Technical report

this is a technical report to overview the technecalities we worked through in this project

* main conrner stones

1. Astar code
2. RRT code
3. Ros and gazebo

Astar code

the astar code we used wasn't developed entirly by us..we used an online available code

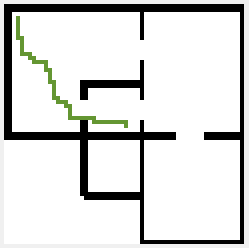
<https://github.com/justinhj/astar-algorithm-cpp>

the code didn't allow for branching factor more than four couldn't take pictures as an input

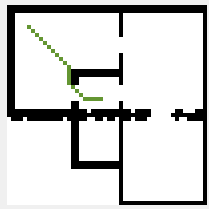
we modifed the code by

1-adding image as input method

2-increasing branching factor to 8 as catbot robot ofcours can move diagonlly

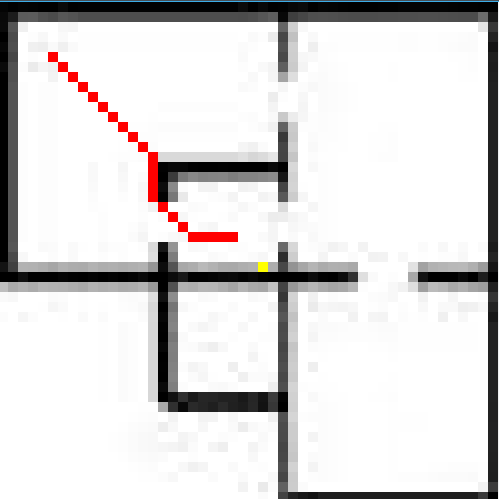
  
befor modification

(matlab visualizatin)

After modification

(matlab visualizatin)

3- add visualization option usingn in c++



4-we also add binary file as input type as the image library in use (image magick/magick++)

can read only 500 pixel value per second (prossesor dependant) but a binary input takes no time to be reade

- we added c++ code (imm2arr) to make it easyer to convert any image into binary array

5- we modified the code to give it's output as text file containg positions of pixels of the path to make it easier to deal with later

6-edge detection code to find edges

7-we added interactive visualisatino option using matlab (show) matlab file

RRT CODE

we developed the rrt code ourselves

main steps of the code :

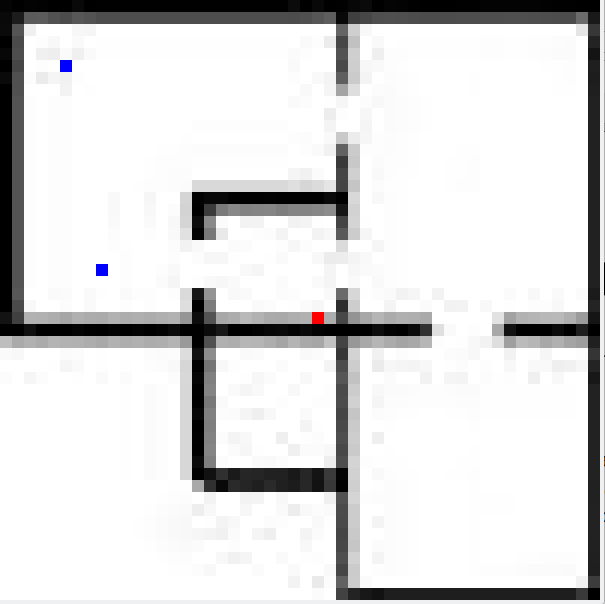
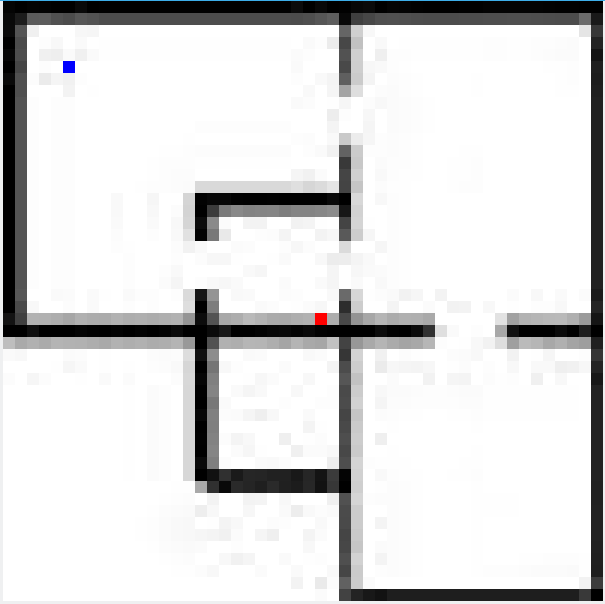
1-input type:

like astar code the rrt takes both jpg image or binary file input

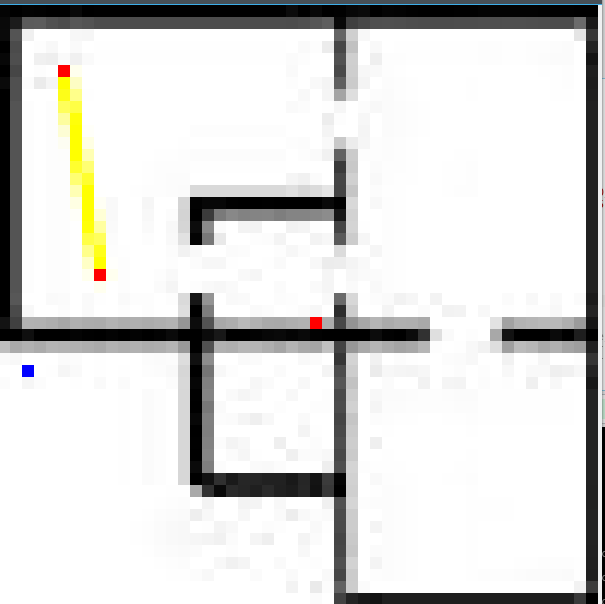
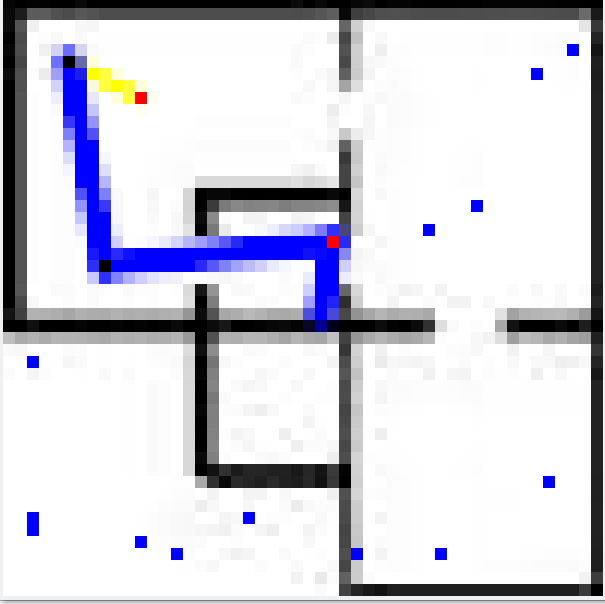
2-visualization:

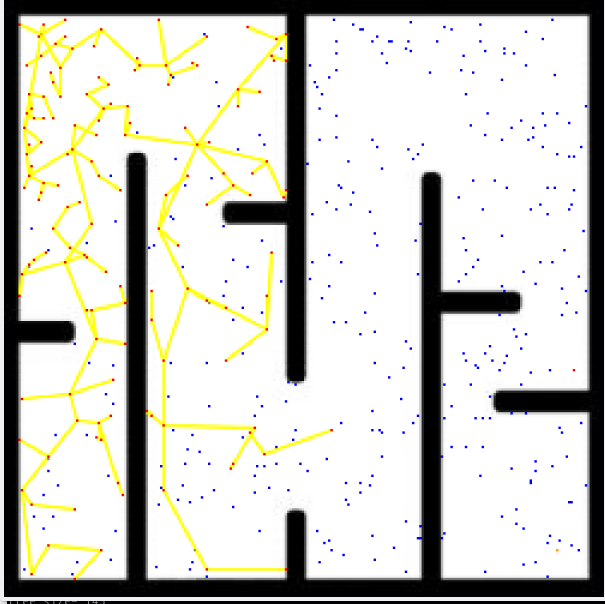
the code allow for visualizing the main steps of building the tree proccess

-show start and goal -suggesting new points



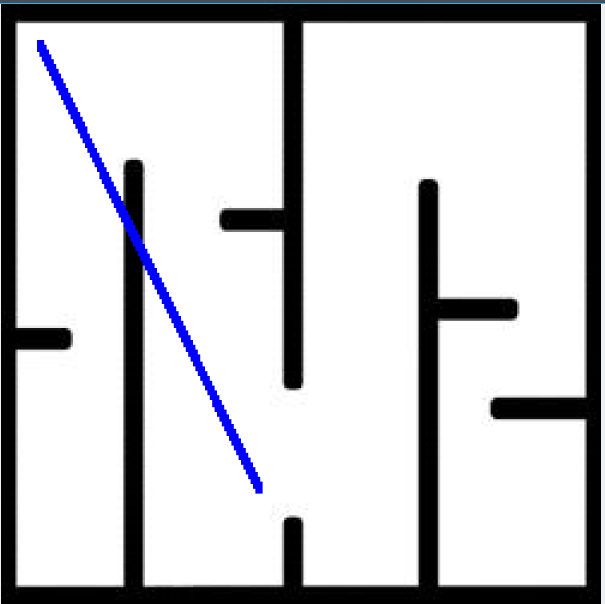
-adding points to the tree - reaching the goal



  
half built tree with threshold option activated

2-check path funciont :

a fundemental function used to check if the path between two points in the map is free or not



-it returns zero in case the bath between the two passed points

has an obstacle

-it has (robot width parabter ) whick allow for detemining the width of the line drawn

(this is our way to make sure the robot is far from the walls

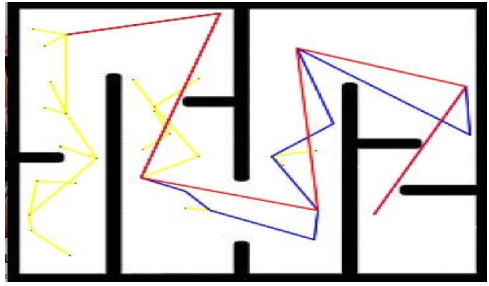
by suitable margin

3-threshold variable:

-its the farest distance a point can be from the tree to be add to the tree

4-extended path

-the path found by rrt is usually not optimum and contains a lot of redundency after calculating the main path we find the new optimum path

  
main path (in blue) and extended path (red)

ROS and GAZEBO

the final part is the implimentation of the code in ros to run a simulation of the catbot runing in obstacled environment using both of the algorethms

-we wrote two ros nodes (astar) node and (rrt) node

-the ros nodes read the files produced by the algorethms and try to reache them accurately

-to reached the goals accurately we developed a **controller** for the catbot tha does the following:

* subscribes to gazebo model state topic to find it's exact positoin.
* takes an order position from either rrt or astar code.
* calculates the angel it has to rotate and apply it with acceptable margin of error.
* calculates the distace to travel and apply it.
* calculates the error in angle along the traveled distance and corrects any error.

Ps:

Videos of the simulation is attached: