Collision Avoidance in Swarm Robots using ORCA (Optimal Reciprocal Collision Avoidance)

Nikhil Mehra[1] , Saket Seshadri Gudimetla[2] and Sayan Brahma[3]

Abstract— **This paper explains our approach of implementation of a global path planner algorithm along with ORCA(Optimal Reciprocal Collision Avoidance) for collision avoidance in Swarm Robots. Swarm robotics is all about controlling a large number of simple robots and in general any algorithm deployed to control these swarm bots are independent of the fact that how many number of robots are active on the platform. Our project is to implement ORCA algorithm for obstacle avoidance along with the use of any commonly available global planner algorithm such as A*. This implementation will take into consideration both dynamic and static obstacles present in the environment.**

I. INTRODUCTION

Exploration using a robot is a difficult task in itself, whether it may be on a known terrain or an unknown terrain. Known environments can include inside of a building or a warehouse etc. Unknown environments can include a forest area, extraterrestrial planetary surface etc. Exploration becomes much more difficult when there are unidentified obstacles and resources in the unexplored region. The harder part is not to differentiate between an obstacle and a resource, but to get around an obstacle without losing its path and efficiency. The process becomes more involved[9]. when along with static obstacles, dynamic obstacles are also at play. The only difference between exploring a known space from an unknown space is the availability of a map (a known environment with pre-defined constraints). Hence exploring an unknown environment becomes a complex job with all these intertwined involved processes. We mainly want to emulate the exploratory behavior of a swarm, preferably three to four, in an extraterrestrial environment. Exploration obviously includes obstacle avoidance from static obstacles and collision avoidance between the robots. It is assumed that all robots communicate with the server and not with each other, therefore each robot can be assumed to act independently[1]. A simple exploratory algorithm such as ray algorithm or spiral algorithm will be used. As robots can not directly communicate with each other we are using ORCA (Optimal Reciprocal Collision Avoidance) algorithm, which is specially designed for such systems. ORCA uses the concept of velocity obstacles (velocity space), where required conditions for the collision-free motion are derived using linear programming. Basically the problem is reduced into a low-dimensional linear program[9].

II. LOCAL PLANNERS

Local planners focus on updating the robot's trajectory based on the data coming from its sensors. The resulting robot motion is both a function of the robots current or recent sensor readings and its goal position and relative location to the goal position.The obstacle avoidance algorithms being discussed depend on a global map and the robot's current location relative to the that map(not ORCA). Diverse range of obstacle avoidance algorithms are discussed in the following sections.

1) Bug algorithm: This is one of the most simplest local planning algorithms. The idea here is that the robot must follow the contour of an obstacle. There are two types of Bug algorithms namely - Bug1 and Bug2.
In Bug1 the robot encircles the obstacle first, determines the best leave point from the obstacle boundary(one with the minimum distance to goal) and then sets off. Bug1's approach is not efficient but can assure robot will reach any reachable target or goal point.
Bug2 algorithm is an improved version of Bug1 in which the robot encircles the obstacle's boundary but does not do so entirely rather, departs immediately when it has a clear way towards the goal.

2) Vector Field Histogram: Borenstein together with Koren developed the Vector Field Histogram (VFH)[5].
The drawbacks of Bug algorithm is that it's output relied on the recent sensor's readings. This is an issue as in some cases the robot's latest sensor readings were not sufficient for performing obstacle avoidance. The VFH overcomes this issue by creating a local map. It uses the concept of occupancy grids that contain relatively recent sensor readings.
For performing obstacle avoidance it creates a polar histogram. The x-axis of which represents the angle at which the obstacle was found and the y-axis represents the probabilities that the obstacle was actually there based on the occupancy grid's values. Using the histogram the steering direction is determined. All the free spaces large enough for the robot to pass through are identified and then a cost is assigned to each of those. Ultimately the free passage with the lowest cost is chosen.

3) Probabilistic Velocity Obstacle(PVO) and Occupancy Grids: This approach consists of reactive obstacle avoidance in dynamic uncertain environment. The algorithm factors in the uncertainty generated due to noise coming from sensors as well as from an erroneous description of a model of the environment[8].
The algorithm relies on occupancy grids that are extremely sensitive to changes in the surroundings of the robot.
The algorithm works in the velocity space by computing the probabilities to collision. Kinematic and dynamic constraints are also considered and they help in determining the reachable velocities.

Based on the time to collision a navigation algorithm is constructed and control inputs are chosen accordingly that are considered to steer the robot in a safe direction within the current interval of time.

4) ORCA: ORCA stands for Optimal Reciprocal Collision Avoidance that as the name suggests finds the best or optimal velocity for a swarm of robots placed in a cluttered environment. It avoids obstacles easily and can be scaled to any number of robots in a swarm[1].
The algorithm works by creating certain admissible velocities for each robot called Collision Avoidance velocities which as their name suggests ensures the robot does not collide with other moving robots/obstacles. These velocities are computed for one robot relative to the approaching robot/obstacle. Each robot has its velocity space that tells it which velocities to take for a time t such that it(and the approaching robot) stay collision free for that period.
It can avoid both static and dynamic obstacles. It is perfect for decentralized swarms as the robots do not need to communicate with each other and thus inter-communication issues and the associated complexities are avoided. It provides a sufficient condition for multiple robots to avoid collisions among one another, and thus can guarantee collision-free navigation for many robots in a cluttered workspace. It achieves convincing motions that are smooth and collision-free. It takes care of the common problem among other obstacle avoidance algorithms called asteroid avoidance which deals with avoiding obstacles that have comparable or greater velocities than the robot.

III. ASSUMPTIONS

- It is assumed that all robots communicate with the server and not with each other, therefore each robot can be assumed to act independently.
- We assume that the robots move in plane R^2 .
- All the robots are circular in shape.
- The Robots are holonomic, i.e., they can move in any direction.
- Preferable velocity for all the robots is

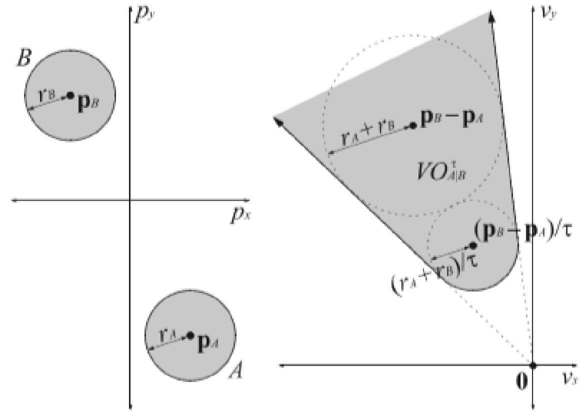
$$V_{robot}^{pref} = V_{robot}^{max}$$

.

- Sensing and Perception is one of the most important part of ORCA, as robots observe the position and velocity of other robots. Hence the sensor data is assumed to be robust and noiseless.
- Robots are able to accurately observe each others current position and velocity (which are external parameters) done through sensing and perception.

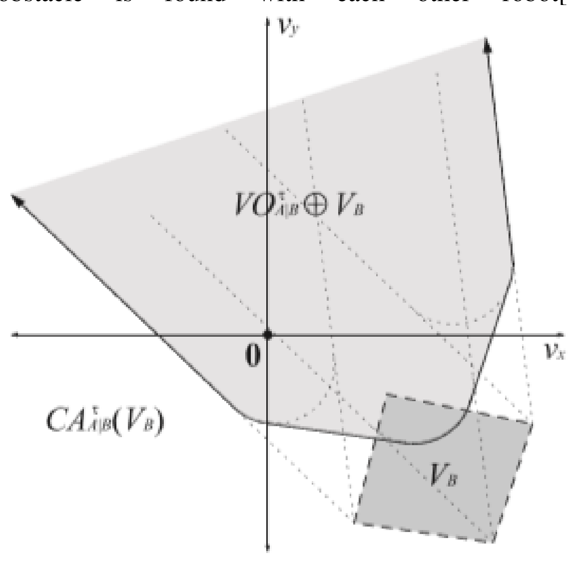
IV. METHOD

- In order to show decentralized/decoupled swarm behaviour, we will take 4-6 robots working independently towards their goal. Decentralized swarm means that each robot though working in collaboration, works independently towards its goal. There is no chain of command, robot receives no information about the position and velocity of the other robot, and decides its position and velocity independently unlike centralized swarms.
- All robots are put on a collision course by giving them a colliding path generated through A*.
- Determine and plot the velocity obstacle for each robot with respect to the other robot in the velocity space.

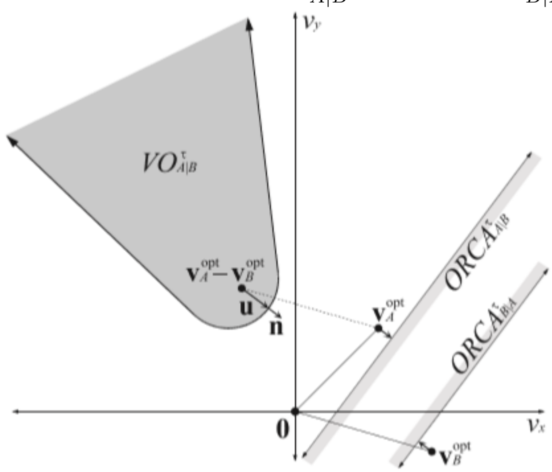


(a) (b)

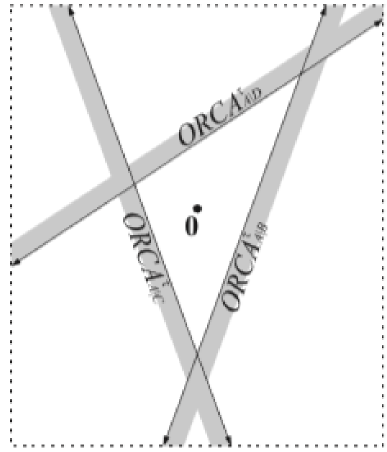
- In the above figures[1] (a) is the configuration of two robots A and B. (b) is the The velocity obstacle $VO^{\tau}_{A|B}$ (gray) which can geometrically interpreted as a truncated cone with its apex at the origin (in velocity space) and its legs tangent to the disc of radius $r_A + r_B$ centered at $p_B - p_A$. The amount of truncation depends on the value of τ , the cone is truncated by an arc of a disc of radius $(r_A + r_B)/\tau$ centered at $(p_B - p_A)/\tau$[1].
- This velocity obstacle $VO^{\tau}_{A|B}$ is the velocity obstacle of A induced by B. Similarly there will be a velocity obstacle $VO^{\tau}_{B|A}$ of B induced by A. The velocity obstacles $VO^{\tau}_{A|B}$ and $VO^{\tau}_{B|A}$ will be symmetric about origin[1].
- Then the Minkowski sum of each robot velocity obstacle is found with each other robot[1].



- The above figure is the Minkowski sum (light gray) of $VO^{\tau}_{A|B}$ and V_B (selected robot B velocity). The set of collision-avoiding $CA^{\tau}_{A|B}(V_B)$ is the complement of the Minkowski sum[1].
- if $v_A - v_B \in VO^{\tau}_{A|B}$, or equivalently if $v_B - v_A \in VO^{\tau}_{B|A}$, then A and B will collide at some moment before time τ if they continue moving at their current velocity. Conversely, if$v_A - v_B \notin VO^{\tau}_{A|B}$, robot A and B are guaranteed to be collision-free for at least τ time[1].
- Hence $CA^{\tau}_{A|B}(V_B)$ and $CB^{\tau}_{B|A}(V_A)$ are the collision avoided velocity of A induced by B and B induced by A respectively.
- Hence the pair V_A and V_B of velocities for A and B are called reciprocally collision avoiding if $V_A \subseteq CA^{\tau}_{A|B}(V_B)$ and $V_B \subseteq CB^{\tau}_{B|A}(V_A)$. If $V_A = CA^{\tau}_{A|B}(V_B)$ and $V_B = CB^{\tau}_{B|A}(V_A)$, we call V_A and V_B reciprocally maximal[1].
- There are infinitely many pairs of sets V_A and V_B that obey these requirements, but among those we select the pair that maximizes the amount of permitted velocities close to optimization velocities v_A^{opt} for A and v_B^{opt} for B. We denote these velocities sets as $ORCA^{\tau}_{A|B}$ for A and $ORCA^{\tau}_{B|A}$ for B[1].
- Derive and plot the half planes equation for optimum velocities such as $ORCA^{\tau}_{A|B}$ and $ORCA^{\tau}_{B|A}$.



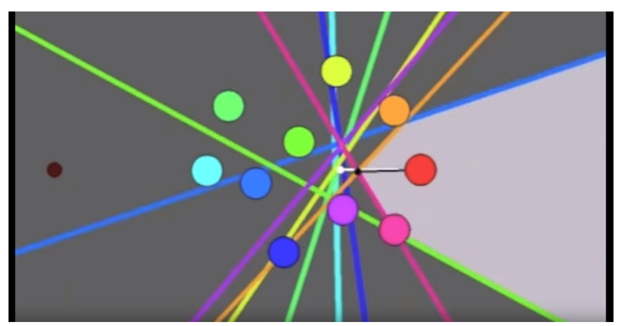
- The above figure[1] shows the half plane plots for $ORCA^{\tau}_{A|B}$ for A and $ORCA^{\tau}_{B|A}$ for B.
- All the corresponding half-plane equations are solved through linear programming.

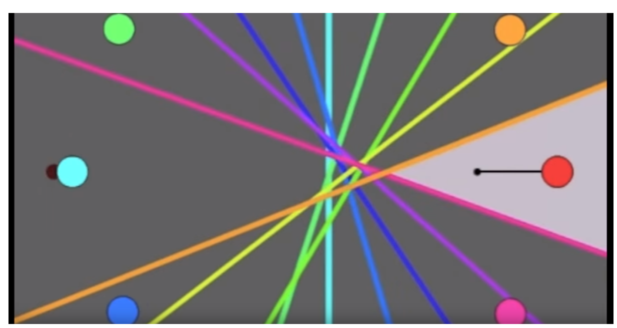


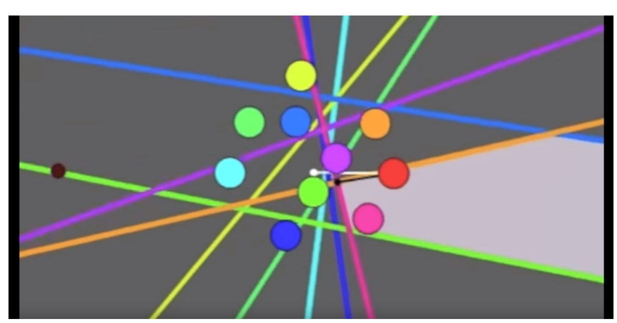
- The above steps are repeated until the robots reach each of their goal destination.

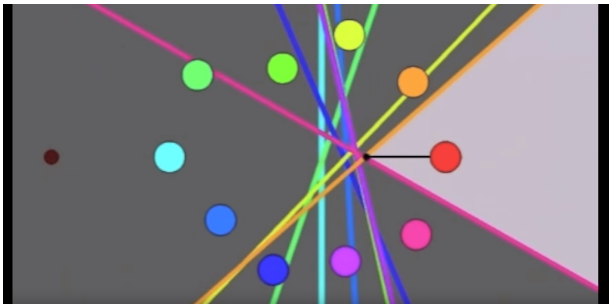
V. EXPECTED RESULT

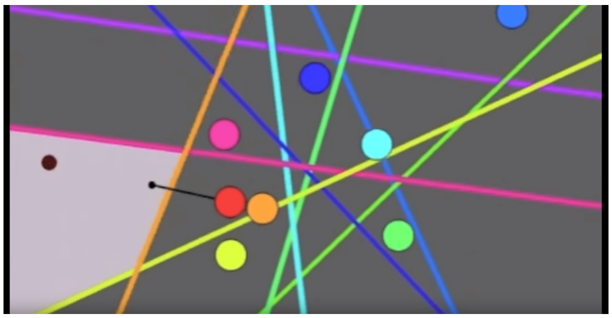
- In the following images it is shown that the robots gradually proceed towards the centre and without colliding with each other find alternative path and revert back in the same direction thus avoiding the collision.
The following figures are referred[11]



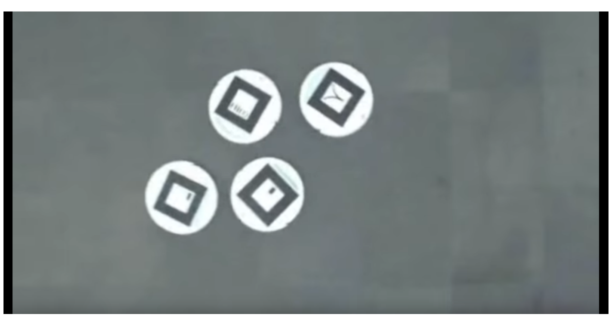


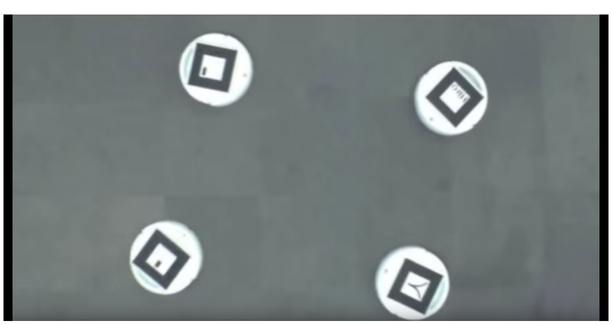






The following images show how it is gone look when the ORCA algorithm is deployed with real swarm robots- 4 in numbers





VI. RESULT AND CONCLUSION

To summarize, we will be implementing a local planning algorithm called ORCA and in order to perform obstacle avoidance in a swarm of robots (3-4) that would be released in an unexplored region. The proposed work can be used for solving real-world problems such as search and rescue, extraterrestrial exploration missions etc. We hope to learn and implement the fundamental aspects of motion planning with this project, and if possible improve upon the existing literature. As discussed above, we will try to implement the same thing but the way of showing the output might differ i.e., we might end up showing the plots of the path taken by each robot in order to avoid collision with each other rather than the real simulation using bots and 3D-obstacle space in ROS or V-rep.Our prime aim will be to generate a list of all the possible collision free velocities for all the robots i.e., 4-6 whereas finding the optimal collision free velocities list for the robots will be in the future work prospect.

Presently, our implementation uses simple robot shapes like circle or discs, the algorithm could be extended to include various types of robot shapes/sizes. In future we could scale this approach to account for robots kinematics and dynamics. Maybe combine path planning in both velocity space and Cartesian space. The current implementation only accounts for 2-D environments. In future this approach could be extended to 3-D.

REFERENCES

[1] Jur van den Berg, Stephen J. Guy, Ming Lin, and Dinesh Manocha Cdric Pradalier, Roland Siegwart, and Gerhard Hirzinger (eds.), Robotics Research: The 14th International Symposium ISRR, Springer Tracts in Advanced Robotics, vol. 70, Springer-Verlag, May 2011, pp. 3-19.

[2] Jamie Snape, Jur van den Berg, Stephen J. Guy, and Dinesh Manocha Proceedings of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), October 2010, pp. 4584-4589.

[3] M. Betke, R. Rivest and M. Singh. Piecemeal learning of an unknown environment. Proc. 5th Conference on Computational Learning Theory, pages 277286, 1993.

[4] S. Albers and M. Henzinger. Exploring unknown environments. Proc. 29th Annual ACM Symposium on Theory of Computing, pages 416425, 1997.

[5] Borenstein, J., Koren, Y.: The vector field histogram - fast obstacle avoidance for mobile robots. IEEE Trans. Robot. Autom. 7(3), 278288 (1991).

[6] Ulrich, I., Borenstein, J.: VFH+: reliable obstacle avoidance for fast mobile robots. In: Proceedings of IEEE International Conference on Robotics and Automation, vol. 2, pp. 15721577, May 1998.

[7] D. Wilkie, J. van den Berg, D. Manocha. Generalized velocity obstacles. IEEE RSJ Int. Conf. Intell. Robot. Syst., 2009.

[8] Fulgenzi, Chiara Spalanzani, Anne Laugier, Christian. (2007). Combining Probabilistic Velocity Obstacles and Occcupancy Grid for safe Navigation in dynamic environments.

[9] R. Simmons. The curvature-velocity method for local obstacle avoidance. IEEE Int. Conf. on Robotics and Automation, pp. 33753382, 1996.

[10] C. Reynolds. Flocks, herds and schools: A distributed behavioral model. Int. Conf. on Computer Graphics and Interactive Techniques, pp. 2534, 1987.

[11] https://youtu.be/zmTgj4SKftc