Dynamic Obstacle Avoidance for assistive autonomy using ORCA

Nagarjun Vinukonda
nvinukonda@wpi.edu

Kavit Nilesh Shah
kshah@wpi.edu

***Abstract*—Mobility in dynamic and crowded environment is essential and challenging task for robot safe navigation. In this paper, we develop obstacle avoidance system which could be used for autonomous as well as semi-autonomous robot navigation in static and dynamic indoor environments. For this purpose, the focus of the project is on use of a state-of-the-art velocity-based planning method called ORCA or Optimal Reciprocal Collision Avoidance, which not only is efficient, but guarantees collision avoidance in static as well as dynamic environments with high degree of scalability.**

***Index Terms*—ORCA, Human-Robot Interaction, Obstacle avoidance, shared control, assistive autonomy**

I. Introduction

Nowadays, service robots are used in indoor environment navigation in offices, schools, shopping malls, etc. For the safe navigation its requires to avoid dynamic obstacles and be aware of human environment. This demands for better perception and navigation modules. But, sometimes there might be problems in perception modules like unable to detect occluded objects and navigation stack unable to move in complex routes, for which shared control will provide a better solution [**9**]. With the human intelligence and robot decision, shared control is expected to provide better navigation in cluttered environment like hospitals, which is the gist our problem statement.

Shared control has been a prominent area of research which intersects with many different domains one of which is robotics. Haptic shared control is a specialized branch which deals with haptic feedback to the human engaging in shared control. The importance of haptic shared control in robotics is very well highlighted in [**2**] which explains enhanced performance with reduced control activity of robot with shared control over fully-automation. The paper laconically explains the term of Level of Haptic Authority (LoHA) as one of the defining parameter of a Human-Robot Interaction system, where higher autonomy have higher LoHA. The assistive-autonomy system which we propose to design employs an intermediate LoHA with intermediate shared control where the human provides velocity commands while the robot moves along the approximate direction & scale of velocity, but generates obstacle avoidance maneuvers on its own whenever necessary.

In shared control it is expected that navigation stack will provide dynamic obstacle avoidance to ease the shared autonomy controlled by human. For which many approach's have been proposed Artificial Potential Fields algorithms, Sampling based algorithms, Probabilistic algorithms, Dynamic Window Approach, Elastic band method, Genetic Algorithms, etc [**3**]. Among them none of the algorithms dive into velocity based approach. The velocity based methods inherently plan their motion in velocity space rather than the conventional position space. This approach provides better ease of implementation over other methods and it is the state of the art method being used in dynamic obstacle avoidance.

Among the Velocity based obstacle methods, ORCA has shown its prominence both in dynamic obstacle avoidance and social aware navigation[**4**] over predicting human behaviour with human interaction and providing high success over collision avoidance.

In this paper we provide significance of ORCA over other VO methods, framework of integrating perception modules with ORCA working over static and dynamic obstacles, robust obstacle detection and its tuning parameters, experimental results of ORCA on detected obstacles highlighting its performance with tuning the hyper-parameters, discussions & conclusions.

II. Related Work

Motion Planning for robots has been one of the popular research domains. It is an integral component of any autonomous & semi-autonomous robot navigation and finds its use in self-driving cars, agricultural robots, tele-operated robots, etc. Motion planning in itself is a wide branch in robotics and many-a-times it is linked to Human-Robot Interaction (HRI). One could consider as a example of a tele-operated nursing robot controlled by human which is catering needs for quarantined patients, as an example of HRI with shared control.

Shared control of teloperted robots requires obstacle avoidance when human is unable to navigate safely. An article [**15**] showcases real-time obstacle avoidance of wheel chair using artificial potential fields in shared control, but their autonomy is more sensor depended and doesn't overcome by itself the intrinsic sensor shortcomings which leads for robot being stuck at few blind spots. On the other hand, Velocity based approach doesn't get stuck rather it moves in certain preferred velocity.

The paper [**5**] coins the term Velocity Obstacles and velocity cones, where in robot is in collision if its velocity lies inside the velocity obstacles generated by the obstacles in the environment. To avoid collision, it tries to compute velocities lying outside the VO. However, this method leads

to oscillatory motion when two agents approach each other. This is solved by using the Reciprocal Velocity Obstacles [**15**] which smartly shifts the robot velocities in the velocity-space to avoid oscillations or reciprocity dance. This works well with two agents, but fails when dealing with 3 agents colliding with each other, as it again endures in oscillations. Similarly, there are other VO based methods which are recently introduced. The paper [**16**] this uses a method called GRCO(Generalized reciprocal collision avoidance) which is mixture of VO [**5**], AVO[**19**], CCO, LQR-O[**21**] and Reciprocity[**1**]. It works for non-homogeneous robot systems with non-linear equations of motion and generates smooth collision-free motion for all robots. But, there are certain limitations where it does not consider uncertainty and its unclear whether the algorithm works in narrow passages.

Another paper [**17**] proposed BRVO. It is a novel approach used for predicting obstacle trajectories using Bayesian statistics. The trajectories generated by this method are more accurate then the trajectories generated by methods with prior calculated each-agent parameters. Although it has many advantages, it is learning based method which has high computational complexity and it is also not easier to implement. Comparatively, ORCA is easier in implementation and provides robust planning for obstacle avoidance.

The paper [**6**] written by the same author as RVO, introduces the concept of ORCA or Optimal Reciprocal Collision Avoidance which addresses and solves all the previous problems and guarantees collision free robot motion. Also it is highly scalable and efficient thereby making it the right choice for the project.

The Section III discusses about the framework and working principle of ORCA & the obstacle tracking approach. The subsequent section (4) talks about the simulation environment used for the project and various experiments conducted upon it. Section V provides the simulation results and exhaustive analysis of the results. We end by talking about future scope and conclusion in section VI. The tabular results are at the end of the report in the Appendix section VIII

III. Methodology

The entire project has been accomplished in Robot Operating System (ROS) with C++ as a programming language. This section explains about detailed implementation of the obstacle avoidance system which comprises of ORCA and Object tracking as the two major components.

A. Framework

The obstacle avoidance system has the following sub-modules and sub-systems :

1) Sensor (LiDAR) : The system incorporates the use of a 2D LiDAR to sense the environment because of its low computation costs & rich 360information.
2) Localization and Mapping : This sub-system uses the LiDAR reading to initially develop environment map using SLAM g-mapping and later uses the LiDAR scans to localize itself into the environment. As a result of localization, the robot possesses its real-time transformations with respect to the world frame.
3) Pedestrian Detection & Tracking : This sub-module detects and tracks moving obstacles and humans in the environment around the robot using the laser scans and Kalman filters. It sends the tracked agent's data to ORCA via ros-msg with message field information shown in fig 1.
4) ORCA : This is the core of the obstacle avoidance system. It takes in the static & dynamic environment information from surroundings, robot's position and velocity & computes the change in velocity if necessary to avoid collision with any &/or every obstacle in the surroundings.
5) Robot Controller : Finally the computed velocities are passed to the low-level robot controller from the ROS-Navigation Stack which simulates the robot motion in GAZEBO

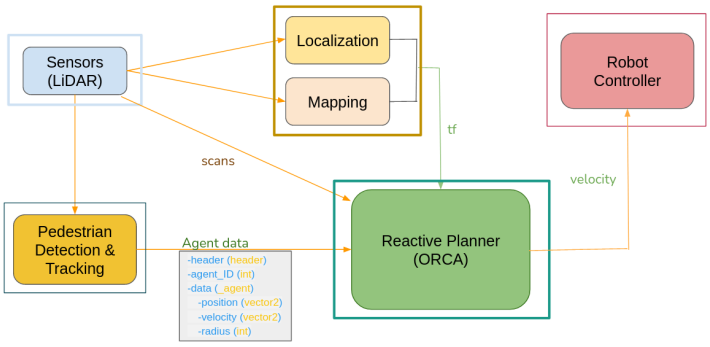


Fig. 1: Framework of dynamic obstacle avoidance system

B. ORCA

The ORCA is by far one of the most highly valued algorithm for robot navigation & also finds its use in game development and actor motion simulation especially in crowded human simulations. This could be accumulated to the fact that, it addresses issues of a many of its predecessor algorithms while it only has a few setbacks, thus overall providing higher number of advantages over disadvantages, thereby making it an appreciable choice of algorithm for simulation.

The ORCA algorithm is build upon the concept of velocity obstacles or VO [**5**]. It was an intuitive but a novel approach to collision detection as seen in Fig.(2). If A is the robot & B represents the obstacle, then VO shrinks robot A to point robot & it inflates the nearby surrounding obstacles by the radius of A. It then computes a cone from point A to the circle B & checks if the relative velocity lies inside of the velocity cone, in case of which it would be a collision, and otherwise not.

We are using the RVO2 C++ library developed by the authors of ORCA [**14**]. The library is not defined for ROS. Making necessary tweaks and changes, we made it compatible to operate with ROS.

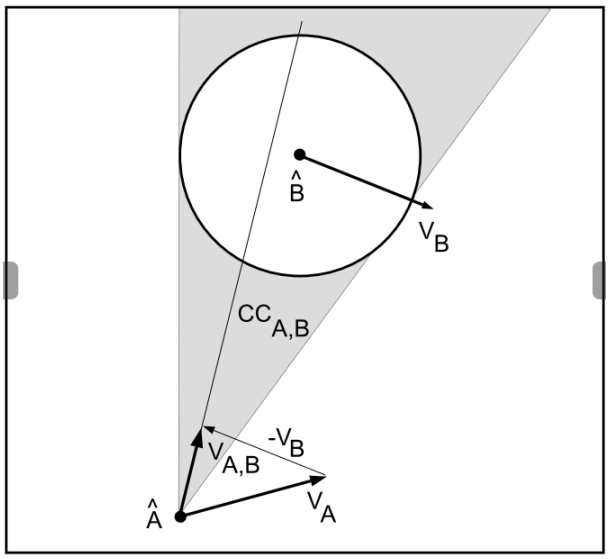


Fig. 2: Velocity Obstacles [**5**]

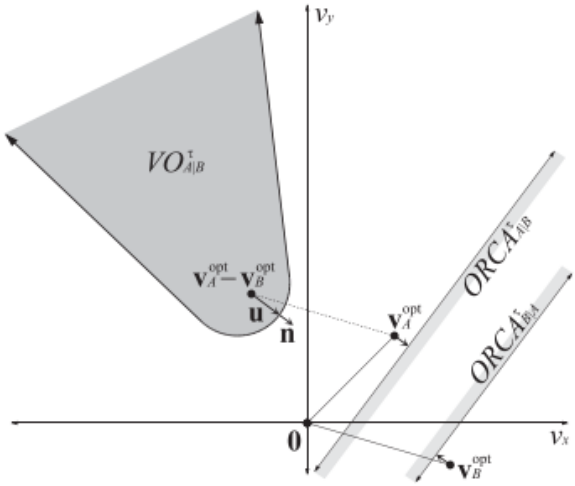


Fig. 3: Velocity Cone and ORCA Lines [**6**]

ORCA though builds upon the concept, is not intuitive to understand, but could be summarized the the following 4 steps:

1) Construct velocity cones for all obstacles around the robot & check if the relative velocity lies inside the velocity obstacle.
2) If so, using optimization, find the smallest possible vector which on adding it to the relative velocity, brings the relative velocity out of the collision cone as seen by the vector u in fig 3. This is the change in velocity which is to be shared by both the robot & obstacle using a distribution factor.
3) On obtaining the minimum collision avoidance velocity vector u for each collision, convert it into an ORCA line as shown in 3. ORCA line is an infinitely long line perpendicular to the change in velocity vector which is added to the current velocity vector of A. The ORCA line divides the region into two half-planes. If robot chooses its new velocity on one-side or half-plane it is guaranteed to avoid collision from the given obstacle, however, on choosing velocities on other side, it may or may not avoid collision.
4) The last step is the consider the ORCA lines for all agents around the robot & using LPP trying to solve for minimum vector which gets the current robot velocity inside of the inclusive zone provided by LPP as shown in fig 4. This final velocity would be the necessary obstacle avoidance velocity for the robot within a given time-horizon (t_h).

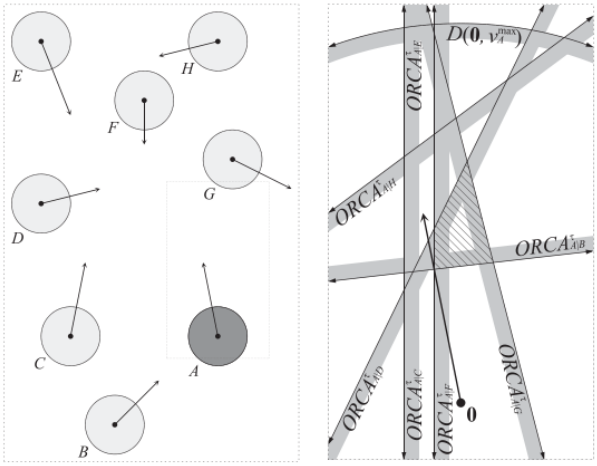


Fig. 4: (a). The robot trying to navigate itself in an agent cluttered environment (b). Robot trying to compute new velocity using LPP on ORCA half-planes for each obstacles [**6**]

C. Obstacle Detection & Tracking

The purpose of obstacle tracking is to determine radius, position and velocity of obstacles in order for ORCA to accurately determine preferred velocity to navigate around moving obstacles. Among different packages we have found, obstacle detector package[**13**] is found to be providing necessary requirements. But, we are require to modify it further to provide accurate results which we will be discussing further.

The obstacle detector has three nodes: scan merger, obstacle extractor, obstacle tracker nodes. The following figure Fig.(5 illustrates framework of the package.

1) scan merger : It subscribes & merge laser scan data and converts them into point cloud.
2) obstacle extractor : It subscribes scan merger data and creates circular obstacles and line segments using split and merge algorithm.
3) obstacle tracker: It subscribes obstacle extractor data and supplements with velocity of obstacles using Kalman filters.

IV. Experiments

A. Simulation Environment

The aim of the project to navigate the mobile robot in hospital environment. Hence, we have chosen our simulation

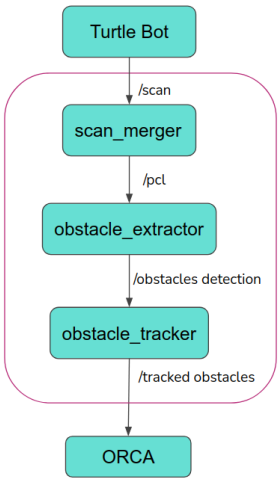


Fig. 5: Obstacle Detector frame work

environment as hospital aisle & rooms and hospital lobby as shown in Fig.(6).

(a) Hospital Aisle & Rooms **(b)** Hospital Lobby

Fig. 6: Simulation Environment

B. ORCA parameter tuning

Running the robot simulation multiple times for different start and goal positions in Hospital Lobby environment brought out some interesting observations. While the robot performed very well for generalized cases of start & goal locations, there exists a certain variety of approach which we refer to as head-on where the robot's goal is at diametrically opposite of the start point. In this case, the robot suffered heavily because it couldn't compute new paths instantly. This warranted a need to look-into the ORCA hyper-parameters and tune them. To maintain uniformity, we set the start point of robot to (-5m, 0m) & goal point to (6m, 0m) which a large cylinder in the path.

1) Robot laser scan range : the range of laser scans
2) Time horizon to obstacles : time into the future from the present time for which the ORCA should check for collisions
3) Preferred velocity : robot's current velocity (for autonomous navigation) or tele-operator specified velocity (for semi-autonomous navigation) which the ORCA assumes the robot is currently moving with to check for collisions.

In order to evaluate the performance after tuning the hyper-parameters, we used the following performance evaluation metrics:

- Success rate (%) : In trials of 3 for each set of the 3 above-mentioned hyper-parameters, the % of trials in which the robot didn't collide.
- Avg. min. clearance (m) : Average of minimum clearance from robot to the near-by surrounding obstacles in each set of 3 trials
- Avg. completion time (sec) : It is the average of total time taken by the robot to reach from start point to the goal point in each set of 3 trials.

4 different experiments have been conducted to study the performance of range of laser scan with variation of th_{obs}.

1) Experiment 1 : In this experiment, robot laser scan range is set to 3.5 m & v_{pref} scaling factor is set to 1. The variation of th_{obs} from 100 units to 1200 units have been observed and recorded in Table I.
2) Experiment 2 : In this experiment, robot laser scan range is set to 4 m & v_{pref} scaling factor is set to 1. The variation of th_{obs} from 100 units to 1200 units have been observed and recorded in Table II.
3) Experiment 3 : In this experiment, robot laser scan range is set to 5 m & v_{pref} scaling factor is set to 1. The variation of th_{obs} from 100 units to 1200 units have been observed and recorded in Table III.
4) Experiment 4:
   - 4a : In this experiment, robot laser scan range is set to 10 m & v_{pref} scaling factor is set to 1. The variation of th_{obs} from 100 units to 1200 units have been observed and recorded in Table IV.
   - 4b : In this experiment, robot laser scan range is set to 10 m & v_{pref} scaling factor is set to 2. The variation of th_{obs} from 100 units to 1200 units have been observed and recorded in Table V.

C. Obstacle detector parameters tuning

The obstacle detector tuning depends on certain hyper parameters for which we require to understand the obstacle extraction. The obstacle extractor framework as follows Fig.7.

1) Obstacle extraction: The concept of obstacle extraction is explained well in the paper[**7**]. There are some set of hyper parameters which play major role in detecting obstacles as shown in Fig.(8). The working of the obstacle extraction node is as follows:

1) Grouping: From the /pcl acquired, we consider N_min points to segregate as groups/clusters when the grouping criteria is not satisfied. d_i is distance between points.

$$d_i - 1 < d_group + R_i * d_p \quad (1)$$

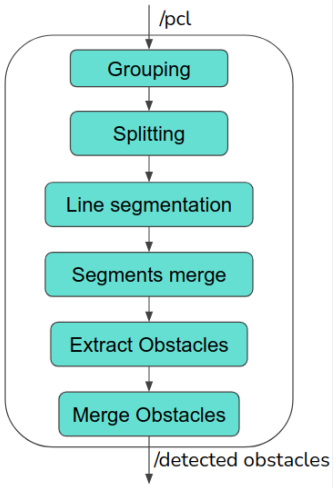


Fig. 7: Obstacle extractor frame work

Parameters	Criteria
Min_group_points (N_min)	Minimum no.of points to create a group
Distance_proportion (d_p)	Allowable distance between points
Max_group_distance (d_group)	Create group points d > d_group
Max_split_distance (d_split)	split a group point d > d_split
Max_merge_spread (d_spread)	Merge line segments, if extreme points d < d_spread
Radius_enlargement (r_merge)	Enlarge radius
Max_merge_separation (d_sep)	Merge obstacles d < d_sep
Max_circle_radius (r_max)	Discard obstacle radius > r_max

Fig. 8: Obstacle extractor parameters

2) Splitting: Further large groups are divided into smaller groups with the following criteria. d_j is distance between the leading line and the farthest point in group.

$$d_j > d_split + R_j * dp \tag{2}$$

3) Line segmentation: After splitting smaller groups the package creates line segments for these groups using ICP or split and merge algorithm.
4) Segments Merge: Later line segments which are close by are merged to form a common line segment.
5) Extract circles: when a group of line segments form a triangle it is inscribed in a circle also known as obstacle. Then the radius of circles are enlarged.
6) Merge obstacles: The obstacles are merged if distance the distance between them is less than d_sep. Followed by Huge circles are discarded if new circle radius is greater than r_max. The reason for r_max is to exterminate unnecessary obstacles formed on corridor walls which are very long when extracted.

2) Parameter tuning: During experiment we found few explicit parameters that effects the detection:

1) Velocity of obstacle: The maximum velocity for better tracking and state estimation of obstacle is found to be $9.5m/s$. Over this velocity the tracking sometimes is not a accurate. It can be enhanced further through increasing process co variance but this will also lead to increase in noise.
2) Rotation of robot: When the robot is rotated more than 1.5 m/s, there is sometimes formation of temporary false obstacles due to kalman filter tracking duration is $1sec$.
3) Distance from obstacle: The laser range we are using in the experiment is $15m$. But, the pcl generated for smaller obstacles like human legs are small from larger distances and pcl increases when we come nearby. After fine tuning the maximum distance for robust detection of Humans is $6m$.

The following are set of parameter used for robust human and obstacle detection as shown in Table.(VI) in Appendix. The difference in tuning parameters is due the less no. of point clouds generated by human legs.

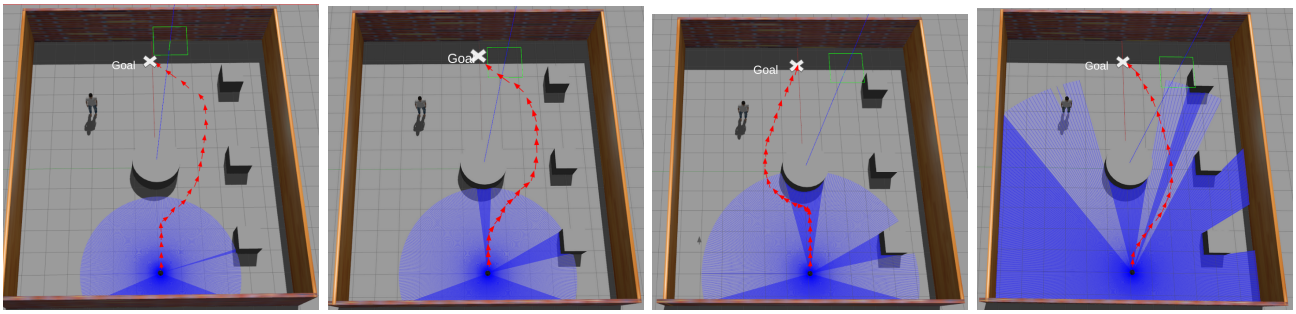
V. Results and Analysis

This section talks about the results of obtained from the aforementioned experiments. The first analysis is of the ORCA experiments.

A. ORCA simulation results

We tried to experiment with the different ORCA hyper-parameters such robot scan range, time-horizon to obstacles & preferred robot velocity

1) Analysis 1 : The results of experiment 1 are presented in Table I. It can be seen that the robot with scan range of 3.5m gives best computation time at $th_{obs} = 400$ units. Being highly narrow-sighted (with smaller th_{obs}) & observing highly local environment causes the robot to perform avoidance maneuvers swiftly as it has very low data to process. It perceives threat from obstacles when it reaches very close to them & only after it avoids them, as seen in fig 9a. As the th_{obs} is increased, the robot's input data to be processed increases as it would now consider more obstacles within the 3.5m range and this slows down the algorithm. Thus, the use of range 3.5m is not recommended for practical purpose.
2) Analysis 2 : The results of experiment 2 as seen in Table II suggest that with increase in th_{obs}, the computation time initially decreases, reaches a minimum at th_{obs} = 600 & then further increases with increase in th_{obs}. The robot starts planning more carefully which increases its computation time. But still this scan range is not capable of seeing larger distance. Thus use of range = 4m is also not application because of narrow-sightedness as well as



(a) Laser scan range = 3.5m **(b)** Laser scan range = 4m **(c)** Laser scan range = 5m **(d)** Laser scan range =10m

Fig. 9: Trajectories followed by the robot while avoiding obstacles with v_{pref} factor of 1 with th_{obs} giving lowest computation time (Table I-V)

high variance in the minimum clearance between robot & obstacles, thus not reliable.

3) Analysis 3 : The results of experiment 3 as seen in Table III show that the robot, for th_{obs} between 400 to 600 has similar performance. Since the range is quite large, robot has already perceived a good extent of nearby environment and could be considered as medium-sighted. However due to this distance information, the computation of best velocity becomes difficult for the robot and it cannot *decide* the best avoidance maneuver till it reaches too close to the obstacle. This "confused" behavior is a good indication that the robot has started to see more distant objects, but still needs more tuning. This motivates us to the last stage of experiments

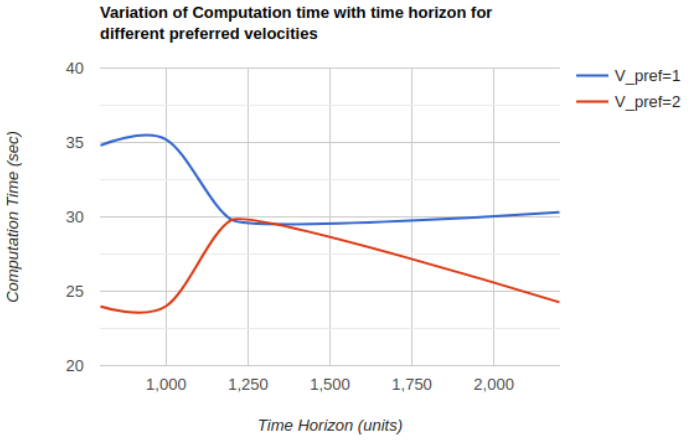


Fig. 10: Variation of th_{obs} with computation time for different values of v_{pref}

4) Analysis 4 :
   - Table IV gives great insights about the performance of ORCA with laser range of 10m. The robot is not longer narrow sighted & is capable of seeing far and wide into the environment. The robot for each of the th_{obs}, not only has a consistent computation time, but also consistent and adequate minimum clearance to the obstacles. Also as seen from the fig 9d, the robot's trajectory is smoother & is takes off to avoid head-on collision quite early in time, indicating robustness of the algorithm & best set of parameters.
   - Another comparison between Tables IV and V show that increasing the preferred velocity from 1 to 2 causes the robot to move more smoothly & with lesser computation time as evident from the fig 10

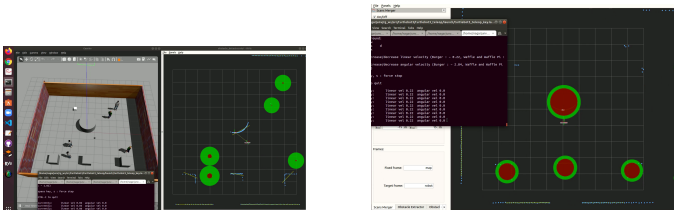
To summarize the discussion of results, the robot for best performance, should be operated at laser scan range of 10m, with preferred velocity scaling factor of 2 and th_{obs} set to 1000 units.

Since the system offers collision avoidance for both tele-operator commands as well as point-to-point motion, we have added a new features the package.

- Autonomous Navigation mode : In this mode the robot works in conjunction with local and global planners, thereby taking fine-tuned local goals & executing the goals using ORCA as active dynamic obstacle avoidance. This makes our system suitable for 'complete autonomous indoor robot navigation' capable to work with both static & dynamic environments
- Semi-autonomous mode : In this mode, the robot takes high-level velocity inputs from the human tele-operator, and keeps travelling with it until it encounters a possible collision with a time-horizon. Then it, using ORCA, performs local collision avoidance thereby being capable for semi-autonomous shared control as well.

B. Obstacle detection results

Through fine tuning the params we are able to detect humans and obstacles robustly as shown in Fig.(11). The obstacle detector params can also be used in humans detection but, it requires the robot to be in vicinity of 2-4m. There are certain limitations for this package: dependency of tuning based on size of obstacles in environment(Large vs small), mixed pixel effect causing package unable to track smoothly sometimes. Overall, for our project which requires human and obstacles detection in hospital environment the package works robustly.



(a) Robust human detection **(b)** Robust Obstacle detection

Fig. 11: Fined tuned package results

VI. Future Scope and Conclusion

During our project we have faced many challenges like deciphering the RVO2 (ORCA) the obstacle tracking libraries, integrating the RVO2 with ROS, fine tuning ORCA hyper parameters, robust detection of humans and able to solve hese problems. Overall, we conclude that ORCA works well for Static Scenarios (as tested) though it has some limitations. The major limitations include workable range of sensor readings, accurate positioning of map odom frames in ROS i.e. tf frames distorts resulting ORCA to provide bad results if odom and map are not positioned properly. Similarly obstacle detection requires proper tuning based on environment we are using. For our project we are able to tune robustly.

In future, continuing this project one can make the laser scan as a cycloidal shape to emphasize more on obstacles right in front of it, modify the hyper-parameters to generalize in all scenarios, add machine learning module to track and detect humans /or objects, classify obstacles into stationary, movable or mobile when detected and provide best fine tuned KF parameters. Finally, we would like to integrate ORCA and obstacle detector packages to navigate safely in hospital environment with extension of project for two weeks.

VII. Acknowledgement

We would like to thank Dr. Zhi Li for the guidance and Kenechukwu Mbanisi for mentoring us throughout the project

References

[1] Van den Berg, Jur, Ming Lin, and Dinesh Manocha. "Reciprocal velocity obstacles for real-time multi-agent navigation." 2008 IEEE International Conference on Robotics and Automation. IEEE, 2008.

[2] Mulder, Mark, David A. Abbink, and Erwin R. Boer. "Sharing control with haptics: Seamless driver support from manual to automatic control." Human factors 54.5 (2012): 786-798.

[3] Mohanan, M. G., and Ambuja Salgoankar. "A survey of robotic motion planning in dynamic environments." Robotics and Autonomous Systems 100 (2018): 171-185.

[4] Chen, Changan, et al. "Crowd-robot interaction: Crowd-aware robot navigation with attention-based deep reinforcement learning." 2019 International Conference on Robotics and Automation (ICRA). IEEE, 2019.

[5] Fiorini, Paolo, and Zvi Shiller. "Motion planning in dynamic environments using velocity obstacles." The International Journal of Robotics Research 17.7 (1998): 760-772.

[6] Van Den Berg, Jur, et al. "Reciprocal n-body collision avoidance." Robotics research. Springer, Berlin, Heidelberg, 2011. 3-19.

[7] Przybyła, Mateusz. "Detection and tracking of 2d geometric obstacles from lrf data." 2017 11th International Workshop on Robot Motion and Control (RoMoCo). IEEE, 2017.

[8] Randhavane, Tanmay, et al. "Pedestrian dominance modeling for socially-aware robot navigation." 2019 International Conference on Robotics and Automation (ICRA). IEEE, 2019.

[9] Hassan, Shabih Ul, et al. "Modified SOM based intelligent semi-autonomous navigation system." 11th Symposium on Neural Network Applications in Electrical Engineering. IEEE, 2012.

[10] Itoh, Makoto, Toshiyuki Inagaki, and Hiroto Tanaka. "Haptic steering direction guidance for pedestrian-vehicle collision avoidance." 2012 IEEE International Conference on Systems, Man, and Cybernetics (SMC). IEEE, 2012.

[11] Hoy, M., Matveev, A. S., Savkin, A. V. (2015). Algorithms for collision-free navigation of mobile robots in complex cluttered environments: A survey. Robotica, 33(3), 463–497. https://doi.org/10.1017/S0263574714000289

[12] Leigh, A., Pineau, J., Olmedo, N., Zhang, H. (2015). Person tracking and following with 2D laser scanners. Proceedings - IEEE International Conference on Robotics and Automation, 2015-June(June), 726–733. https://doi.org/10.1109/ICRA.2015.7139259

[13] https://github.com/tysik/obstacle_detectors

[14] http://gamma.cs.unc.edu/RVO2/

[15] Petry, Marcelo R., et al. "Shared control for obstacle avoidance in intelligent wheelchairs." 2010 IEEE Conference on Robotics, Automation and Mechatronics. IEEE, 2010.

[16] Bareiss, Daman, and Jur van den Berg. "Generalized reciprocal collision avoidance." The International Journal of Robotics Research 34.12 (2015): 1501-1514.

[17] Kim, Sujeong, et al. "Brvo: Predicting pedestrian trajectories using velocity-space reasoning." The International Journal of Robotics Research 34.2 (2015): 201-217.

[18] Wilkie, David, Jur Van Den Berg, and Dinesh Manocha. "Generalized velocity obstacles." 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, 2009.

[19] PVan Den Berg, Jur, et al. "Reciprocal collision avoidance with acceleration-velocity obstacles." 2011 IEEE International Conference on Robotics and Automation. IEEE, 2011.

[20] Snape, Jamie, et al. "The hybrid reciprocal velocity obstacle." IEEE Transactions on Robotics 27.4 (2011): 696-706.

[21] Bareiss, Daman, and Jur Van den Berg. "Reciprocal collision avoidance for robots with linear dynamics using LQR-obstacles." 2013 IEEE International Conference on Robotics and Automation. IEEE, 2013.

[22] Balachandran, Avinash, Matthew Brown, Stephen M. Erlien, and J. Christian Gerdes. "Predictive haptic feedback for obstacle avoidance based on model predictive control." IEEE Transactions on Automation Science and Engineering 13, no. 1 (2015): 26-31.

[23] Cooperative human-robot haptic navigation." In 2014 IEEE International Conference on Robotics and Automation (ICRA), pp. 2693-2698. IEEE, 2014.

VIII. APPENDIX

1) The codes of our project are available on Github : https://github.com/KavitShah1998/DynamicObstacleAvoidance

2) The YouTube link of our project presentaion can ne found here https://youtu.be/ZYMectHrw4U

3) Result tables :

Th_obs	Success rate(%)	Min. clearance(m)	Computation time(sec)	Status
100	33	-	-	Collision
200	66	-	-	Collision
400	100	1.04 (low σ)	27.84	Success
600	100	1.03 (low σ)	31.2	Success
800	100	0.91	32.6	Success
1000	100	0.79	33.05	Success
1200	100	0.893(high σ)	35.31	Equidistant

TABLE I: Th_obs metric comparison for Laser Scan 3.5m

Th_obs	Success rate(%)	Min. clearance(m)	Computation time(sec)	Status
100	50	-	-	Collision
200	50	-	-	Collision
400	100	0.92	38.96	Success
600	100	0.433(high σ)	29.31	Success
800	100	0.84	47.02	Success
1000	100	0.85	52.24	Success
1200	100	0.39(high σ)	44.6 (high σ)	Success

TABLE II: Th_obs metric comparison for Laser Scan 4m

Th_obs	Success rate(%)	Min. clearance(m)	Computation time(sec)	Status
100	0	-	-	Deadlock
200	25%	-	-	Deadlock
400	100	0.87	37.84	Success
600	100	0.92	40.84	Success
800	100	0.85	39.46	Success
1000	100	0.84(high σ)	55.87	Success
1200	100	0.88	54.729 (high σ)	Success

TABLE III: Th_obs metric comparison for Laser Scan 5m

Th_obs	Success rate(%)	Min. clearance(m)	Computation time(sec)	Status
100	0	-	-	Deadlock
200	33%	-	-	Deadlock
400	66	0.87	37.84	Stuck, Slow
600	66	0.95	34.84	Stuck, Slow
800	100	0.95	34.846	Success
1000	100	0.97	35.20	Success
1200	100	0.92	29.8	Success
2200	100	0.95	30.32	Success

TABLE IV: Th_obs metric comparison for Laser Scan 10m & V_{pref} scaling factor of 1

Th_obs	Success rate(%)	Min. clearance(m)	Computation time(sec)	Status
800	100	1	23.98	Success
1000	100	0.99	24	Success
1200	100	0.97	29.8	Success
2200	100	0.93	24.26	Success

TABLE V: Th_obs metric comparison for Laser Scan 10m & V_{pref} scaling factor of 2

parameters	Obstacle params	Human params
N_min	3	2
d_p	0.06	0.09
d_group	0.1	0.4
d_split	0.3	0.4
d_spread	0.3	0.6
d_sep	0.3	0.8
r_merge	0.2	0.7
r_max	1.2	1

TABLE VI: Parameters used for robust large obstacle and Human detection