ROBOTICS: PLANNING AND NAVIGATION ASSIGNMENT 3 REPORT

DYNAMIC VELOCITY-OBSTACLE FORMULATION

Team - no
Haripraveen Subramanian - 2018102031
Loay Rashid - 2018102008

Introduction and Theory:

A Velocity obstacle (VO) refers to the set of all velocities of a robot which will result in collision with another robot moving in the same space. This formulation is used in obstacle avoidance such that we make sure that the robot(s) choose(s) a velocity which does not lie in VO thereby, avoiding collision.

There are two methods to solve this problem -

- By framing it as a sampling problem.
- By framing it as an optimization problem.

We took the sampling approach. At each time step, we look at all the possible velocities which the robot may take, subject to the constraint that the chosen velocity lies between a user-specified range of minimum and maximum velocities (vmin and vmax respectively). Then, from among these velocities we look at all the velocities which do not lead to a collision. And then from among the set of velocities which lie between vmin and vmax and also don't lead to a collision, we choose the velocity which minimizes the distance between robots present position and its goal.

Note: For the purpose of this assignment, we have framed the problem as a sampling problem, and have solved it for the case of a holonomic robot.

We run this code for several iterations.We have a limit on the total number of iterations that we allow for the algorithm to find a path . If it fails to do so we exit to avoid going into an infinite loop.This algorithm is followed in each iteration:

- For all robots-
  - Get a set of velocities which lie in range v_min and v_max.
  - Now, from among these sets of velocities, further Iter to get the set of velocities which do not result in a collision.
  - Once the set of velocities which do not result in a collision have been identified, from.
- These velocities, choose the velocity which minimizes the distance between robots present location and its goal.

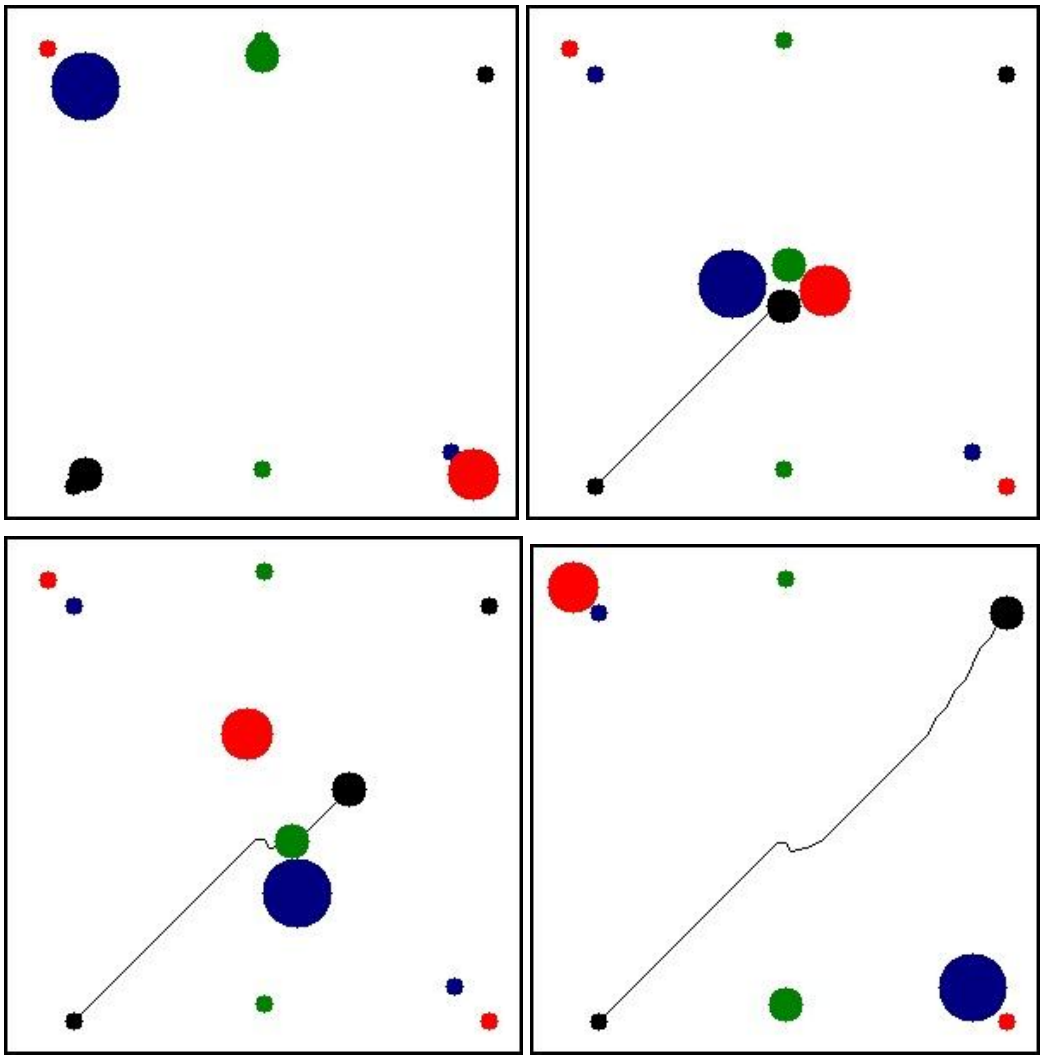
- Keep doing the above process for each robot, till the robot is within a threshold distance of its goal.

Code Outline

1. Create an empty image. This empty image represents our maze.
2. Create a set of robots. The dynamic obstacles too have been modelled as robots and our actual robot tries to avoid collision with these moving robots.
3. Define the parameters to those robots.
4. Once the robot parameters have been added, they are added to our world by calling the get_maze(). Once all the parameters have been created, we go to the solver() function. The get_path() function is the main function which solves the problem

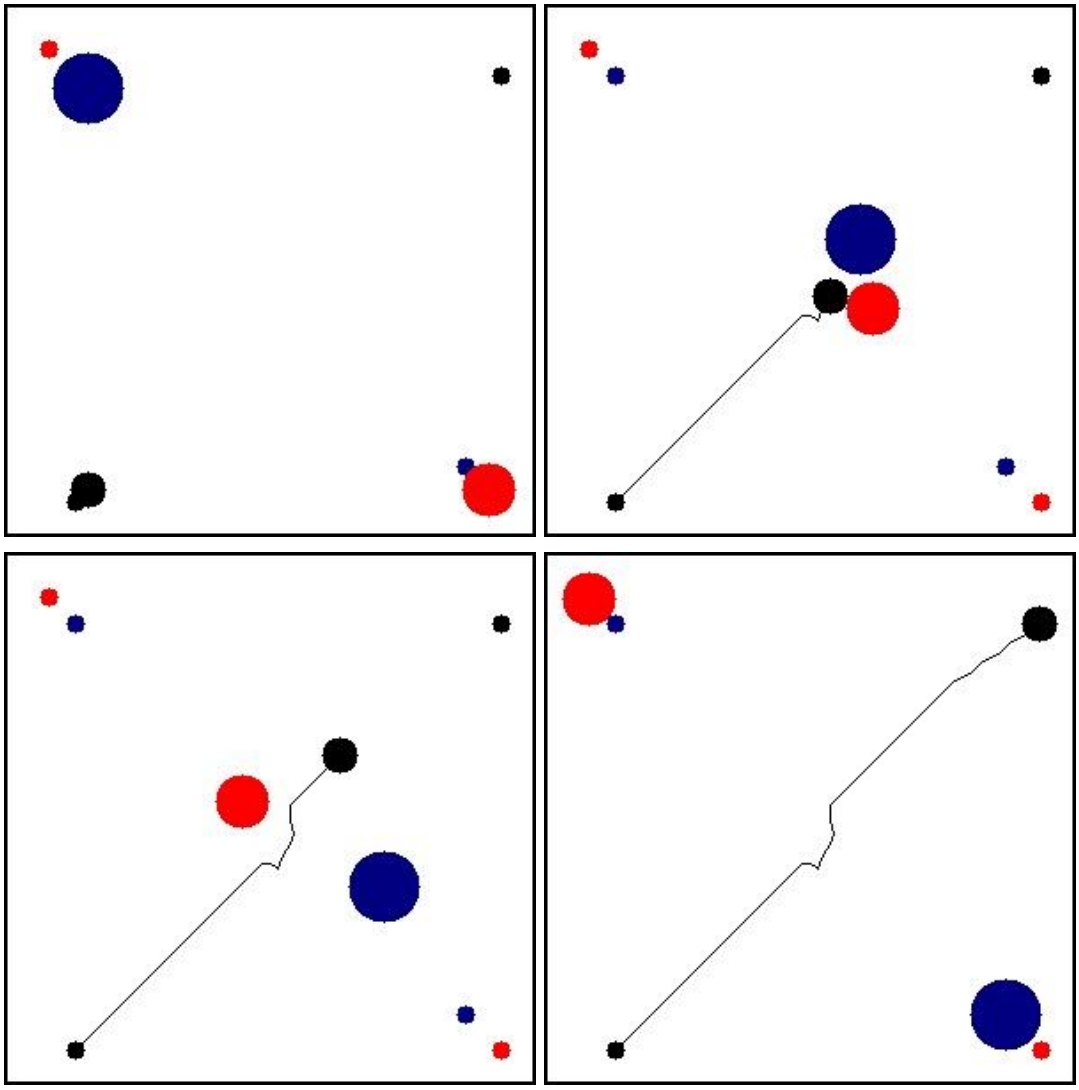
Example Outputs:

1 robot, 3 moving obstacles:



https://drive.google.com/file/d/1ijDOiNTmMG2WlMwJhoXdy5HDC5g8YmVg/view?usp=sharing

1 robot, 3 moving obstacles:



https://drive.google.com/file/d/1WaZRq__s8IG2QhRELKj7cDT4OQ0wZUnK/view?usp=sharing