SC 649
Assignment 6
Name: Abeer Mishra
Roll no: 200260001

Q1. Set a constant linear velocity and an angular velocity to drive the robot. (*Select appropriate values so that the robot is able to complete a circular trajectory in a reasonable amount of time*)

The following code implemented in the “controller.py” script sets a constant linear velocity of 0.5 m/s and an angular velocity of 0.5 rad/s to drive the robot.

```
def main():
    rospy.init_node("controller",anonymous=True)
    pub = rospy.Publisher("/cmd_vel",Twist,queue_size=10)
    vel = Twist()
    vel.linear.x,vel.linear.y, vel.linear.z = 0.5,0,0
    vel.angular.z = 0,0,0
    rate = rospy.Rate(10)

    while not (rospy.is_shutdown()):
        vel.linear.x = 0.5
        vel.angular.x,vel.angular.y,vel.angular.z = 0,0,0.5
        pub.publish(vel)
    rate.sleep()


if __name__ == "__main__":
    try:
        main()
    except rospy.ROSInterruptException:
        pass
```

Q2. Plot the robot’s actual positions (x, y) (obtained from the robot pose data topic) versus the estimated positions (x̂, ŷ) obtained using the triangulation algorithm.

We calculate the robot’s position using the triangulation algorithm (with known orientation using the following code:

```
# Triangulation (with known orientation) implemented using landmarks A and
B
       Robot_Position_x1 = (state_y2 - state_y1 + state_x1*np.tan(thetaA)
- state_x2*np.tan(thetaB))/(np.tan(thetaA) - np.tan(thetaB))
       Robot_Position_y1 = (state_y2*np.tan(thetaA) -
state_y1*np.tan(thetaB) + (state_x1 -
state_x2)*np.tan(thetaA)*np.tan(thetaB))/(np.tan(thetaA) - np.tan(thetaB))
```

Here, thetaA and thetaB are the angles wrt x axis made by the line joining the robot and the landmarks A and B respectively. thetaA and thetaB are calculated as follows:

```
   thetaA = np.pi/180*(theta + state_b1)
   thetaB = np.pi/180*(theta + state_b2)
   thetaC = np.pi/180*(theta + state_b3)
```

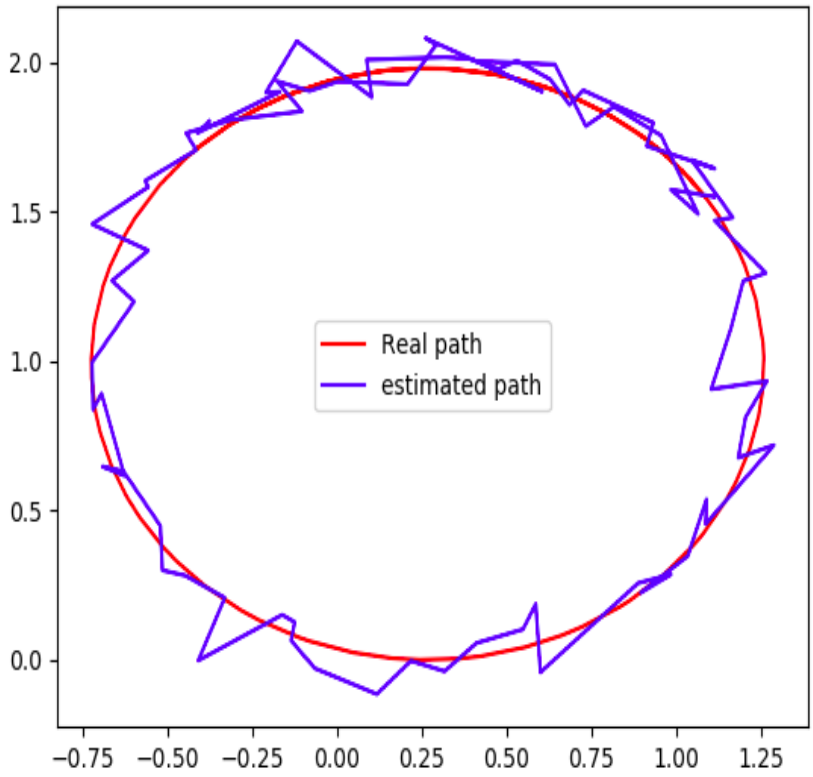
Here, theta is the known orientation of the robot wrt the x axis and state_b1, state_b2 and state_b3 are the bearing measurements made wrt landmarks A, B and C respectively.

The following part of code in “Robot_Position_Finder.py” script was used to plot the trilateration-estimated and the real path followed by the robot.

```
# Neglect the first 3 points in the trajectory as they are not part of the
circle
       if count > 3:
           realX.append(actual_x)
           realY.append(actual_y)
           estX.append(Robot_Position_x)
           estY.append(Robot_Position_y)

       # Collect the next 80 points and then display them
       if len(realX) > 80:
           plt.plot(realX,realY,"r",label="Real path")
           plt.plot(estX,estY,"b",label="estimated path")
           plt.legend()
           plt.show()
```

We obtain the following plot:



This was obtained using landmarks A and B.

Q3. Provide your reasoning for the mismatches in the actual versus estimated robot positions obtained using the triangulation algorithm. Are the position estimates in some portions of the robot's trajectory consistently worse than in the other portions? If so, provide your reasoning for the same. (Use annotations in the plots and explain)

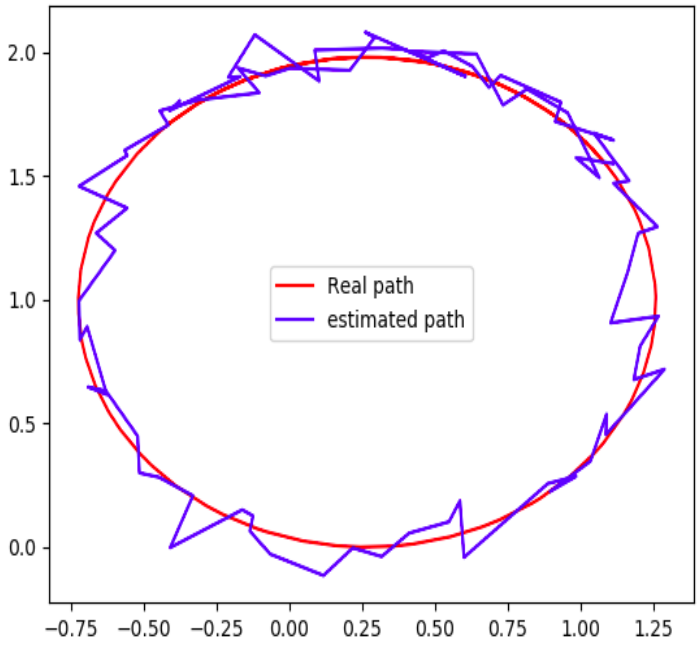
The mismatch is due to the Gaussian noise that we introduce in the bearing measurements in the code.

Q4. Obtain bearing measurements from all the following combinations - (landmarkA, landmarkB), (land-markB, landmarkC), (landmarkC, landmarkA), and obtain the robot's position estimates using each ofthe same. Compare their accuracy with the actual robot positions obtained from the robot pose data topic. Use mean squared error (MSE) as a performance metric.

Obtaining bearing measurements from which of the landmark pairs resulted in the best possible estimates? Provide your reasoning.

Using landmarks A and B:

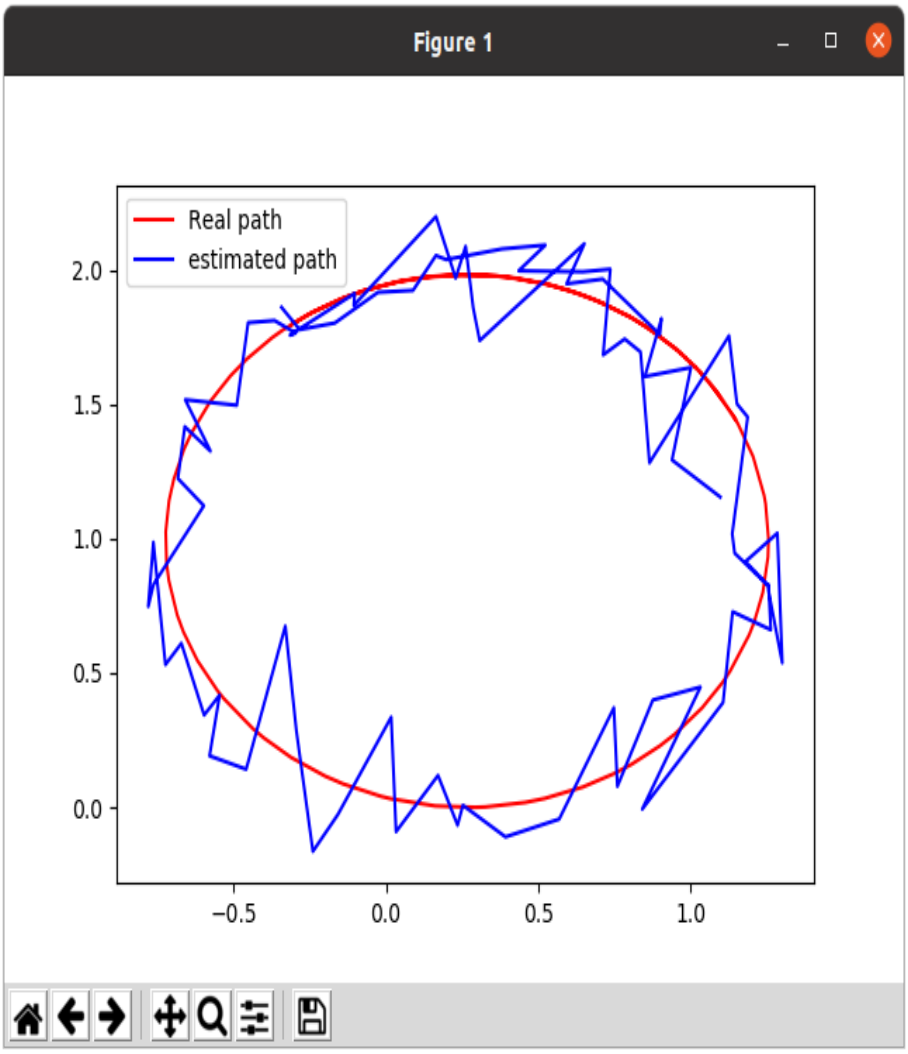
Plot of the estimated path



Mean squared error: 0.012369619149007515

Using landmarks B and C:

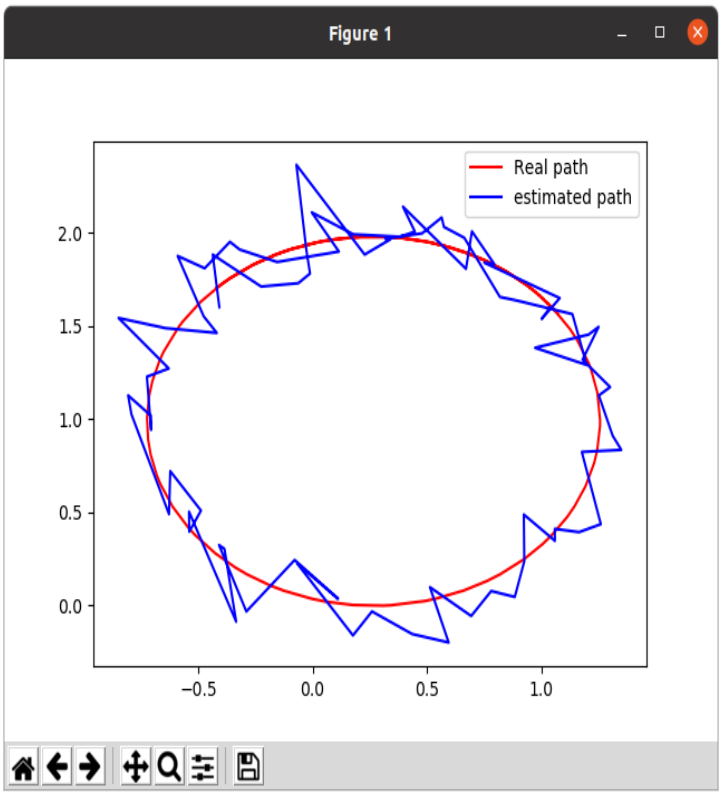
Plot of estimated path



Mean squared error: 0.029190984791727347

Using landmarks A and C:

Plot of estimated path



Mean squared error: 0.024439915526578836

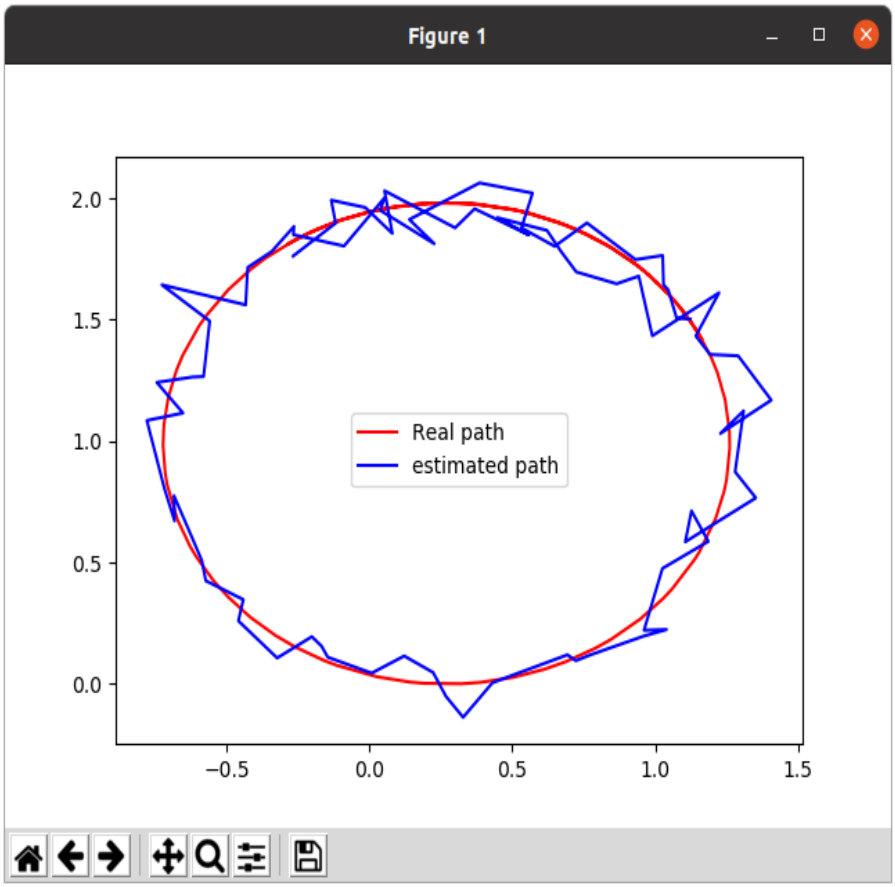
Landmark A and B seem to provide the best bearing estimates. This seems to be because they are placed symmetrically wrt the circle's position.

Q5. Come up with a strategy to combine the position estimates obtained from each of the three individual landmark pairs to obtain an enhanced position estimate using the triangulation algorithm. Show the improvement in accuracy using the MSE metric defined above. Provide your reasoning for the enhancement in accuracy.

We can average over the measurements obtained from each of the above position estimates to obtain a better estimate of the robot's position.

```
Robot_Position_x,Robot_Position_y = (Robot_Position_x1 + Robot_Position_x2
+ Robot_Position_x3)/3,(Robot_Position_y1 + Robot_Position_y2 +
Robot_Position_y3)/3
```

Plot of estimated path



Mean squared error: 0.00912648700590708

Clearly, this strategy reduces the mean squared error.
This happens as by averaging over different readings we can cancel out the errors in the individual readings.