COMP550 - Project 5

Anja Conev - ac121
Dejan Grubisic - dx4

November 30, 2019

1. **The problem statement and a short motivation for why solving this problem is useful.**

   We have paired up and decided to work on the problem of decentralized Multi-robot coordination. In this problem the goal is to find a decentralized algorithm for each agent that will lead all agents on their goal state, while avoiding collisions with other agents and obstacles. The main motivation behind this problem is the opportunity to achieve coordination of many agents that could be important for any application in the swarm robotics. On the other hand, this problem is particularly challenging because of potentially high number of agents, that make traditional centralized planning impossible.

2. **The details of your approach and an explanation of how/why this approach solves your problem.**

   We implemented the algorithm for Reciprocal Velocity Obstacles(RVO), based on paper [1]. The main idea in RVO algorithm is to iteratively calculate the velocities for each robot in each simulation step, so they stand outside of the RVO area. RVO area represents "dangerous area" for velocity vector of some robot and if the robot has velocity inside that area and keep that direction all the time, robots will collide eventually. Otherwise, we can be sure that there will be no collision. In the situation with many agents on narrow space this is not always possible and in that case we pick the velocity with minimal penalty based on intensity of velocity and expected time of the collision.

3. **A description of the experiments you conducted. Be precise about any assumptions you make on the robot or its environment.**

   In this project we chose python and matplotlib package for easy implementation and good visualization.

   For analysis purposes we implemented four scenarios. The first scenario consists from two robots that should switch position. In order to do this they go around each other based on RVO algorithm. Figure1 shows this situation with ploted triangles that represent RVO area.

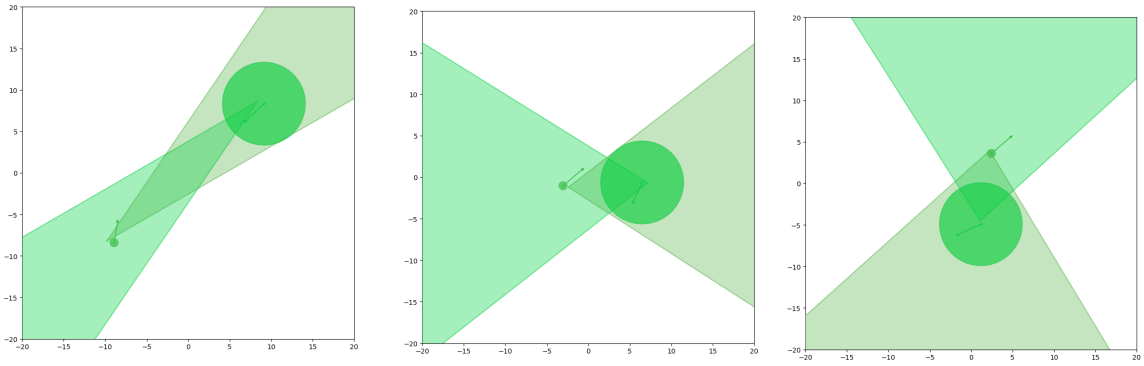

Figure 1: Scenario of circumvent of two robots. Circles-represent robots, rectangles-represent RVO, arrows-represent velocities

In the second scenario there are six robots in the circle and the goal for each of them is to come on the other side of circle. As the shortest path to opposite side for each robot goes through center of the circle, the density of robots near the center increase and the challenge is to find collision free path. On the Figure2 we show robots and their velocities. From the beginning (most left) it could be seen that robots choose velocities pointing not directly toward goal state, but rather slightly to the left, so it will be easier to avoid other robots in the later stages of simulation.

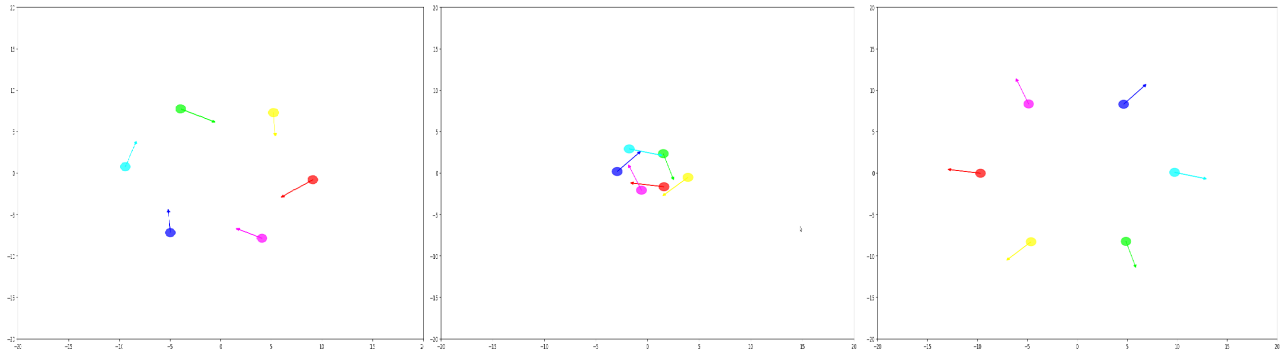

Figure 2: Scenario of 6 robots placed in circle.

In addition, this scenario is simulated for one hundred robots Figure3 in order to analyze scalability of the algorithm. In this case the simulation takes about 2 minutes, while in the case of 6 robot time is about 4 seconds. The implemented algorithm has quadratic complexity in worst case for a given number of robots, because for every robot, calculating RVO area requires calculation for every other robot. To mitigate this we are calculate RVO only for the robots closer than some defined constant. This approach lowers the execution time, but still in the worst case calculation could take $O(n^2)$ time.

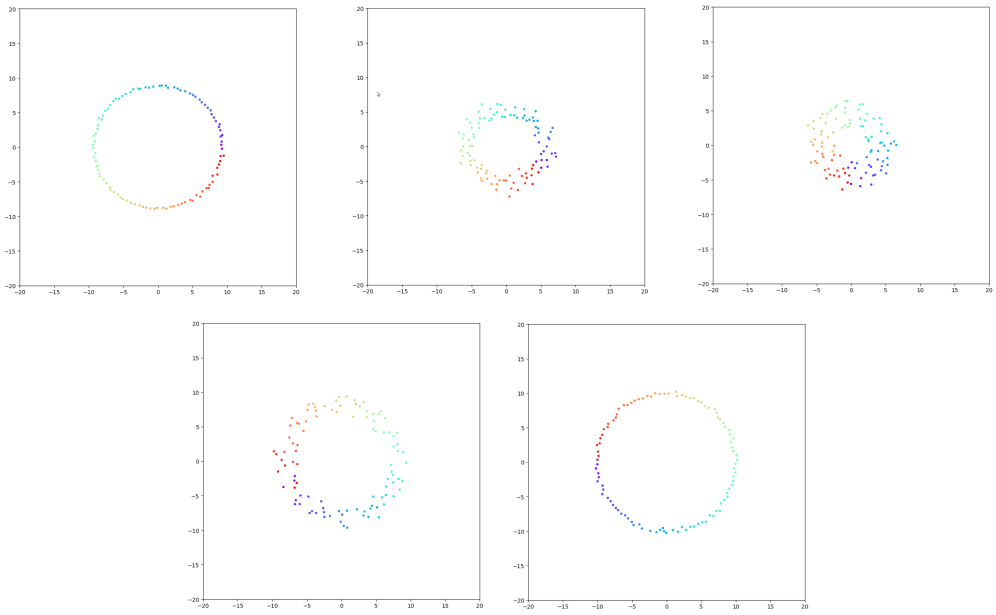

Figure 3: Scenario of 100 robots placed in circle.

In the third scenario Figure4 there are four groups of robots (in each corner of the space) that try to reach opposite corners, avoiding four square obstacles in the center of the map. In this situation, robots successfully avoids collisions and static obstacles and reach the goal.

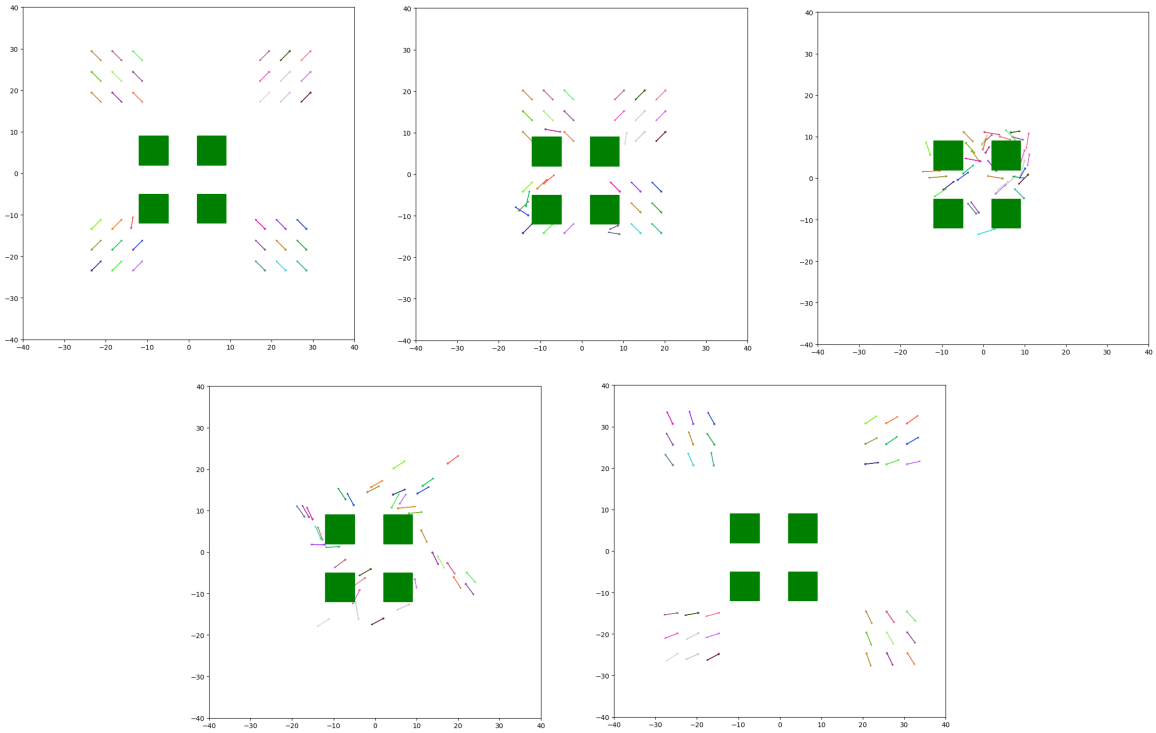

Figure 4: Scenario of 6 robots placed in circle.

In the last scenario all robots are aligned on the left side of the space and their goal is to reach the right side, while avoiding moving the obstacle that moves downwards in the center of the space Figure5. Robots calculate the trajectory of the moving obstacle and adjust their direction to that in addition to standard robot to robot interaction. Therefore from the second scene it could be seen that robots that would collide with moving obstacle, by just going straight, tend to go up so they could pass the obstacle from behind.Additionally, in Figure5 we showed the RVO for green robot.

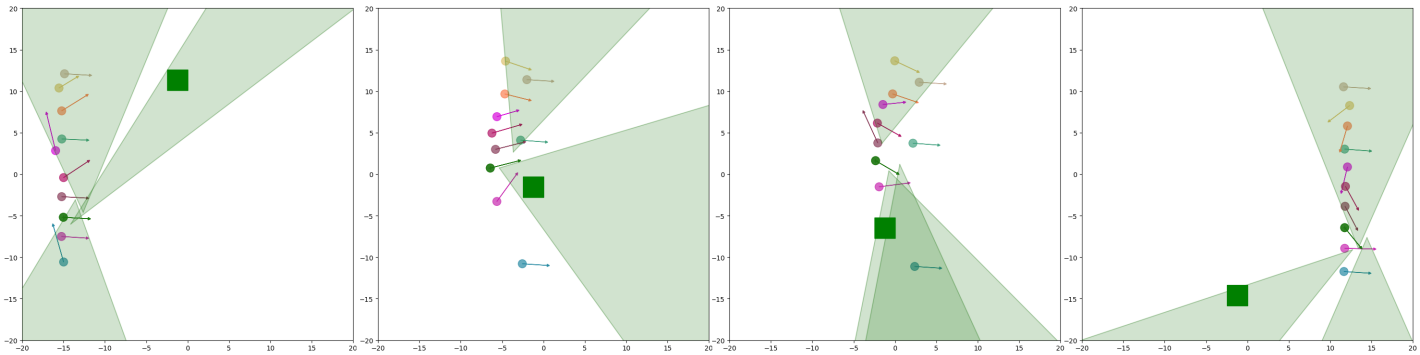

Figure 5: Scenario of 6 robots placed in circle.

4. **A quantitative and qualitative analysis of your approach using the experiments you conduct.**

   (a) **Is your method always successful in finding valid paths?** Our implementation has found valid paths for majority of situation that we simulated. Nevertheless, in the situations with high density of robots on the small area, the algorithm is not always capable of finding collision free paths.

   (b) **Expound upon the kinds of environments and/or robot configurations that may be easy or difficult for your approach?** Implemented algorithm works great in the situations when the number of robots is not extremely high (like in example with 100 robots) and there is no narrow passages that will increase a density of robots on small area. In all other situations implemented approach gives quickly the accurate trajectory for each robot.

   (c) **How well does your method scale as you increase the number of robots?** As we previously explained, number of robots matter, especially in the situations when robots are close to one another, because in that case algorithm becomes quadratic.

   (d) **Qualitatively, how do the solution paths look?** The simulated trajectories has straightforward paths until the robot becomes close to the obstacle or another robot. In this situation robot chose alternative path to avoid collision. Nevertheless, solution paths seems reasonably short.

5. **Rate the difficulty of each exercise on a scale of 1–10 (1 being trivial, 10 being impossible). Give an estimate of how many hours you spent on each exercise, and detail what was the hardest part of the assignment.**

Project Exercises	Difficulty(1-10)	Time(h)	Problems
1. Implement RVO for two robots	6	5	We had no problems
2. Introduce neighbor region and penalty	4	3	We had no problems
3. Implement Circle-scenario	2	2	Scalability
4. Implement Square-scenario	2	2	Hitting obstacles (solved)
5. Implement Moving object-scenario	2	1	We had no problems

References

[1] J. van den Berg, M. Lin, and D. Manocha, Reciprocal Velocity Obstacles for Real-Time Multi-Agent Navigation. In IEEE Int'l Conference on Robotics and Automation, pp. 1928-1935, 2008. https://www.cs.unc.edu/geom/RVO/icra2008.pdf.