Motion of Mobile Robots in Environments with Dynamic Obstacles and Arbitrary Directions

Maram Ali
Centre for Environmental Mathematics,
Faculty of Environment, Science and Economy,
University of Exeter, Penryn Campus,
Cornwall TR10 9FE, United Kingdom.
Email: ma935@exeter.ac.uk

Saptarshi Das, *Member, IEEE*
Centre for Environmental Mathematics,
Faculty of Environment, Science and Economy,
University of Exeter, Penryn Campus,
Cornwall TR10 9FE, United Kingdom.
Email: S.Das3@exeter.ac.uk, saptarshi.das@ieee.org

***Abstract*—This paper presents an improved study on the motion of mobile robots with dynamic obstacle environments and arbitrary directions. This study focuses on incorporating the concept of inertia into the movement of obstacles to enhance the capabilities of mobile robots in complex environments. Unlike random movements, the obstacles in this study possess inertia, which constrains their motion in predictable patterns. This inertia can be learned or predicted by the robot, enabling it to better anticipate the obstacle positions. This research employs a grid-based simulation environment with systematically moving obstacles. By considering inertia, the robot gains the ability to understand and leverage the predictable aspects of obstacle motion, resulting in improved navigation performance. The robot can predict obstacle trajectories more effectively, reducing the likelihood of collisions and increasing overall efficiency by using the velocity obstacle algorithm. By incorporating inertia into the movement of obstacles, the robot gains valuable insights that enable it to plan its movements more intelligently. Incorporating inertia as a factor in obstacle motion contributes to a more systematic and predictable environment, allowing the robot to make informed decisions based on the anticipated positions of both fixed and moving obstacles. This research opens up possibilities for further advancements in mobile robot navigation in complex environments and dynamic scenarios.**

***Index Terms*—Mobile robots, dynamic obstacles, arbitrary direction, grid environment, inertia, velocity obstacle.**

I. Introduction

The significance of mobile robots has surged across all facets of life. These machines have become increasingly intelligent, enabling them to operate autonomously in dynamic and unpredictable environments [1]. Research in robot development can be categorized into four main areas: motion control, obstacle avoidance, path planning, and localization [1]. Path planing involves the mobile robot detecting the boundaries of obstacles, devising alternative paths, and calculating its instantaneous velocity and angular heading. This allows the robot to avoid collisions while in motion [1] . The bug algorithm is a well-known technique used for collision avoidance. It has been used since the early days as one of the approaches to address this problem [2]. In the bug algorithm, the first step is to detect the presence of obstacles. Once obstacles are detected, the robot subsequently navigates along the boundaries of these objects as it moves, ensuring collision avoidance. This algorithm is only effective in a stationary environment. An additional approach that could be considered is the potential field method [3]. The global path planning method, also referred to as potential field navigation, considers the target and obstacles as valleys and hills in a highland region, where the lowest and highest values represent the potential gravity. To avoid obstacles, the robot maneuvers through the repulsive field, while simultaneously moving towards the goal using the attractive field. Despite these advantages, this algorithm still has limitations and struggles in narrow passages. Additionally, the bug algorithm itself has certain drawbacks, and its implementation can be challenging in real-time systems. In this paper, we present a study on mobile robots with arbitrary directions. This study has been implemented in a grid environment with a more systematic movement of obstacles. Instead of obstacles that move randomly, they have inertia which constrains their motion. This can then be learned or predicted by the robot.

979-8-3503-0551-7/23/$31.00 ©2023 IEEE

II. Related Previous Works

It's crucial for mobile robots to navigate through dynamic environments while avoiding obstacles to reach their destination safely [1] . One approach is the motion saliency method, which calculates the saliency of dynamic objects and identifies dangerous ones using segmentation [4]. By predicting the movement of these objects through B-spline curves, the robot can use nonlinear model predictive control to steer clear of potential collisions [4]. In unknown dynamic environments, a robot can detect its surroundings, gather distance information, and generate trajectories in real-time. One such method is the hybrid fuzzy potential field, which combines fuzzy logic and potential field techniques to achieve autonomous motion planning [5], [6]. The Velocity Obstacle algorithm is known for its remarkable efficiency in local path planning. It enables the incorporation of dynamic and unforeseen obstacles into the planning process [7]. By doing so, the robot can effectively navigate and avoid collisions by choosing its velocity exclusively from the collision-free region. [8] However, when comparing the cumulative time taken based on the minimum

distance, it became evident that VO was significantly safer than APF. Moreover, the VO method also completed the formation faster, as compared to APF [9]. The authors in [1] introduced the VO method for motion planning in dynamic environments and showed its effectiveness in avoiding collisions with moving obstacles in real time. Another collision avoidance method is based on motion saliency for a dynamic environment [2]. In this method, segmented dynamic objects are used to calculate the saliency of dynamic objects and segment dangerous dynamic objects. The B-spline curve is a powerful tool for forecasting the trajectory of hazardous, dynamic objects. When combined with a nonlinear model predictive control method, it becomes an effective approach to navigating around perilous obstacles in a robot's dynamic environment. [4]. In unknown dynamic environments, a robot would detect environments, incrementally obtain distance information of its current state and obstacles and then generate a trajectory online [10]. The utilisation of a hybrid fuzzy potential field has been proposed in mobile robot dynamic motion planning [10]. The Velocity Obstacle (VO) method is a popular and effective algorithm for local path planning that allows for the consideration of dynamic and unexpected obstacles [7].

III. Research Objectives

- Developing a systematic movement algorithm using MATLAB environment.
- Prove the effectiveness of the proposed method by reducing collision.
- Use some performance metrics such as number of collisions, time, and path efficiency.

As compared to our previous study [11], there are several aspects which differentiate it from the current study. The key differences are summarized as the first study on the A* pathfinding algorithm as:

- Implement a variation of the A* algorithm on a 15 by 15 grid using Python.
- Test the Maximum Velocity Obstacle (MVO) algorithm for dynamic obstacle avoidance.
- Demonstrates that the MVO algorithm is efficient and reliable for grid-based environments.
- Suggests the possibility of optimizing the algorithm further by combining proposed method with other pathfinding techniques like artificial neural networks.

While our study on motion mobile robots with inertia in dynamic obstacle environments:

- Explores the concept of incorporating inertia into the movement of obstacles to enhance the capabilities of mobile robots.
- Uses MATLAB to implement the study on a larger 50 by 50 grid setting.
- Introduces the idea of obstacles possessing inertia, which constrains their motion in predictable pattern.
- Shows that by considering inertia, the robot gains the ability to leverage predictable aspects of obstacle motion.
- Implements the Velocity Obstacle Algorithm to improve navigation performance and reduce collisions.
- Emphasizes the potential of inertia in enhancing mobile robot performance in dynamic obstacle environments.
- In this study, we add moving and fixed obstacles to the experiment.
- Indicates the implications for real-world applications like search and rescue missions and navigating cluttered environments. Both share a common goal of improving the performance and efficiency of robots in navigating through complex and changing environments.

IV. Material and Methods

A. Implementing the Grid Environment

The grid environment was designed with a more systematic movement of obstacles. Instead of obstacles that move randomly, they have inertia which constrains their motion. This can then be learned/predicted by the robot.

B. Obstacle Movement Modeling

The obstacle movement was modeled using a B-spline curve which can be employed to anticipate the motion of hazardous and dynamic entities. The obstacles were individually segmented based on their saliency, and the identification process focused specifically on dangerous dynamic objects. The nonlinear model predictive control method was used to circumvent perilous obstructions within the ever-changing surroundings of the robot.

C. Robot Motion Algorithm

The Velocity Obstacle (VO) method is a commonly used and highly effective algorithm in the field of robot motion planning for dynamic obstacle avoidance. It allows a robot to navigate from its initial starting position to its intended goal position while avoiding collisions with obstacles along the way. The Velocity Obstacle (VO) method operates by establishing a collection of velocities for a robot. These velocities are identified as those which, if pursued, would lead to a collision with another robot or obstacle if it keeps moving with its existing velocity. By selecting a velocity within the velocity obstacle, the two robots, or the robot and obstacle, will ultimately collide. Conversely, if a velocity is chosen outside the velocity obstacle, it ensures that no collision will occur. This approach helps in navigating safely while considering potential collisions. One of the reasons why the VO method is a preferred choice for robot motion planning is its simplicity and good real-time performance. It allows for quick and efficient calculations to be made in real-time, enabling the robot to effectively navigate around obstacles and avoid collisions. The VO method has numerous applications in robotics and motion planning. It can be used for autonomous navigation of robots in dynamic environments, where obstacles may be constantly moving or changing. It can also be used for multi-robot systems where multiple robots need to coordinate their movements to avoid collisions with each other.

D. Simulation Runs and Data Collection

Simulation runs were conducted to collect data on the robot's motion behavior and obstacle avoidance strategies. The data was then analyzed to evaluate the performance of the proposed method in terms of collision avoidance and path efficiency.

E. Mathematical Model of Velocity Obstacle Method

Velocity Obstacle Algorithm: The method involves creating velocity obstacles [1], [4], generating velocity obstacles by taking into account the present positions and velocities of both the robot and the obstacles. To derive the equation for the Velocity Obstacles method, consider a two-dimensional scenario.

Assumptions: The agent and obstacles are point-like objects moving in a two-dimensional space. The agent has a known position and velocity position: (x_a, y_a), velocity: (vx_a, vy_a). Each obstacle has a known position and velocity (position: (x_o, y_o), velocity: (vx_o, vy_o).

Relative Velocity: Defining the relative velocity of an obstacle with respect to the robot as follows:

$$v_{rel_x} = vx_o - vx_a, \tag{1}$$

$$v_{rel_y} = vy_o - vy_a. \tag{2}$$

Velocity Obstacle: The velocity obstacle is a geometric representation that defines the set of velocities that would result in a collision between the agent and the obstacle. It consists of two boundary lines: the tangent line and the acceleration line.

(a) Tangent Line: The tangent line represents the velocities that would result in a collision if the agent and the obstacle continue on their current trajectories. The tangent line is calculated as shown:

$$\text{tangent}_{\text{Slope}} = \frac{\text{v}_{\text{rel}_\text{y}}}{\text{v}_{\text{rel}_\text{x}}}, \tag{3}$$

$$\text{tangent}_{\text{intercept}} = \text{y}_\text{a} - \text{tangent}_{\text{slope}} \times \text{x}_\text{a}. \tag{4}$$

(b) Acceleration Line: The acceleration line represents the velocities that would result in a collision if the agent and the obstacle accelerate maximally. The acceleration line is perpendicular to the tangent line and passes through the position of the agent. Its slope is the negative reciprocal of the tangent line slope:

$$\text{acceleration}_{\text{slope}} = -\frac{1}{\text{tangent}_{\text{slope}}}, \tag{5}$$

$$\text{acceleration}_{\text{inter}} = \text{y}_\text{a} - \text{accelarition}_{\text{slope}} \times \text{x}_\text{a}. \tag{6}$$

Feasible Velocities: To determine the feasible velocities for the agent, we need to consider the space outside the velocity obstacle. Any velocity outside the velocity obstacle guarantees collision avoidance. The equation for the velocity obstacle as follows :

$$\text{velocity}_{\text{obstacle}} = \frac{\text{v}_{\text{rel}_\text{y}}}{\text{v}_{\text{rel}_\text{x}}} * (\text{v}_\text{x} - \text{vx}_\text{a})) + (\text{y}_\text{a} - (\frac{\text{v}_{\text{rel}_\text{y}}}{\text{v}_{\text{rel}_\text{x}}}) * (\text{v}_\text{x} - \text{vx}_\text{a})) \geq 0. \tag{7}$$

This equation represents the upper half-plane of the velocity obstacle. Any velocity (vx, vy) that satisfies this equation ensures collision avoidance.

In MATLAB, the following algorithm was implemented using the above "Velocity Obstacle Method"

1. Defines minimum obstacle avoidance distance:

$$d = \delta. \tag{8}$$

2. Calculates desired velocity vector towards the goal position:

$$v_{\text{desired}} = \frac{p_{\text{goal}} - p_{\text{robot}}}{\| p_{\text{goal}} - p_{\text{robot}} \|}, \tag{9}$$

where, $p_{\text{rob.}} \in R^n, p_{goal} \in R^n$, and

$$||p_{\text{goal}} - p_{\text{robot}}|| = \sqrt{\sum (p_{\text{goal}} - p_{\text{robot}})^2}. \tag{10}$$

3. Adds random acceleration to create variability in the movement:

$$\alpha_{\text{rand}} = N(0,1) * \alpha_{\text{max}}, \tag{11}$$

where, $N(0,1)$ generates a random vector from a multivariate normal distribution with mean 0 and covariance matrix equal to the identity matrix.

4. Updates the robot's velocity based on desired velocity and acceleration:

$$v_{\text{robot}} = v_{\text{robot}} + (v_{\text{desired}} + \alpha_{\text{rand}}) * \Delta t, \tag{12}$$

where, Δt is the time step size.

5. Checks if any obstacles are too close to the robot:

$$\text{too}_{\text{close}} = \Delta < \text{obstacle}_{\text{distances}}. \tag{13}$$

6. If any obstacles are too close, calculate a new velocity vector that avoids them:

$$a_{\text{avoid}} = \sum (p_{\text{obstacle}} - p_{\text{robot}}) * \text{too}_{\text{close}}. \tag{14}$$

This sums up the vectors pointing away from any obstacles that are too close, using -1 or 1 to invert each vector for obstacles that are too close behind versus in front of the robot.

7. Updates the robot's velocity by adding the avoidance vector:

$$v_{\text{robot}} = v_{\text{robot}} + \frac{a_{\text{avoid}}}{||a_{\text{avoid}}||}. \tag{15}$$

8. Updates the position of the robot and all obstacles:

$$p_{\text{robot}} = p_{\text{robot}} + v_{\text{robot}} * \Delta t, \tag{16}$$

$$p_{\text{obstacle}} = p_{\text{obstacle}} + v_{\text{obstacle}} * \Delta t, \tag{17}$$

$$p_{\text{obstacle}} \in R^{nxm}, v_{\text{obstacle}} \in R^{nxm},$$

TABLE I
ROBOT MOVEMENT IN TWO SCENARIOS

Robot Movement			
Scenario	No of collision	Count of iteration to reach goal	Time taken (sec)
with Vo algorithm	3	168	90.37
without algorithm	7	122	66.77

TABLE II
AVERAGE PATH EFFICIENCY IN TWO CASES

Average Path Efficiency	
Case	Average
with VO algorithm	0.9962
without algorithm	0.4878

where, $p_{\text{obstacle}} \in R^{nxm}$ is a matrix containing the positions of all obstacles, and $v_{\text{obstacle}} \in R^{nxm}$ is a matrix containing the velocities of all obstacles.

F. Motions of Obstacles

To derive the discrete approximation of the continuous change in velocity from Newton's second law of motion [12], starting with the equation :

$$F = m * a, \tag{18}$$

where, F represents the net force acting on an object m represents the mass of the object and a represents the acceleration of the object. Rearranging the equation to solve for acceleration:

$$a = \frac{F}{m}. \tag{19}$$

To approximate the change in velocity over a small-time interval dt. Considering an initial velocity v and the final velocity v' after a time interval dt. The average acceleration over this time interval is approximated as:

$$a_{avg} = \frac{v' - v}{dt} \tag{20}$$

Using the relationship between acceleration and net force $F = m * a$, we can substitute a_{avg} into the equation:

$$F = m * a_{avg}, \quad F = m * \frac{v' - v}{dt}. \tag{21}$$

Rearranging the equation to solve for the final velocity:

$$v' = v + \frac{F * dt}{m}. \tag{22}$$

This equation represents the discrete approximation of the continuous change in velocity v' based on the initial velocity v, net force F, mass m, and time step size dt. It is derived from Newton's second law of motion and allows for calculation of the updated velocity over discrete time steps.

V. RESULTS

Based on Table I there are some notable differences between the scenarios with and without the Velocity Obstacle (VO) algorithm: Number of Collisions,and Count of Iterations to Reach Goal. The VO algorithm is designed to help the robot avoid collisions by considering the velocity of the obstacles. In the given scenario including the Velocity Obstacle (VO) algorithm, the robotic system successfully demonstrated its capability to traverse the surrounding area while effectively reducing the occurrence of collisions. Conversely, in the absence of the algorithm, the robot lacked the capability to proactively evade obstacles by considering their velocities, leading to a greater incidence of collisions, specifically amounting to 7. This implies that the virtual obstacle (VO) algorithm has efficacy in mitigating collisions. The count of iterations represents the number of steps or time taken for the robot to reach the goal. In this case, it seems that the scenario without the VO algorithm required fewer iterations to reach the goal compared to the scenario with the algorithm. There are other things that could potentially contribute to this phenomenon.The lack of obstacle avoidance that considers obstacle velocities in the scenario without the algorithm may have enabled the robot to follow more direct routes towards the target, leading to quicker convergence. The VO algorithm, although proficient in the task of collision avoidance, has the ability to include supplementary course deviations or cautious movements in order to prevent prospective collisions. This may lead to an extended trajectory and an increased number of iterations necessary to achieve the objective, thereby resulting in a longer duration.

We calculate the path efficiency using this equation:

$$\text{pathefficiency} = \frac{\text{No of obs} - \text{free grid element}}{\text{Tot No of grid elements}}. \tag{23}$$

The increased path efficiency observed in Table II, as a result of using the VO algorithm, suggests that the robot successfully navigated with greater efficiency while effectively avoiding obstacles. As a result, the robot could reach the goal with minimal deviations and optimized trajectories. On the other hand, the case without the VO algorithm had a lower average path efficiency. This implies that the robot faced more challenges in avoiding obstacles and encountered sub optimal paths. The lack of an obstacle avoidance strategy like the VO algorithm may have resulted in more collisions or inefficient maneuvers, leading to reduced path efficiency.

By comparing the two scenarios, the following differences can be observed: Without inertia, the obstacles' movements appear more erratic and less predictable. They can change direction abruptly at each time step due to the random updates in their positions. This behavior can lead to sudden collisions with other objects or unexpected interactions with the environment. With inertia, the obstacles' movements were smoother and exhibited a sense of

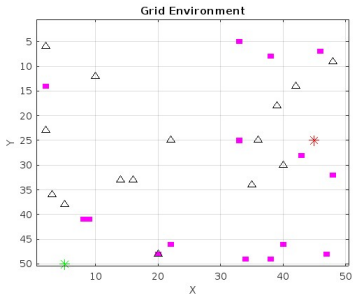

Fig. 1. A grid size of 50 x 50 was set up, containing 15 randomly generated obstacles (colored pink) and 15 fixed obstacles in the form of triangles. The grid includes a designated start position (colored green) and a goal position (colored red) for the robot.

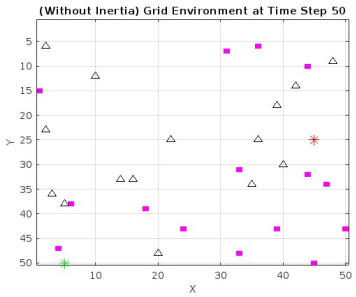

Fig. 2. In this case, the obstacles' positions were updated at each time step by adding a random value between -1 and 1 to their current positions. This means that the obstacles can move in any direction with equal probability. The absence of inertia implies that the obstacles do not have a tendency to maintain their current velocities or resist changes in their motion. As a result, the obstacles can change direction abruptly at each time step.

momentum. They tend to maintain their current velocities and change directions more gradually in response to the applied acceleration. This behavior allows for more controlled and realistic motion, reducing the likelihood of sudden collisions and providing a more natural interaction with the environment.

The proposed method, Motion of Robotic with Velocity

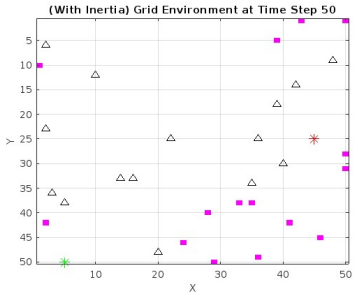

Fig. 3. In this case, the obstacles' positions and velocities are updated using inertia. The obstacles experience a random acceleration at each time step, which affects their velocities. The velocities, in turn, influence the positions of the obstacles. The inertia provides a sense of momentum to the obstacles, making their motion smoother and more continuous compared to the case without inertia. The obstacles tend to maintain their current velocities and gradually change their directions based on the applied acceleration.

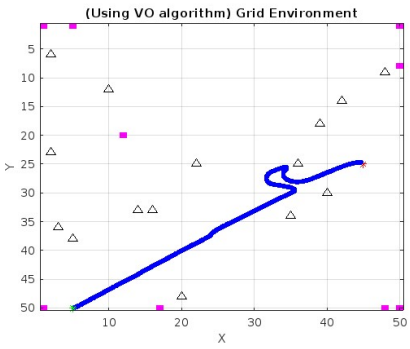

Fig. 4. The robot's motion path (blue) was simulated from its initial starting position to its intended goal position. During the simulation, the robot demonstrated its capabilities by effectively navigating around obstacles and avoiding any collisions.

TABLE III
COMPARISON WITH PREVIOUS STUDIES

Ref	Motion Planing Algorithm	Obstacle Handling Method	Performance Matrix
[3]	Artificial potential fields	Repulsive forces	Computational efficiency
[13]	Sampling based motion planing	Probabilistic road map	Planing time
[14]	Cooperative multi_robot path	Rapidly exploring random trees	Coordination efficiency
[15]	Theory of mind for humanoid robot	Mental state inference	Human interaction capability
[16]	Ant colony optimization	Pheromone based communication	Convergence speed
[17]	Random walk search strategy	fixed obstacle	Time to explore the area

Obstacle Algorithm, demonstrates similarities with the compared methods in terms of motion planning and obstacle handling. However, it distinguishes itself in several aspects, such as its unique motion planning algorithm, obstacle handling approach, and the performance metric. Specifically, the method incorporates the Velocity Obstacle algorithm to consider obstacle velocities and emphasizes dynamic obstacle avoidance for single robot navigation. The performance metric employed in this study is the time it takes for the robot to reach its destination.

VI. DISCUSSIONS

The algorithm works by first computing the Velocity Obstacle for each obstacle in the robot's environment [3]. The robot determines a preferred velocity that enables it to move towards its goal while efficiently avoiding collisions with obstacles. The algorithm checks if the preferred velocity falls inside any of the Velocity Obstacles computed earlier. If it does, the robot must select a new velocity that avoids the obstacle.

In the given scenario, a grid of size 50x50 was initialized with 15 randomly generated obstacles with fixed obstacles, including the start and goal positions for the robot . The obstacles were then made to move with the

inertia that constrained their motion for a time step of 50. The robot's motion was simulated from its initial starting position to its intended goal position using the Velocity Obstacle (VO) method.

During the simulation, the robot demonstrated its advanced capabilities by effectively navigating around obstacles and avoiding any collisions. The effective execution of the simulation serves as evidence of the robot's advanced design and programming.The robot's ability to reach its goal position while avoiding obstacles is a crucial step in ensuring its real-world functionality and effectiveness [3], [18]. The utilization of the velocity obstacle (VO) technique in this particular context facilitated rapid and fast computations in real-time, empowering the robot to adeptly maneuver amidst dynamic impediments and evade potential collisions. The proficient implementation of the Velocity Obstacle (VO) approach in this particular context illustrates its viability for application in various other practical situations that necessitate the ability to navigate around moving obstacles. This encompasses the independent navigation of robots in dynamic settings characterised by moving or changing obstacles, as well as the coordination of motions among many robots in order to prevent collisions.

VII. Conclusion and Future Work

Based on the simulation, the velocity obstacle method is effective in avoiding moving obstacles. The results of this simulation demonstrate the practicality and efficiency of the VO method in ensuring safe and efficient motion planning for the robot. This geometric method of collision avoidance enables quick and efficient computations to be conducted in real-time, enabling the robot to effectively navigate around obstacles and avoid collisions. The possible applications of the VO approach extend beyond this scenario to include the autonomous navigation of robots in dynamic environments and multi-robot systems where many robots need to coordinate their movements. The VO method is a powerful tool for robot motion planning and has the potential to revolutionize the field of robotics. For future work in the field of robotics and motion planning, using the VO method could include investigating the use of the VO method in multi-robot systems to enable coordination and collision avoidance between multiple robots [19], [20]. Also, exploring the integration of the VO method with other approaches [21], and collision avoidance to develop hybrid methods that can leverage the strengths of multiple approaches.

References

[1] A.-T. Nguyen and C.-T. Vu, "Obstacle avoidance for autonomous mobile robots based on mapping method," in *Proceedings of the International Conference on Advanced Mechanical Engineering, Automation, and Sustainable Development 2021 (AMAS2021)*. Springer, 2022, pp. 810–816.

[2] V. J. Lumelsky and A. A. Stepanov, "Path-planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape," *Algorithmica*, vol. 2, no. 1-4, pp. 403–430, 1987.

[3] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The International Journal of Robotics Research*, vol. 5, no. 1, pp. 90–98, 1986.

[4] B. Guo, N. Guo, and Z. Cen, "Motion saliency-based collision avoidance for mobile robots in dynamic environments," *IEEE Transactions on Industrial Electronics*, vol. 69, no. 12, pp. 13 203–13 212, 2021.

[5] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *The International Journal of Robotics Research*, vol. 17, no. 7, pp. 760–772, 1998.

[6] J. Van Den Berg, P. Abbeel, and K. Goldberg, "Lqg-mp: Optimized path planning for robots with motion uncertainty and imperfect state information," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 895–913, 2011.

[7] M. Kim and J. Oh, "Development of an optimal velocity selection method with velocity obstacle," *Journal of Mechanical Science and Technology*, vol. 29, pp. 3475–3487, 2015.

[8] J. Alonso-Mora, A. Breitenmoser, P. Beardsley, and R. Siegwart, "Reciprocal collision avoidance for multiple car-like robots," in *2012 IEEE International Conference on Robotics and Automation*. IEEE, 2012, pp. 360–366.

[9] H.-J. Jo, S.-R. Kim, J.-H. Kim, and J.-Y. Park, "Comparison of velocity obstacle and artificial potential field methods for collision avoidance in swarm operation of unmanned surface vehicles," *Journal of Marine Science and Engineering*, vol. 10, no. 12, p. 2036, 2022.

[10] M. A. K. Jaradat, M. H. Garibeh, and E. A. Feilat, "Autonomous mobile robot dynamic motion planning using hybrid fuzzy potential field," *Soft Computing*, vol. 16, pp. 153–164, 2012.

[11] M. Ali and S. Das, "Mobile robots with dynamic obstacle avoidance," in *2023 IEEE 3rd International Conference on Sustainable Energy and Future Electric Transportation (SEFET)*, 2023, pp. 1–6.

[12] G. Smith, "Newton's Philosophiae Naturalis Principia Mathematica," in *The Stanford Encyclopedia of Philosophy*, Winter 2008 ed., E. N. Zalta, Ed. Metaphysics Research Lab, Stanford University, 2008.

[13] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

[14] R. Regele and P. Levi, "Cooperative multi-robot path planning by heuristic priority adjustment," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2006, pp. 5954–5959.

[15] B. Scassellati, "Theory of mind for a humanoid robot," *Autonomous Robots*, vol. 12, pp. 13–24, 2002.

[16] M. Dorigo and G. Di Caro, "Ant colony optimization: a new metaheuristic," in *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*, vol. 2. IEEE, 1999, pp. 1470–1477.

[17] M. Ali and S. Das, "Swarms of mobile robots for area exploration," in *2023 Sixth International Conference of Women in Data Science at Prince Sultan University (WiDS PSU)*. IEEE, 2023, pp. 138–143.

[18] Y. Abe and M. Yoshiki, "Collision avoidance method for multiple autonomous mobile agents by implicit cooperation," in *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No. 01CH37180)*, vol. 3. IEEE, 2001, pp. 1207–1212.

[19] S. R. Lindemann and S. M. LaValle, "Current issues in sampling-based motion planning," in *Robotics Research. The Eleventh International Symposium: With 303 Figures*. Springer, 2005, pp. 36–54.

[20] S. J. Guy, J. Chhugani, C. Kim, N. Satish, M. Lin, D. Manocha, and P. Dubey, "Clearpath: highly parallel collision avoidance for multi-agent simulation," in *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2009, pp. 177–187.

[21] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23–33, 1997.