University of Girona

Spain

Autonomous Robotics
Lab 2 - Sampling based Algorithms - Rapidly-Exploring Random Trees(RRT)

Submitted by :
Nayee Muddin Khan DOUSAI

1 Introduction

In this lab, we are going to implement and test Rapidly-Exploring Random Trees (RRT) algorithm, sampling algorithm mostly used for path planning in robotics. Path planning is based on connecting the samples from starting to the goal point. The algorithm is coded in MATLAB and the results are attached in .m file and pictures labelled in the report. This lab work is divided into two parts, one is to find the path and then smoothing for the given two environments.

2 RRT

2.1 Path Planning

Path planning algorithm for RRt has implemented and tested in MATLAB. The instructions and the rules to follow are completely given in the instruction manual. The aim of this algorithm is to reach to q_goal position. The steps are followed as:

- The maps are loaded by .mat files from the given file
- q_start and q_end are given for both the environments
- q_start is the starting position and q_end is the end position of the environment with their respective coordinates
- delta_q can be calculated by the distance q_new and q_near
- q_new is the generated at delta_q distance
- Initialize the vertices with q_start and edges as empty
- Find the probability with q_rand function
- If the q_new belongs to the free space we add: q_new to vertices and check if q_new is equal to q_goal then we will the path and break RRT
- path follows the connection of the points and gives the path of algorithm

2.2 Smoothing

Once we find the path planning for both the given environments, we will implement smoothing based on below steps for the better and less noisy path. The steps are explained as below:

- For every iteration, at first q_start is always q_start is always set to the start point

- Find the unit vector from the q_start to q_end
- check if it is free space between q_start and q_end (add delta*unit vector)
- If agree, directly set q_end to q_start, put the q_end into path_smooth and this iteration finishes
- If no, update q_start to the next element in the path list, start from second point again

3 Results

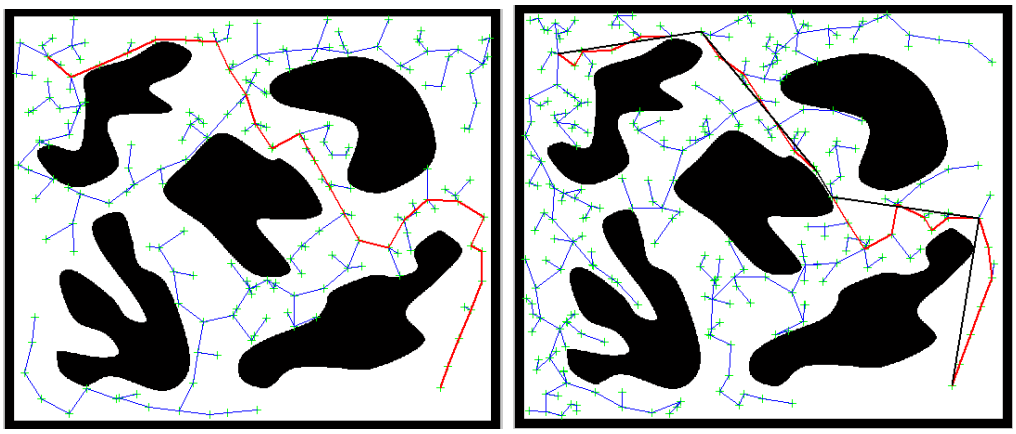

Figure 1: Path planning and smoothing for map

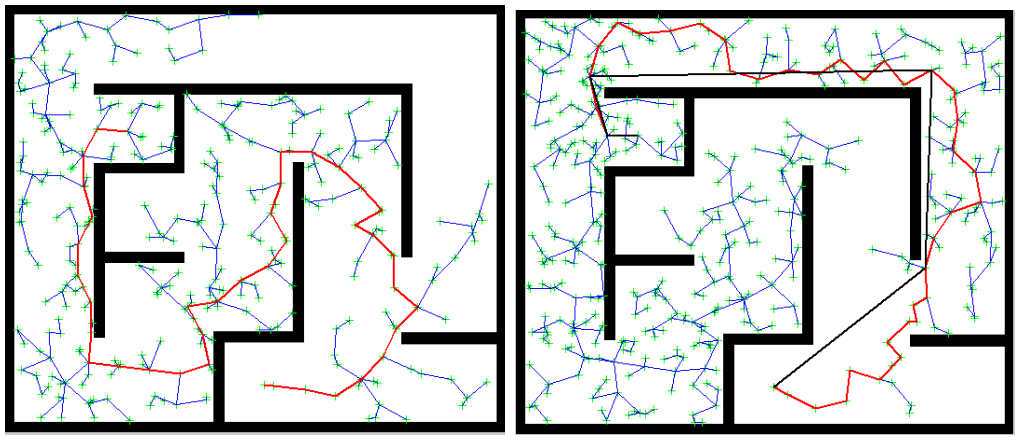

Figure 2: Path planning and smoothing for maze

4 Conclusion

In this lab, we have implemented RRT path planning algorithm and then we having given smoothing algorithm. The results are tested by different iterations and changing the probability values.