AIRO2

Assignment 2 Report

**Group Members:**

*Yara Abdelmottaleb*

*Omotoye Adekoya*

*Amanzhol Raisov*

*Farzin Sarvari*

*Christelle Lekeufack*

In this report, we will indicate what we have changed in the visits\_module code. These changes were made to add the functionality of computing both the Euclidean distance cost and the uncertainty in localization when going from one region to another.

We have modified the regions’ waypoints and the landmarks as follows:

* 5 regions (r0, r1, .., r4) with way points as in the figure below.
* 2 landmarks (in red) that are located along the path from r0 to r4, and from r2 to r3.

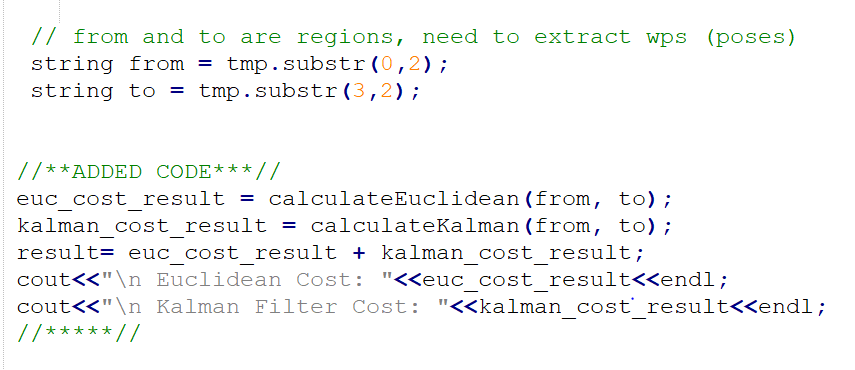
Chart

Description automatically generated

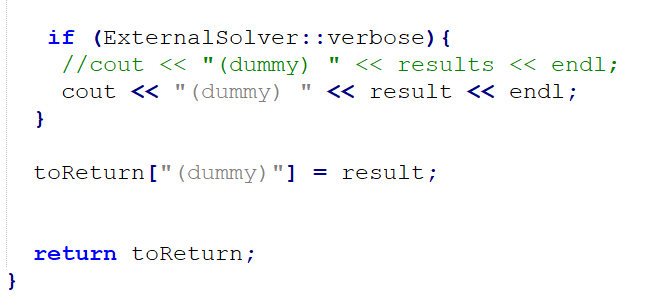
The domain file was not changed. The same go\_to\_region action was kept as it is without adding another action. All the changes were made inside visit\_solver.cpp and visit\_solver.h to modify the cost value of going from one region to another returned to the dummy variable.

Inside VisitSolver::callExternalSolver() function:

Both the Euclidean distance cost and the uncertainty in localization cost are computed and added.

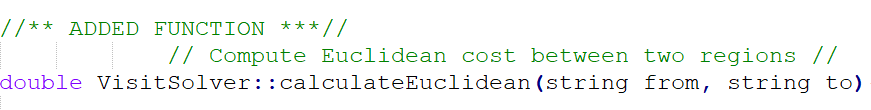


Then, returned to the dummy variable.

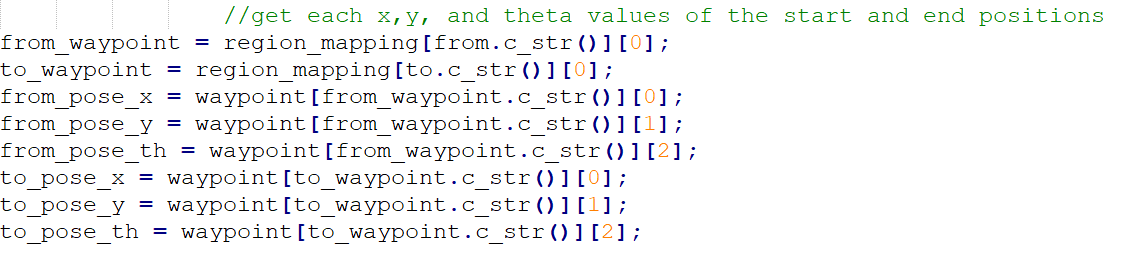


**Euclidean Distance Cost:**

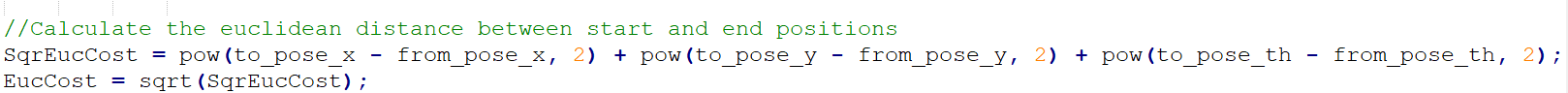
This function takes as input the start and goal regions.



Then, the x, y, and theta values are extracted for each region



Then, the Euclidean distance is computed as

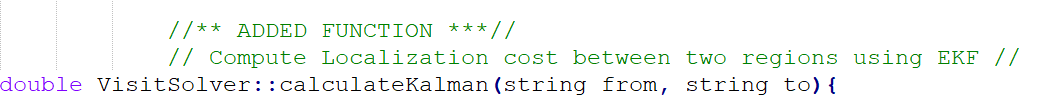


And returned.

**Localization Cost:**

“Eigen” library was used to facilitate matrix calculations in C++

This function also takes as input start and goal regions



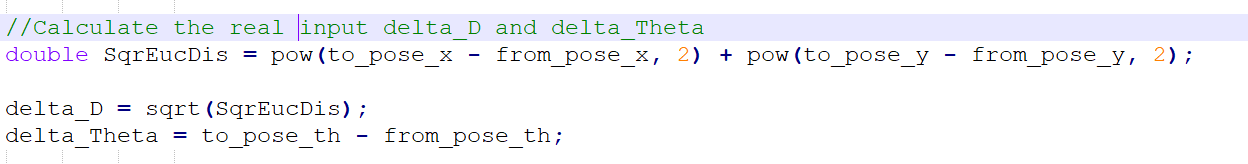
We assume that the robot’s odometry model as follows:

The error added to the input is an indication of the low resolution of the encoder. The subscription “real” means the real input value delivered to the robot, and the subscription “measured” means the values that encoder read.

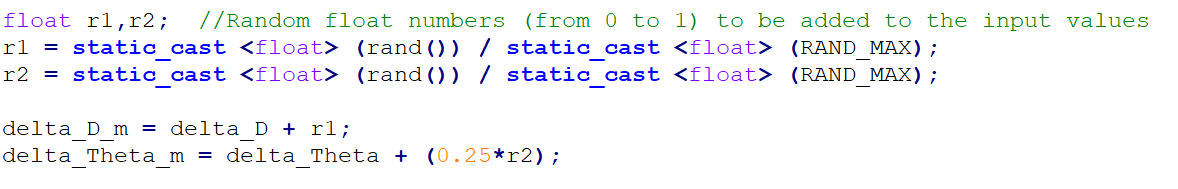
Text, letter

Description automatically generated

In the code, delta\_D and delta\_Theta are computed based on the from and to region’s poses.



After that, random values are added to them to indicate the measured input values by the encoder.



The state vector X, the input vector U, and the observation matrix as follows:

The subscription *“l”* means landmark

Text, letter

Description automatically generated

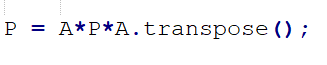
Prediction Phase:

We predict the state in the next time instant based on the odometry equations. Then, we compute the A matrix.

A picture containing letter

Description automatically generated

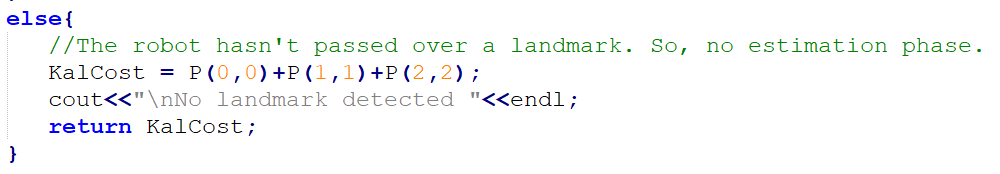
And the P matrix is updated based on the odometry prediction.



Estimation Phase:

After that, we check if the robot has passed over a landmark in its way from the start to goal region. This is done by checking if the start, goal, and landmark points are on the same line.

If they are not, return the cost of the current P matrix (the trace of it)



If yes, compute the C matrix.

A picture containing schematic

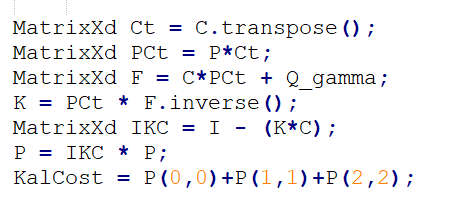
Description automatically generated

And the Q\_gamma (assuming white gaussian noise from -1 to 1).

Text, letter

Description automatically generated

Then, update the P matrix (uncertainty in robot’s location) based on the estimation phase, and get the trace of this matrix as the cost of localization.



Return this cost.

The output result indicates a higher Kalman Filter cost in the paths that does not contain landmarks (for example, r2 to r1). With a constant preference to the plan: r0 to r4, then r4 to r3, then r3 to r2, then r2 to r1.