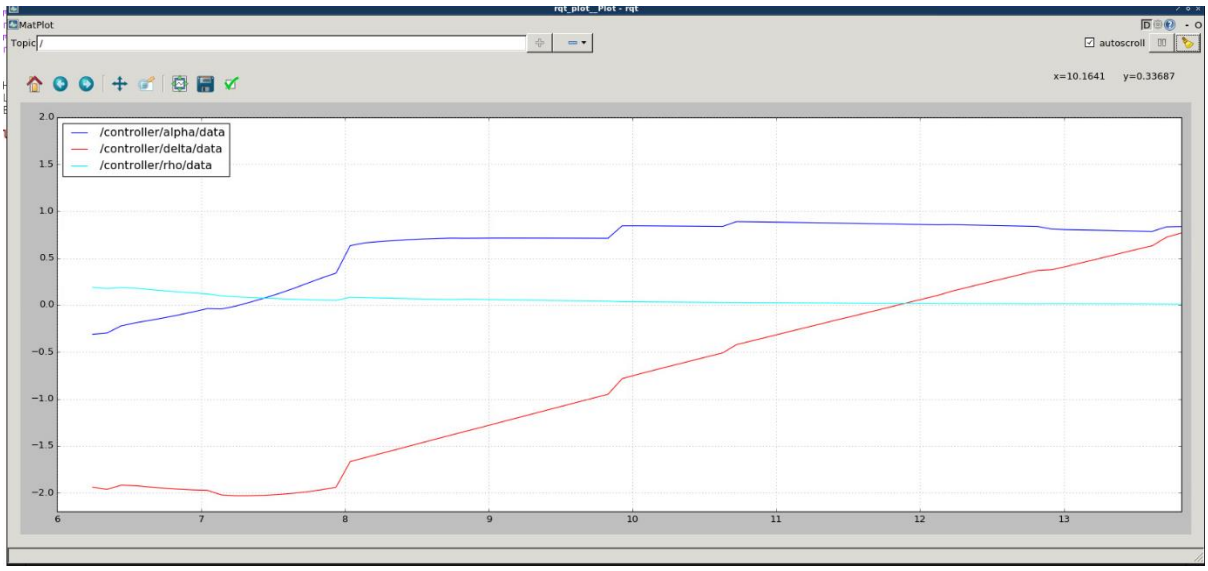
```
import numpy as np
import math
from utils import wrapToPi
import rospy
from std_msgs.msg import Float64

# command zero velocities once we are this close to the goal
RHO_THRES = 0.05
ALPHA_THRES = 0.1
DELTA_THRES = 0.1

class PoseController:
    """ Pose stabilization controller """
    def __init__(self, k1, k2, k3, V_max=0.5, om_max=1):
        self.k1 = k1
        self.k2 = k2
        self.k3 = k3

        self.V_max = V_max
        self.om_max = om_max
        self.init_publisher()

    ################################Section 6 Publisher#########################
######
    def init_publisher(self):
        # rospy.init_node('controller',anonymous=True)
        self.pub_alpha = rospy.Publisher('controller/alpha',Float64,queue_size
=10)
        self.pub_delta = rospy.Publisher('controller/delta',Float64,queue_size
=10)
        self.pub_rho   = rospy.Publisher('controller/rho',  Float64,queue_size
=10)

        self.alpha_msg = Float64()
        self.delta_msg = Float64()
        self.rho_msg   = Float64()
        # rate = rospy.Rate(1)

    ##############################End Section 6 Publisher#####################
######

    def load_goal(self, x_g, y_g, th_g):
        """ Loads in a new goal position """
```

```
        self.x_g = x_g
        self.y_g = y_g
        self.th_g = th_g

    def compute_control(self, x, y, th, t):
        """
        Inputs:
            x,y,th: Current state
            t: Current time (you shouldn't need to use this)
        Outputs:
            V, om: Control actions

        Hints: You'll need to use the wrapToPi function. The np.sinc function
        may also be useful, look up its documentation
        """
        ########## Code starts here ##########

        rho   = np.sqrt((x-self.x_g)**2+(y-self.y_g)**2)
        alpha = wrapToPi(np.arctan2((self.y_g-y),(self.x_g-x)) - th)
        delta = wrapToPi(alpha+th-self.th_g)

        V     = self.k1*rho*np.cos(alpha)
        om    = self.k2*alpha + self.k1*np.sinc(alpha/np.pi)*np.cos(alpha)*(al
pha+self.k3*delta)

        self.alpha_msg.data , self.delta_msg.data , self.rho_msg.data = [alpha
,delta,rho]
        self.pub_alpha.publish(self.alpha_msg)
        self.pub_delta.publish(self.delta_msg)
        self.pub_rho.publish(self.rho_msg)

        ########## Code ends here ##########

        # apply control limits
        V = np.clip(V, -self.V_max, self.V_max)
        om = np.clip(om, -self.om_max, self.om_max)

        return V, om
```

Problem 2: What command did you use to record the requested topics to a particular _le
name?

rosbag record -o Section6

Problem 3: Take a screenshot of the resulting plot in rqt plot and include it in your submission.


rqt_plot__Plot - rqt
MatPlot
Topic /
autoscroll
x=10.1641 y=0.33687
/controller/alpha/data
/controller/delta/data
/controller/rho/data
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
6
7
8
9
10
11
12
13