Collision Avoidance with Optimal Path Replanning for Mobile Robots*

Vibhakar Mohta[1,3][0000−0001−5531−9383], Sagar Dimri[2,3][0000−0001−7008−3655], Hariharan R.[2,3][0000−0002−5342−4291], and Sikha Hota[2,3][0000−0001−8924−560X]

[1] Department of Mechanical ez Engineering
[2] Department of Aerospace Engineering
[3] Indian Institute of Technology, Kharagpur

Abstract. This paper generates a collision-free trajectory for wheeled mobile robots in presence of dynamic obstacles. The existing literature solves the collision avoidance problem by changing the velocity vector instantaneously, which is not feasible due to the non-holonomic constraints of robots. So in this work, a smooth change in the velocity vector along with constraints in turn radius has been considered for any required maneuvers. This work also re-plans the path evading re-collision to reach the goal ensuring minimum deviation from the initial path, which was also not addressed in the literature. The low computational requirement of the proposed algorithm allows for online applications on wheeled mobile robots with limited computational resources. The approach is validated through simulations on multiple randomized configurations.

Keywords: Collision Avoidance · Optimal Path · Dynamic Obstacles

1 Introduction

Mobile robots play an integral role in shaping mankind's lifestyle but have many challenges associated to address. Reaching a specified goal while avoiding unwanted obstacles in a cluttered environment is one of the important requirements for automation. Collision-free navigation depends upon the vehicle model, sensor arrangement and optimality. [11] and [19] reviewed different avoidance systems for collision-free navigation.

Motion-planning algorithms like vector-field histograms and the bug algorithm [22] can provide a feasible path to the goal but they are not optimal. Such algorithms use occupancy-grid methods to model the environment. A^* [15], *RRT* [16] and Delaunay triangulation [25] can provide a near-optimal path in finite time. But, these algorithms are fraught with the problem of large computational costs. Geometry-based algorithms, which have significantly lower computational costs, have recently been used for collision avoidance. They are extensively used for optimal path planning in various missions both in two-dimensional (2D) plane ([3] and [14]) as well as in three-dimensional (3D) space ([10] and [9]).

* We would like to thank ECR grant, SERB and GOI for their support.

Finding the shortest path to converge to a circular path [3] and reaching a target via circular boundaries for vehicles with bounded curvature [14] are a few planning algorithms based on Dubins curves [5].

Dynamic-Window Approach (DWA) ([7] and [24]) discusses two-dimensional space-search algorithms for translational and rotational speeds to provide the permissible trajectories for short-intervals of time. Constraints over the velocities are taken into account while creating the dynamic window. Hybrid DWA [20] utilises a 3D search space for collision avoidance. Potential field-based methods [4] and [29] create a field-based upon forces of attraction towards goals and repulsion from obstacles. These methods are mainly used with velocity obstacle methods [27] and can be implemented on manipulators, Autonomous Underwater Vehicles [6] and quadrotors [12]. Potential field methods have a limitation of getting stuck in local minima, but extensions like simulated annealing [30] and modified Artificial Potential Field [23] have the potential overcome them.

A powerful collision cone approach was proposed in [2], which was utilised in reactive-collision avoidance maneuvers [21], velocity obstacle methods [13] and conflict detection and resolution techniques in aircraft [8] and [1]. [28] uses the dynamics of an omnidirectional robot to solve the avoidance problem in a cluttered environment. Analytical solutions for the optimal path can be obtained by integrating vehicle dynamics with obstacle geometry. [17] and [18] proposed a velocity obstacle technique for collision avoidance with spherical and cylindrical safety bubbles for multiple aircraft. But, it requires an instantaneous change in velocity directions, which is not feasible for the dynamical constraint of vehicles.

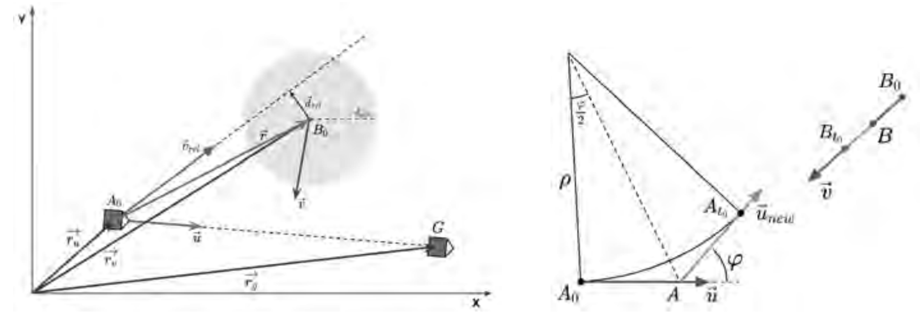
The novelty of this work is to ensure smooth changes of the velocity vectors for all the maneuvers required for collision avoidance and re-planning the path making it implementable in nonholonomic robotic platforms. This work also finds a solution to evade re-collision while re-planning the path to the goal ensuring minimum deviation from its initial trajectory, which was also not explored in the literature. The low computational requirement of the proposed algorithm makes it online implementable on wheeled mobile robots.

2 Problem Formulation

Let us consider a ground robot initially at A_0, with the position vector $\overrightarrow{r_u}$, and the velocity $\overrightarrow{u}$ directed towards its goal (G) $\overrightarrow{r_g}$. The state of the robot is $X : [x, y, \theta]$ and the minimum turn radius is ρ. It detects a dynamic obstacle, B_0 at $\overrightarrow{r_v}$ headed on a collision course towards it with a velocity $\overrightarrow{v}$. The objective is to re-plan the path for the robot such that it avoids the obstacle B_0 with a safety radius $d_{\min}$, and reach the goal, G with state $X_g : [x_g, y_g, \theta_g]$. The kinematic equations of motion are:

$$\dot{x} = |\overrightarrow{u}| \cos\theta \qquad \dot{y} = |\overrightarrow{u}| \sin\theta \qquad \dot{\theta} = \omega \tag{1}$$

where, ω is the angular rate at any given instant. It is assumed that the robot moves with a constant speed $|\overrightarrow{u}|$ and the velocity of the obstacle remains constant. All vectors are measured in an x-y inertial frame as shown in Fig 1a.



(a) Position vectors of the mobile robot A_0, obstacle B_0 and goal G

(b) Turning maneuver of robot in a circular path

Fig. 1: Problem formulation and collision avoidance maneuver

3 Collision Detection and Avoidance

The avoidance maneuver is executed only after collision with the obstacle is predicted. In Fig. 1a, the *minimum separation* in vector between the robot and the obstacle is given by $\overrightarrow{d_{\rm rel}} = (\vec{r}.\hat{v}_{\rm rel})\hat{v}_{\rm rel} - \vec{r}$, where, $\vec{r}$ is the relative position of the obstacle with respect to the robot and $\hat{v}_{\rm rel}$ is the unit vector along $\overrightarrow{v_{\rm rel}}$. Let, d_{min} be the *radius of the obstacle avoidance sphere*. Collision is possible if the following conditions are satisfied:

$$|\overrightarrow{d_{\rm rel}}| \leq d_{min} \text{ and } \dot{r} < 0 \tag{2}$$

In addition to the above conditions, collision is certain if time to reach the goal t_g is greater than the time of collision t_c,

$$t_g = |\vec{u}|^{-1}|\overrightarrow{r_G} - \overrightarrow{r_u}| \qquad t_c = |\overrightarrow{v_{\rm rel}}|^{-1}\left[\sqrt{|\vec{r}|^2 - |\overrightarrow{d_{\rm rel}}|^2} - \sqrt{d_{min}^2 - |\overrightarrow{d_{\rm rel}}|^2}\right] \tag{3}$$

A geometry-based collision avoidance algorithm that avoids the detected dynamic obstacle is proposed. It is assumed that the speed of the robot remains constant throughout the avoidance maneuver. Let us consider that the robot takes a turn with *minimum turn radius*, ρ for time t_0 with a constant speed $|\vec{u}|$ and avoids the incoming obstacle. The relation between the angle subtended at the centre φ, minimum turn radius ρ and t_0 is:

$$\varphi = \frac{|\vec{u}|t_0}{\rho} \tag{4}$$

Let A_{t_0},B_{t_0} be the respective positions of the robot and the obstacle respectively, after time t_0. As shown in Fig. 1b the backward extension of $\overrightarrow{u_{\rm new}}$ intersects the original trajectory at A. If the robot kept moving with the same velocity, and had it travelled to A, then the time taken to reach A would have been,

$$t = \frac{AA_{t_0}}{|\vec{u}|} = \frac{\rho\tan(\varphi/2)}{|\vec{u}|} \tag{5}$$

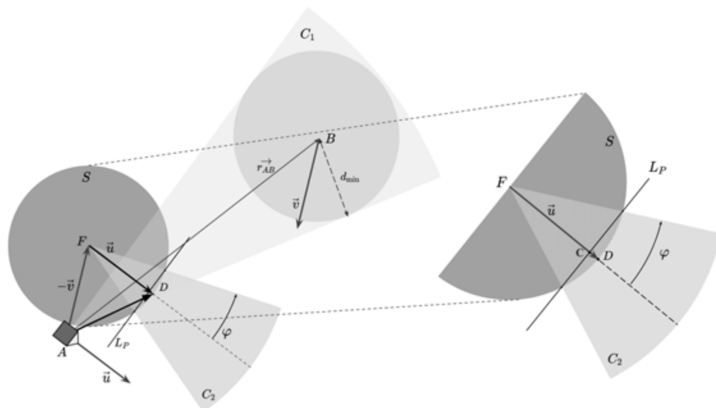


Fig. 2: Collision avoidance formulation

Hence position vectors of virtual positions $\overrightarrow{r_A}$ and $\overrightarrow{r_B}$ are dependant on t, which in-turn is a function of φ and can be calculated as

$$\overrightarrow{r_A} = \overrightarrow{r_u} + \overrightarrow{u}.t = \overrightarrow{r_u} + \rho\overrightarrow{u}\left[\frac{\tan(\varphi/2)}{|\overrightarrow{u}|}\right] \tag{6}$$

$$\overrightarrow{r_B} = \overrightarrow{r_v} + \overrightarrow{v}.(t_0 - t) = \overrightarrow{r_v} + \rho\overrightarrow{v}\left[\frac{\varphi - \tan(\varphi/2)}{|\overrightarrow{u}|}\right] \tag{7}$$

Hence the problem statement can be restated as follows: *"Given that the robot and the obstacle are at virtual positions, A and B, with velocities $\overrightarrow{u}$ and $\overrightarrow{v}$, respectively, find the instantaneous change in the velocity vector $\overrightarrow{u}$ to $\overrightarrow{u_{\text{new}}}$ required to avoid the collision"*

In Fig. 2, points A and B are the virtual position of the robot ($\overrightarrow{r_A}$) and obstacle ($\overrightarrow{r_B}$), respectively. $\overrightarrow{AF}$ is such that its direction is opposite to $\overrightarrow{v}$ and magnitude is the same as $\overrightarrow{v}$. Similarly, we have $\overrightarrow{FD}$ whose magnitude is $|\overrightarrow{u}|$ and the direction is the same as $\overrightarrow{u}$. The resultant of the vectors $\overrightarrow{AF}$ and $\overrightarrow{FD}$, i.e. $\overrightarrow{AD}$, gives the sense of the relative velocity between the robot and the obstacle. Hence, the position vectors of the points, D and F, can be given as

$$\overrightarrow{r_D} = \overrightarrow{r_A} + \overrightarrow{AD} \quad \text{and} \quad \overrightarrow{r_F} = \overrightarrow{r_A} + \overrightarrow{AF} \tag{8}$$

C_1 is the collision cone and any solution that moves the relative velocity vector outside the cone will avoid the collision. However, optimal conflict resolution is provided if the new relative velocity vector is tangential to the collision cone $C1$ [8]. A circle, S centered at F is defined with the radius equalling to the magnitude of $\overrightarrow{u}$. The vector joining A to any point on the circle, S, represents a possible configuration of the relative velocity vector with constant robot speed. The cone, C_2 is constructed such that the new velocity vector, $\overrightarrow{u_{\text{new}}}$, makes an angle, φ, with $\overrightarrow{u}$. Since the vertex of the cone lies at the center of the circle, the intersection is a straight line, say L_P. The slope of L_P is perpendicular to $\overrightarrow{u}$ and it passes through a point, $\overrightarrow{r_C}$ given as

$$\overrightarrow{r_C} = \overrightarrow{r_F} + (1 - cos\varphi)\overrightarrow{FD} \tag{9}$$

Equations of the different curves shown in Fig. 2 are,

$$L_P : u_x(x - x_C) + u_y(y - y_C) = 0 \tag{10}$$

$$S : (x - x_F)^2 + (y - y_F)^2 - |\overrightarrow{u}|^2 = 0 \tag{11}$$

$$C_1 : (|\overrightarrow{r_{AB}}|^2 - d_{min}^2)[(x')^2 + (y')^2] - [x_{AB}(x') + y_{AB}(y')]^2 = 0 \tag{12}$$

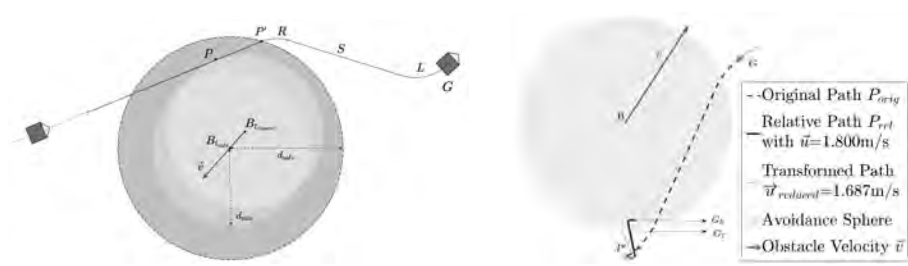
where, $x' = x - x_A$, $y' = y - y_A$, $\overrightarrow{r_{AB}} = \overrightarrow{r_B} - \overrightarrow{r_A}$, x_{AB} and y_{AB} are the x and y components of $\overrightarrow{AB}$, and u_x, u_y are the x and y components of $\overrightarrow{u}$. $\overrightarrow{r_F}$ and $\overrightarrow{r_C}$ can be computed using (8) and (9). It is now required to find out the point $Q(x, y)$ which satisfies constraints (10), (11) and (12) and minimizes the change in $\overrightarrow{u}$, i.e. the point is nearest to D. Hence this can be formulated as a multi-variable optimization problem as follows:

$$\min_{x,y,\varphi} f = \sqrt{(x - x_D)^2 + (y - y_D)^2} \qquad \text{s.t. } S = 0, C_1 = 0, L_P = 0, \varphi > 0 \tag{13}$$

where (x_D, y_D) is the position of point D. Using x and y components obtained from the above optimization problem, we can find $\overrightarrow{u_{\text{new}}} = (x - x_F, y - y_F)$, where (x_F, y_F) is the position of point F. The duration for which turning takes place can be obtained using (4), where φ is calculated from (13).

4 Path Re-planning

The collision avoidance maneuver is complete when the robot passes the point of the closest approach with the obstacle. Let this point be P. It now has to re-plan



(a) Illustration of the safe re-planning maneuver with $r_{\text{safe}} = 1.4$. Re-planning begins after reaching a distance d_{safe} away from the obstacle at point P' and time t_{safe}.

(b) Illustration of the re-planning algorithm in case of re-collision with $|\vec{u}| = 1.8$m/s. G_R and G_T are the transformed goal poses with $|\overrightarrow{u}_{\text{in}}| = 1.8$m/s and 1.687m/s ($a_{rp} = 3m/s^2$)

Fig. 3: Re-planning Maneuver

its path back to the final goal. If the re-planning maneuver starts as soon as it finishes its collision avoidance phase, there is a possibility that the re-planned trajectory may intersect with the obstacle, leading to a re-collision. Hence, to minimize the probability of this event, a design parameter has been proposed which is the ratio of the *safe re-planning distance* (the safety distance between the obstacle and the robot after which the re-planning maneuver begins), d_{safe} and the *radius of the obstacle avoidance sphere*, d_{min}: $r_{\text{safe}} = \frac{d_{\text{safe}}}{d_{\text{min}}}$.

After reaching a distance d_{safe} away from the obstacle at the point, P', the robot then plans a Dubins-like path [5] back to the goal point as illustrated in Fig. 3a. The Dubins path provides the shortest path between any two poses of a robot with a bounded turn radius. The procedure for generating Dubins paths has been described in [26] and [15]. The optimal Dubins path, P_{orig} is generated maintaining a constant speed to reach the goal $X_g : [x_g, y_g, \theta_g]$.

In an unlikely event of the re-planned path coming on the way of the obstacle, a strategy which involves lowering the speed of the robot until the collision is re-avoided is proposed. The approach of re-planning a longer path to the goal is avoided due to the increased actuation cost. Re-collision with the obstacle is checked by projecting the original Dubins path P_{orig} to the obstacle frame. To carry out this projection, we sample points P_{orig_i} (an array of sampled points) on the path. During sampling, the path corresponding to motion primitive S (straight line) can be sampled by just at its start and end points. For L (Left turn with the minimum turn radius) and R (Right turn with the minimum turn radius) motion primitives, a hyper parameter (sample density, s_D) which represents samples per unit length of the curved path, is used to uniformly sample the points on the curve. We assume the robot velocity decreases from $\overrightarrow{|u|}$ to $\overrightarrow{|u|}_{\text{in}}$ with a deceleration, a_{rp}. Hence we define $\overrightarrow{|u|}_i$ as

$$\overrightarrow{|u|}_i = \begin{cases} \sqrt{\overrightarrow{|u|}^2 + \frac{2a_{rp}i}{s_D}} & i \leq \left[\frac{s_D(\overrightarrow{|u|}_{\text{in}}^2 - \overrightarrow{|u|}^2)}{2a_{rp}}\right] \\ \overrightarrow{|u|}_{\text{in}} & \text{otherwise} \end{cases} \tag{14}$$

Then the relative path, P_{rel}, in the obstacle fixed frame can be obtained:

$$\overrightarrow{P_{\text{rel}_i}} = \overrightarrow{P_{\text{orig}_i}} - \overrightarrow{v}\left[\frac{PathDist(\overrightarrow{P_{\text{orig}_i}})}{\overrightarrow{|u|}_i}\right] \tag{15}$$

Obstacle velocity $\overrightarrow{v}$, final robot speed $\overrightarrow{|u|}_{\text{in}}$ and the original sampled path points P_{orig_i} are the inputs to this equation. $PathDist(\overrightarrow{P_{\text{orig}_i}})$ gives the distance between the initial and ith point along the path. It can be seen that for larger $\overrightarrow{|u|}_{\text{in}}$ values, the relative path will almost be the same as the original path, and for a smaller $\overrightarrow{|u|}_{\text{in}}$ value, points further away in the path will deviate by a large amount along $\overrightarrow{v}$. The strategy for finding an optimal lower speed $|\overrightarrow{u}|_{\text{reduced}}$ is elaborated in Algorithm 1. Fig. 3b illustrates this procedure.

Algorithm 1: Path Re-planning

$P_{\text{orig}} \leftarrow$ Plan Dubins path between P' and G;
$P_{\text{rel}} \leftarrow$ *TransformPath*$(P_{\text{orig}_i}, |\overrightarrow{u}|_{\text{in}} = |\overrightarrow{u}|)$;
if *CheckCollision*(P_{rel})=*true* **then**
  Binary search in $(0, |\overrightarrow{u}|)$ to obtain $|\overrightarrow{u}|_{\text{reduced}}$ for a fixed a_{rp};
  Decrease speed to $|\overrightarrow{u}|_{\text{reduced}}$ by decelerating with a_{rp} while following P_{orig};
else
  Follow P_{orig} with speed $|\overrightarrow{u}|$;
end

5 Simulation and Analysis

5.1 Implementation and Trajectory Visualization

The optimization problem described in (13) can be solved using an Interior Point Algorithm. During the analysis, it is found that a randomized initial condition fails to converge to the optimal point in many cases. Hence we run multiple instances of the solver parallelly, and report back the most optimum solution. This addition greatly improves the accuracy of the algorithm while keeping the computation time the same due to parallel implementation. An i7-7700HQ 8GB RAM machine is used to carry out all simulations.

For visualizing the planned path of the robot, a random simulation with the parameters shown in Table 1 is performed. Figure 4 illustrates the various aspects

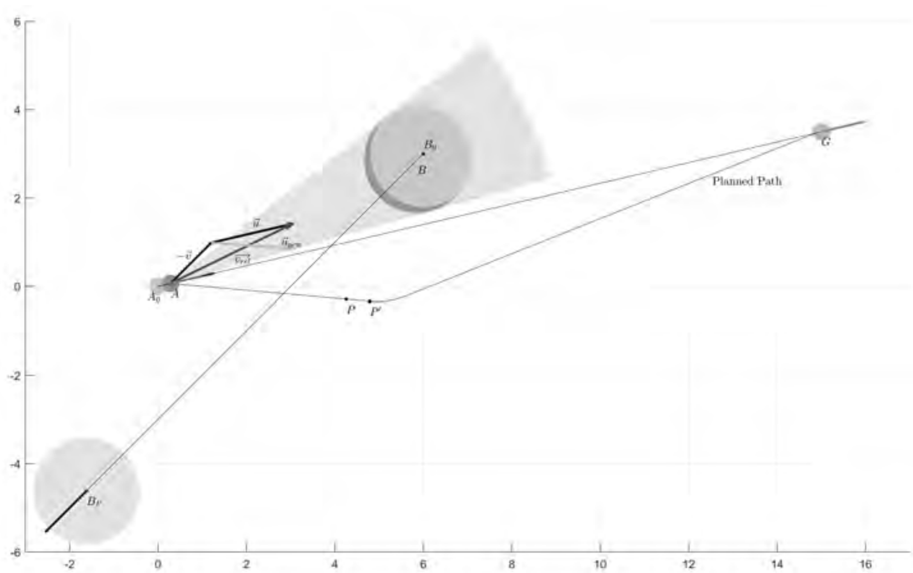


Fig. 4: Trajectory visualization with parameters as mentioned in Table 1

Table 1: Parameters for trajectory visualization

Goal State X_g	$[15\text{ m}, 3.5\text{ m}, 13.124°]$
Robot initial state X_0	$[0, 0, 13.124°]$
Robot initial velocity $\overrightarrow{u_0}$	$[1.85, 0.431]$ m/s
Minimum turn radius ρ	1.8 m
Obstacle avoidance distance d_{min}	1.2 m
Obstacle initial position B_0	$[6, 3]$ m
Obstacle velocity $\overrightarrow{v}$	$[-0.92, -0.92]$ m/s
Re-planning safe distance ratio r_{safe}	1.25
Path sample density s_D	100 pts/m

of this simulation. B_0 is the position of the obstacle when it is initially detected. The optimizer outputs a turning time $t_0 = 0.301$s with a $\overrightarrow{u_{\text{new}}} = [1.893, -0.168]$ m/s. The corresponding instantaneous change virtual positions are marked as A and B, and the collision cone with various vectors at this position is shown in the figure. The robot performs a smooth turn for t_0 seconds to change its velocity from $\overrightarrow{u}$ to $\overrightarrow{u_{\text{new}}}$. It continues with the same velocity till it reaches its point of the closest approach at P. The re-planning maneuver begins after reaching a safe distance d_{safe} away from the obstacle at P'. Finally the robot plans a Dubins path to the final goal, and as there is no re-collision, it traverses this path with the same speed and reaches the goal at $t = 8.266$ s.

5.2 Monte Carlo Simulation

Monte carlo analysis has been used to validate the proposed algorithm. Consider the setup shown in Fig 5. Two simulation setups with parameters are as shown in Table 2. Parameters have been chosen such that Set 1 is slightly more aggressive

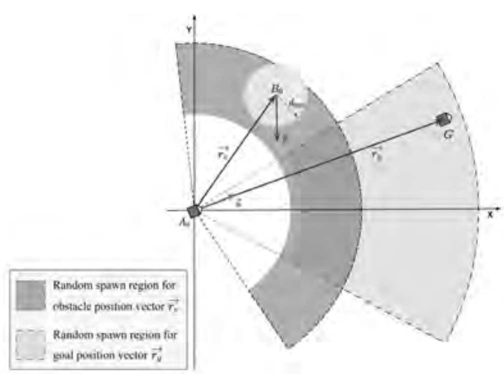


Fig. 5: Setup for randomized simulation

Table 2: Parameter variation with simulation results for randomized simulation

Parameter		**Set 1**	**Set 2**
Goal State	$X_g = \begin{bmatrix} r_{X_g} \\ \theta_{X_g} \end{bmatrix}$	$\begin{bmatrix} U(20,40) \text{ m} \\ U(-70,70)^\circ \end{bmatrix}$	$\begin{bmatrix} U(60,100) \text{ m} \\ U(-80,80)^\circ \end{bmatrix}$
Robot			
Robot initial state	$X_0 = \begin{bmatrix} X_{x_0} \\ X_{y_0} \\ X_{\theta_0} \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0 \\ \theta_{X_g} \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0 \\ \theta_{X_g} \end{bmatrix}$
Robot initial velocity	$\vec{u_0} = \begin{bmatrix} \lvert\vec{u_0}\rvert \\ \angle\vec{u_0} \end{bmatrix}$	$\begin{bmatrix} U(1.0,2.5) \text{ m/s} \\ \theta_{X_g} \end{bmatrix}$	$\begin{bmatrix} U(1.2,3.5) \text{ m/s} \\ \theta_{X_g} \end{bmatrix}$
Minimum turn radius	ρ	$U(0.8,1.2)$ m	$U(1.2,1.5)$ m
Obstacle			
Avoidance distance	$d_{\min}$	$U(1.2,3)$ m	$U(1.8,3.5)$ m
Initial position	$B_0 = \begin{bmatrix} r_{B_0} \\ \theta_{B_0} \end{bmatrix}$	$\begin{bmatrix} U(15,50) \text{ m} \\ \theta_{X_g} + U(-60,60)^\circ \end{bmatrix}$	$\begin{bmatrix} U(35,70) \text{ m} \\ \theta_{X_g} + U(-70,70)^\circ \end{bmatrix}$
Obstacle Velocity	$\vec{v} = \begin{bmatrix} \lvert\vec{v}\rvert \\ \angle\vec{v} \end{bmatrix}$	$\begin{bmatrix} U(1.5,3.5) \text{ m/s} \\ U(-180,180)^\circ \end{bmatrix}$	$\begin{bmatrix} U(2.2,4) \text{ m/s} \\ U(-180,180)^\circ \end{bmatrix}$
Safe re-planning ratio	r_{safe}	1.25	1.25
Path sample density	s_D	100 pts/m	100 pts/m
Simulation Results			
Success / Number of Simulations		6901 / 7000	6944 / 7000
Collisions / Optimization failures		5 / 94	11 / 45
Velocity deviation ($V_{D_{\max}}, V_{D_{\text{avg}}}$)		(0.6775,0.1378)	(0.6117, 0.0986)
Path deviation ($P_{D_{\max}}, P_{D_{\text{avg}}}$)		(1.4121,1.0174)	(1.0942,1.0038)
Accuracy		**98.58%**	**99.20%**

than Set 2. Two metrics used for comparative analysis are given below:

$$\text{Path deviation } P_D = \frac{PathDist(\text{Planned Path})}{PathDist(\text{Initial Path})} \tag{16}$$

$$\text{Velocity deviation } V_D = \frac{|\overrightarrow{u_{\text{new}}} - \vec{u}|}{|\vec{u}|} \tag{17}$$

A total of 7000 random simulations were run on each set of parameters. The direction of velocity of the obstacle was chosen randomly such that collision with the robot was certain. Table 2 also presents the results obtained. Optimization failures represent conditions where the turn radius of the robot was not sufficient to avoid collision going with constant speed. It is observed an accuracy of over 98.5% in set 1 and over 99% in set 2. (It was found only 2 of the total 14000 simulation run required the speed lowering re-planning maneuver, hence 99.986% of the initial re-planned paths are collision free).

6 Conclusion and Future Work

A geometry-based strategy for generating a smooth trajectory avoiding dynamic obstacles has been presented in this paper. A novel re-planning approach has been proposed which generates the shortest path to the goal while avoiding re-collision with the obstacle. The proposed algorithm has been validated by conducting 14,000 random simulations and an average accuracy of 98.89% has been obtained. The proposed algorithm can be extended to consider vehicles dynamics, and non-linear controllers like back-stepping or sliding mode control can be developed to track the generated path. It can also be extended to irregularly shaped obstacles. A strategy to avoid multiple collisions can be devised by introducing avoidance hierarchies based on obstacle speeds and collision times.

References

1. Carbone, C., Ciniglio, U., Corraro, F., Luongo, S.: A novel 3d geometric algorithm for aircraft autonomous collision avoidance. In: Proceedings of the 45th IEEE Conference on Decision and Control. pp. 1580–1585. IEEE (2006)
2. Chakravarthy, A., Ghose, D.: Obstacle avoidance in a dynamic environment: A collision cone approach. IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans **28**(5), 562–574 (1998)
3. Chen, Z.: On dubins paths to a circle. Automatica **117**, 108996 (2020)
4. Deng, M., Inoue, A., Sekiguchi, K.: Lyapunov function-based obstacle avoidance scheme for a two-wheeled mobile robot. Journal of Control Theory and Applications **6**(4), 399–404 (2008)
5. Dubins, L.E.: On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. American Journal of mathematics **79**(3), 497–516 (1957)
6. Fan, X., Guo, Y., Liu, H., Wei, B., Lyu, W.: Improved artificial potential field method applied for auv path planning. Mathematical Problems in Engineering **2020** (2020)
7. Fox, D., Burgard, W., Thrun, S.: The dynamic window approach to collision avoidance. IEEE Robotics & Automation Magazine **4**(1), 23–33 (1997)
8. Goss, J., Rajvanshi, R., Subbarao, K.: Aircraft conflict detection and resolution using mixed geometric and collision cone approaches. In: AIAA guidance, navigation, and control conference and exhibit. p. 4879 (2004)
9. Hota, S., Ghose, D.: Optimal geometrical path in 3d with curvature constraint. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 113–118. IEEE (2010)
10. Hota, S., Ghose, D.: Optimal path planning for an aerial vehicle in 3d space. In: 49th IEEE Conference on Decision and Control (CDC). pp. 4902–4907. IEEE (2010)
11. Hoy, M., Matveev, A.S., Savkin, A.V.: Algorithms for collision-free navigation of mobile robots in complex cluttered environments: a survey. Robotica **33**(3), 463–497 (2015)
12. Iswanto, A.M., Wahyunggoro, O., Cahyadi, A.I.: Artificial potential field algorithm implementation for quadrotor path planning. Int. J. Adv. Comput. Sci. Appl **10**(8), 575–585 (2019)

13. Jenie, Y.I., van Kampen, E.J., de Visser, C.C., Ellerbroek, J., Hoekstra, J.M.: Three-dimensional velocity obstacle method for uncoordinated avoidance maneuvers of unmanned aerial vehicles. Journal of Guidance, Control, and Dynamics **39**(10), 2312–2323 (2016)
14. Jha, B., Chen, Z., Shima, T.: On shortest dubins path via a circular boundary. Automatica **121**, 109192 (2020)
15. LaValle, S.M.: Planning algorithms. Cambridge University Press (2006)
16. LaValle, S.M., et al.: Rapidly-exploring random trees: A new tool for path planning (1998)
17. Luongo, S., Carbone, C., Corraro, F., Ciniglio, U.: An optimal 3d analytical solution for collision avoidance between aircraft. In: 2009 IEEE Aerospace conference. pp. 1–9. IEEE (2009)
18. Luongo, S., Corraro, F., Ciniglio, U., Di Vito, V., Moccia, A.: A novel 3d analytical algorithm for autonomous collision avoidance considering cylindrical safety bubble. In: IEEE Aerospace Conference. pp. 1–13. IEEE (2010)
19. Mahjri, I., Dhraief, A., Belghith, A.: A review on collision avoidance systems for unmanned aerial vehicles. In: International Workshop on Communication Technologies for Vehicles. pp. 203–214. Springer (2015)
20. Moon, J., Lee, B.Y., Tahk, M.J.: A hybrid dynamic window approach for collision avoidance of vtol uavs. International Journal of Aeronautical and Space Sciences **19**(4), 889–903 (2018)
21. Mujumdar, A., Padhi, R.: Nonlinear geometric and differential geometric guidance of uavs for reactive collision avoidance. Tech. rep., INDIAN INST OF SCIENCE BANGALORE (INDIA) (2009)
22. Rajko, S., LaValle, S.: A pursuit-evasion bug algorithm. In: Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (2001)
23. Rostami, S.M.H., Sangaiah, A.K., Wang, J., Liu, X.: Obstacle avoidance of mobile robots using modified artificial potential field algorithm. EURASIP Journal on Wireless Communications and Networking **2019**(1), 1–19 (2019)
24. Seder, M., Petrovic, I.: Dynamic window based approach to mobile robot motion control in the presence of moving obstacles. In: Proceedings 2007 IEEE International Conference on Robotics and Automation. pp. 1986–1991. IEEE (2007)
25. Shanmugavel, M., Tsourdos, A., White, B.A.: Collision avoidance and path planning of multiple uavs using flyable paths in 3d. In: 2010 15th International Conference on Methods and Models in Automation and Robotics. pp. 218–222. IEEE (2010)
26. Shkel, A., Lumelsky, V.: Classification of the dubins set. Robotics Auton. Syst. **34**, 179–202 (2001)
27. Tan, C.Y., Huang, S., Tan, K.K., Teo, R.S.H.: Three dimensional collision avoidance for multi unmanned aerial vehicles using velocity obstacle. Journal of Intelligent & Robotic Systems **97**(1), 227–248 (2020)
28. Williams, R.L., Wu, J.: Dynamic obstacle avoidance for an omnidirectional mobile robot. Journal of Robotics **2010** (2010)
29. Yang, H., Fan, X., Shi, P., Hua, C.: Nonlinear control for tracking and obstacle avoidance of a wheeled mobile robot with nonholonomic constraint. IEEE Transactions on Control Systems Technology **24**(2), 741–746 (2015)
30. Zhu, Q., Yan, Y., Xing, Z.: Robot path planning based on artificial potential field approach with simulated annealing. In: Sixth International Conference on Intelligent Systems Design and Applications. vol. 2, pp. 622–627. IEEE (2006)