SC 627 - Motion Planning and Coordination for Autonomous Vehicles
Assignment I: Bug 1 Algorithm

Gagan Jain | 180100043

Implementation Methodology

My implementation primarily relies on the definition of the potential fields, as given in the problem statement. The attractive potential gradient is easy to compute and is defined on the basis of the proximity threshold to the goal that has been set up. The repulsive potential gradient requires the knowledge of the normal to the obstacle at the current position which has been added as a functionality to the helper file created for assignment 1. Now, for each obstacle, the repulsive potential gradient is computed based on its proximity. The gradients are then summed up to provide the overall potential.

While this should have been the end of the story, the planner gets stuck using this gradient as its velocity direction. This might happen in one out of two ways. It can reach a local minima, other than the goal position, and thus get stuck over there with a zero gradient. For this reason, the planner cannot have a zero gradient as a termination condition. So instead I have retained the distance based termination condition which will only turn true if the bot reaches close to the goal. The other way in which the bot can get stuck is by having an oscillating behaviour. It might repeatedly visit two or more points with gradients pointing in each other's direction, and thus never get out of the periodic loop. This is the reason why the bot was not able to reach the goal position without any tweaks, even in pure Python which had no environment randomness.

To counter this, I have added a shoot functionality with a patience level of 3. In case the bot is very close to a point from its previous 3 timesteps, it invokes the shoot function, which just blindly makes the bot move in the direction of the goal for next 3 timesteps without worrying about the potential values or the obstacle locations. While this might have the danger of hitting an obstacle, it gets us out of the loop and allows us to reach the goal point. A more studied implementation might also be able to ensure that obstacles are never hit, but since this was anyway beyond the scope of this assignment, I have kept myself restricted to the basic idea that leads us to the goal point under certain assumptions regarding the obstacle distances from shoot points.

Simulation results

The given test case produced the following results upon simulation of the Potential Field Planner Algorithm:

Result of the motion planning algorithm: Success
Time taken by the algorithm is: 560.3965628147125 seconds

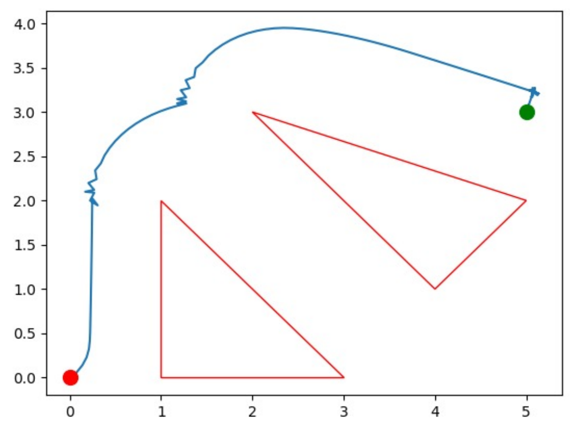

Figure 1: Path visualization (Python)

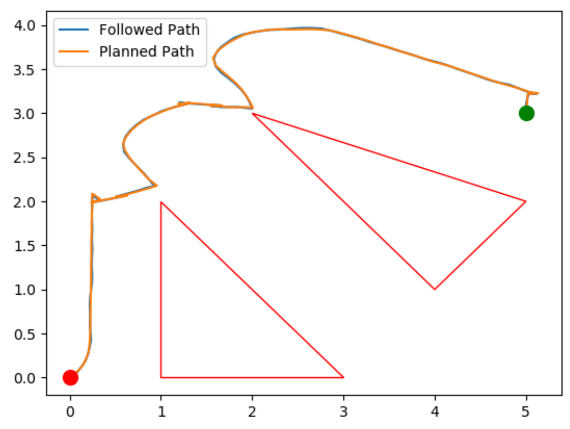

Figure 2: Path visualization (ROS)