Comparison of Optimizing Path Planning for Mobile Robots with Obstacle Avoidance

Maram Ali, Saptarshi Das, Stuart Townley
Centre for Environmental Mathematics,
Faculty of Environment, Science and Economy,
University of Exeter, Penryn Campus,
Cornwall TR10 9FE, United Kingdom.
Email: {ma935, s.das3, s.b.townley}@exeter.ac.uk

***Abstract*—This research delves into a thorough examination of two distinct path-planning algorithms, denoted as algorithm A and algorithm B, operating in dynamic environments replete with moving obstacles. The primary objective of the study is to optimize the performance of algorithm A, thereby evolving it into the innovative algorithm B. Employing rigorous simulation examples, this study assesses the performance metrics of both algorithms, shedding light on their respective strengths and limitations. When confronted with dynamic settings featuring swiftly moving obstacles, algorithm B demonstrates a slight advantage owing to its optimized features. Despite this improvement, both algorithms face challenges in densely populated environments, leading to increased failure rates. Key metrics, including failure rate occurrences and navigation efficiency as a function of the distance covered towards the target, offer valuable insights into the nuanced behavior of these algorithms. The study's outcomes highlight the intricacies associated with the development of path-planning algorithms for real-world applications, particularly in dynamic and densely populated environments. The research accentuates the perpetual necessity for refinement and optimization, focusing specifically on adaptive algorithms capable of effectively managing high-velocity moving obstacles. These insights hold paramount importance in the advancement of intelligent robotic systems, empowering them to navigate intricate and dynamic environments with unparalleled precision and efficiency.**

***Index Terms*—Path Planning Algorithms, Dynamic Environments, Moving Obstacles, Obstacle Avoidance**

I. Introduction

Path planning entails the identification of the optimal trajectory within a specified area, connecting two points in a manner that maximizes efficiency. It aims to optimize the path between a source and destination by identifying the shortest and optimal route connecting them [1]. It is a well-studied topic due to its numerous applications in fields such as autonomous navigation, circuit board route and layout design, logistics, network routing, and video games, just to mention a few. As a result, researchers have developed and analyzed numerous algorithms. There are classical methods such as cell decomposition (CD) [2], roadmap approach (RA), artificial potential field (APF) [3], A* algorithm [1] and reactive methods such as reactive approaches such as genetic algorithm (GA) [4], fuzzy logic (FL), neural network (NN) [5], firefly algorithm (FA), particle swarm optimization (PSO) [6], ant colony optimization (ACO) [7], bacterial foraging optimization (BFO) and artificial bee colony (ABC) [8].

One of the biggest challenges faced by autonomous systems is the ability to navigate complex environments. Numerous techniques utilise the random walk methodology [9], it is not the optimal or most efficacious solution. To overcome this challenge, path-planning algorithms are used to enable robots to chart optimal routes while avoiding obstacles. With the increasing demand for automation in various industries, the need for sophisticated path-planning techniques has become more important than ever before. There are two distinct approaches to navigation and path planning: global and local. Global navigation is applicable in environments where the agent possesses prior knowledge of its environment and the obstacles' trajectory. It is typically used in environments where the path is precomputed before the agent commences its movement. In contrast, local navigation occurs in dynamic environments, necessitating real-time computation of path planning strategies based on the current state of the surroundings. The local navigation problem is inherently indeterminate, making it impossible to predict whether the agent will successfully reach its target position, a situation often denoted as an NP-hard problem. Consequently, this unpredictability can lead to mobile robots becoming trapped in local optima, resulting in infinite loops or the inability to navigate around obstacles. Global and local navigation are also known as off-line and on-line navigation [10].

The path planning algorithm for a familiar environment relies on classical methods like A*, CD, RA, and APF. While widely used due to their simplicity, these traditional algorithms have limited intelligence and often result in high computation costs and failure in densely populated and/or dynamic environments. Lately, reactive methods, including genetic algorithms, fuzzy logic, neural networks, and various others, have gained popularity in mobile robot navigation. These approaches are preferred over conventional methods due to their adeptness in handling environmental uncertainties [11]. In mobile robot obstacle avoidance, the grid-based approach is a fundamental technique. It discretizes the environment into non-overlapping cells, each corresponding to a specific location, indicating obstacle presence or free space [12], [13]. Its prevalence is owed to several key advantages. Firstly, this method offers a granular understanding of the environment, facilitating informed decision-making based on obstacle information. Additionally, its ease of implementation and efficient storage and retrieval

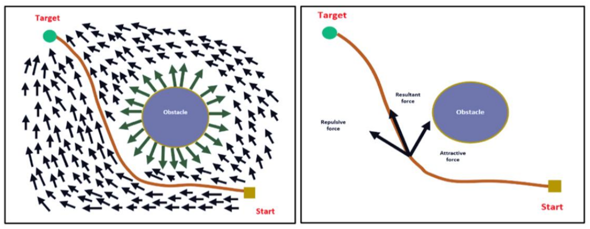

Fig. 1: Mobile robot navigation employing the artificial potential field (APF) technique [11].

of grid data in computer memory make it a practical choice for real-time navigation [14]. This grid-based representation is foundational, empowering robots with the intelligence and adaptability necessary for navigating complex environments safely and effectively. This study introduces a new method designed to enhance the efficiency of classical pathfinding algorithms in dynamic settings. The approach is tested on a 2D regular square grid environment and integrates some concepts from velocity obstacles (VO) and artificial potential fields (APF) to improve dynamic environment path-finding performance.

II. Related Works

A. Artificial Potential Fields

Artificial potential fields (APF) is a classic technique used in path planning and obstacle avoidance (see Fig. 1). It creates a virtual field where obstacles repel the robot and the goal attracts it. The robot moves by following the gradient of this field, navigating efficiently in real-time. Although APF algorithms can get trapped in local minima, researchers use advanced methods like escape functions to enhance their performance, making APF a valuable tool for navigating robots in complex and dynamic environments [15]. The APF method offers several advantages in mobile robot navigation. It is characterized by its simplicity and ability to plan the path in real time, especially in dynamic environments [11]. However, there are notable drawbacks to consider. APF algorithms can get trapped in local optima, leading to less-than-optimal paths, especially in intricate environments. Tuning the parameters correctly is crucial, and striking the right balance can be challenging [15].

B. Cell Decomposition

Cell decomposition is a fundamental technique in robotic path planning, essential for navigating complex environments. Dividing the grid into cells that are either free or occupied in the presence of obstacles By representing the environment discretely, cell decomposition simplifies the path-planning process. Additionally, the use of cell decomposition provides benefits in terms of improving computational speed. It minimizes the complexity of path planning algorithms by transforming the environment into a structured grid, enabling robots to navigate efficiently. Additionally, the technique supports global planning, allowing the robot to consider long-range paths while making decisions [16].

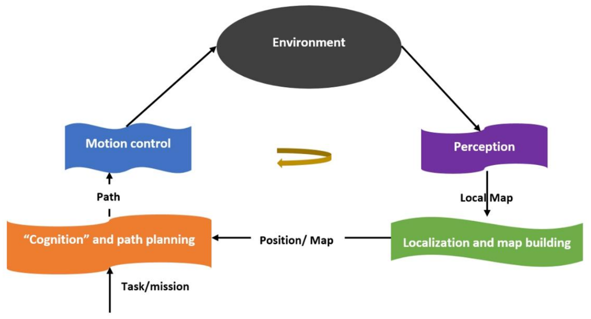

Fig. 2: Building blocks of a MR navigation system [10].

C. The A* Algorithm

The A* algorithm is used for pathfinding and is renowned for its efficiency and optimality. It operates on a graph, systematically exploring possible paths from the start to the goal node. A* employs a heuristic function that estimates the cost from the current node to the goal, guiding the search process. By considering both the actual cost from the start and the heuristic estimate, A* intelligently selects nodes to explore, effectively balancing between finding the shortest path and computational efficiency. One of its key strengths lies in its ability to guarantee the discovery of the shortest path, given an appropriate heuristic [1].

D. Velocity Obstacle Method

The Velocity Obstacle (VO) method is a crucial concept in motion planning. It operates by predicting potential future collisions based on the relative velocities and positions of moving objects within a robot's vicinity. By calculating velocity obstacles, which delineate the forbidden velocity space for the robot to avoid collisions, this method facilitates real-time decision-making. VO techniques allow robots to dynamically adjust their velocities to navigate complex and dynamic environments efficiently [17]. By offering a predictive framework, the VO method stands as a cornerstone in enhancing the autonomy and safety of robotic systems, finding valuable applications in various domains such as robotics, autonomous vehicles, and human-robot interaction scenarios. On the positive side, VO techniques offer proactive collision avoidance, enabling robots to anticipate potential collisions and adjust their paths accordingly. This predictive ability enhances safety and enables seamless navigation in dynamic environments [18]. Moreover, VO methods are highly adaptable and can be employed in decentralized systems, allowing individual robots to make collision avoidance decisions independently. Additionally, the reliance on predictable motion patterns of surrounding objects might limit the method's effectiveness in highly unpredictable scenarios. In considering VO for applications, striking a balance between its predictive capabilities and the challenges associated with its implementation is crucial for optimal utilization in real-world robotic systems [17].

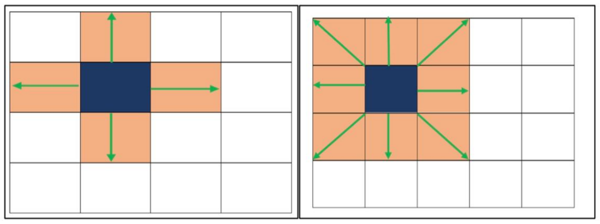

Fig. 3: 4-node (left) and 8-node (right) motion control approaches [11].

III. Materials and Methods

Mobile robot (MR) navigation encompasses a multifaceted process that entails environmental perception, localization, map construction, cognitive decision-making, path planning, and motion control. Sensory data interpretation falls under the purview of perception, while localization determines the precise robot position, and map construction assembles an accurate representation of the robot's surroundings. Cognitive decision-making is a pivotal stage where the optimal path is strategized, setting the stage for trajectory execution via motion control.All these components are essential to fulfill the navigation requirements, as illustrated in Fig. 2.

A. Implementing the Grid Environment

The environment is represented as a 2D array, a departure from the typical binary grid systems, prevalent in existing models [19]. A 100x100 grid is used for all the simulations discussed in this paper.

B. Motion Control

In regular, square grid environments, two prevalent motion control methods are employed: the 4-node and 8-node approaches (see Fig. 3). This study adopts the 8-node approach due to its provision of greater movement possibilities within the environment. Utilizing the grid's inherent coordinate system, motion control is achieved through a 2D translation vector, specifically the velocity vector, as illustrated below:

$$p'_x = p_x + v_x, \tag{1}$$

$$p'_y = p_y + v_y, \tag{2}$$

where, (p'_x, p'_y) are the new position in 2D, (p_x, p_y) are the current position and (v_x, v_y) the agent's velocity.

C. Path Planning Algorithm

The A* algorithm emerged as a key contender for this study, owing to its intricate equilibrium between complexity, performance, and computational efficiency. Consequently, this paper utilizes the A* algorithm as a foundational framework. This study utilizes a foundation to propose, put into action, examine, and assess different optimization techniques. These techniques are aimed at enhancing the algorithm's accuracy in identifying optimal paths within dynamic environments.

Central to the A* algorithm lies a crucial heuristic function responsible for approximating the cost of transitioning from the present state to the desired target. In the context of this research, a minimal-distance function was chosen for the A* algorithm's implementation. The improved A* algorithm accounts for dynamic obstacles [1], [20], [21] and assigns the lowest cost to cells in the opposite direction. This enhanced version will adapt to changing environments while maintaining efficient path planning. Provided below are the outlined steps for the A* algorithm [22], [23]:

```
Initialization:
Initialize an open set containing the start node;
Initialize an empty closed set;
Set the cost of the start node: $g(\text{start}) = 0$;
Set the heuristic estimate from start to goal:
  $h(\text{start}) = \text{heuristic(start, goal)}$;
Calculate the total cost: $f(\text{start}) = g(\text{start}) + h(\text{start})$;
while the open set is not empty do
    Select the node with the lowest $f$ value (current
      node);
    Move the current node from the open set to the
      closed set;
    if current node is the goal node then
        Terminate the search and backtrack the path;
    end
    Expand the current node: for each neighbor of the
      current node do
        Calculate the cost to move from the current
          node to the neighbor: $g(\text{neighbor}) =$
          $g(\text{current}) + \text{cost(current, neighbor)}$;
        if the neighbor is not in the closed set or the
          new cost is lower then
            Update the neighbor's cost: $g(\text{neighbor})$;
            Calculate the heuristic estimate from
              neighbor to goal:
              $h(\text{neighbor}) = \text{heuristic(neighbor, goal)}$;
            Calculate the total cost:
              $f(\text{neighbor}) = g(\text{neighbor}) + h(\text{neighbor})$;
            Add the neighbor to the open set;
        end
    end
end
Backtrack path:once the goal node is reached,
  reconstruct the path from goal to start ;
```

Algorithm 1: A* Path Planning Algorithm

The heuristic function $h(n)$ is often based on the Euclidean distance between two nodes $(I\,(x_i, y_i))$ and $(J\,(x_i, y_j))$, and the Euclidean distance (D) calculated as:

$$D = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}. \tag{3}$$

This distance acts as an estimate of the cost from the current node to the target node. The A* algorithm combines this heuristic estimate with the actual cost $g(n)$ to guide the search efficiently. Below show the steps for improved A* algorithm:

```
Initialization:
Initialize an open set containing the start node;
Initialize an empty closed set ;
Set the cost of the start node: $g(start) = 0$;
Set the heuristic estimate from start to goal:
  $h(start) = heuristic(start, goal)$;
Calculate the total cost:
  $f(start) = g(start) + h(start)$;
while the open set is not empty do
    Select the node with the lowest $f$ value as the current node;
    Move the current node from the open set to the closed set;
    if the current node is the goal node then
        Terminate the search;
    end
    Expand Current Node: for each neighbor of the current node do
        Calculate the cost to move from the current node to the neighbor: $g(neighbor) = g(current) + cost(current, neighbor)$;
        if the neighbor is already in the closed set and the new cost is higher then
            Skip it;
        end
        if the neighbor is not in the open set or the new cost is lower then
            Update the neighbor's cost: $g(neighbor)$;
            Calculate the heuristic estimate from neighbor to goal: $h(neighbor) = heuristic(neighbor, goal)$;
            Calculate the total cost: $f(neighbor) = g(neighbor) + h(neighbor)$;
            Add the neighbor to the open set;
        end
    end
    Dynamic Obstacle Handling: if a dynamic obstacle is detected then
        Identify the cells affected by the obstacle;
        Assign a temporary high cost to these cells;
        Update the costs of neighboring cells accordingly;
        Reevaluate the path by repeating the A* process;
    end
end
Opposite Direction Cost : if recalculating the path due to dynamic obstacles: then
    For cells in the opposite direction (opposite to the original path);
    a.Assign a lower cost($f(opposite\_cell) = g(opposite\_cell) + h(opposite\_cell)$)to encourage exploration away from the obstacle;
    b.Update the costs of neighboring cells as needed;
end
Backtrack path: once the goal is reached, reconstructs the path from the goal node to the start node;
```

Algorithm 2: Improved A* Algorithm

IV. Optimization Techniques used in Algorithm B

1) Improved A algorithm:* The proposed variation of the A* algorithm, denoted as algorithm B, introduces a novel application of artificial potential fields (APF) to enhance path planning around dynamic obstacles. This optimization technique aims to improve the algorithm's efficiency in navigating dynamic environments by repelling the mobile robot (MR) from dynamic obstacles.

2) Collision Course Determination: To further enhance efficiency, the APF technique is selectively activated only when the MR and an obstacle are on a collision course. This determination is made by calculating the dot product between the MR's position vector with respect to the obstacle and the obstacle's velocity vector. If the dot product yields a positive value, indicating a converging trajectory, the APF mechanism is activated to influence the path planning process.

3) Simplicity and Compatibility: By introducing only repulsive forces when necessary, the algorithm remains computationally efficient, balancing the need for responsiveness in dynamic environments with the overall simplicity of the path-planning framework. The idea of a local map is introduced and incorporated into both the algorithms. This is a computational optimization method that reduces the search space of the algorithm, thus increasing the computation and memory performance of the algorithm. The local map is characterized by the MR's maximum allowable velocity and a factor of safety constant. Using the MR's maximum velocity, the algorithm searches only the cells, reachable from its current position. The factor of safety represents an area around a dynamic obstacle(s) where detection is possible, thus allowing the MR to safely react to the obstacle.

V. Mathematical Modelling of the A* Algorithm

The implemented A* algorithm utilizes one essential mathematical component, $h(n)$. This is the heuristic function that estimates the cost of the cheapest path from node n to the goal node. This function provides an approximation of the remaining cost from n to the goal, serving as a guiding factor for the search algorithm. As previously mentioned, the heuristic function $h(n)$ is calculated by computing the Euclidean distance between a designated cell or location and the target. Consequently, the resultant function for each iteration of the algorithm is computed by computing the minimum value of $h(n)$ in a set of all possible nodes of length m, shown as:

$$f(n) = \min\left(\{h(n)_1, \quad h(n)_2, \quad \ldots, h(n)_m\}\right). \tag{4}$$

Repeating this function for each iteration configuration state, until the goal is achieved constitutes the algorithm's approach to optimal path planning.

VI. Mobile Robot's Artificial Intelligence and its Impact on Navigation Performance

The success of any path-planning algorithm in dynamic environments heavily relies on the intelligence embedded within the mobile robot (MR). In the context of the present study, the term *intelligence* refers to the capabilities of the MR to perceive its surroundings, make informed decisions, and adapt its path planning strategy in real-time. The foundation of the mobile robot's artificial intelligence lies in its perception and sensing capabilities. The MR gathers real-time information about its environment. This sensory input allows the robot to identify obstacles, discern their trajectories, and continuously update its understanding of its dynamic surroundings. The heart of the MR's artificial intelligence will certainly lie in its decision-making algorithms. These algorithms, often driven by machine learning or rule-based systems, enable the robot to interpret the sensory data and make intelligent decisions regarding its path. The ability to assess risk, predict obstacle movements, and dynamically adjust the path planning strategy contributes significantly to the MR's adaptability in dynamic environments. In dynamic scenarios, where obstacles move unpredictably and environmental conditions change rapidly, the MR's artificial intelligence will definitely play a pivotal role. The MR's ability to adapt its path planning in real-time based on the evolving situation is a key determinant of its success. This adaptability ensures that the robot can navigate through complex and dynamic environments efficiently and safely. The integration of artificial intelligence into the mobile robot's decision-making processes directly influences the outcomes of the present study. The proposed path planning algorithms, including the optimized A* algorithm, are designed to synergize with the MR's intelligence, leveraging real-time data to navigate through dynamic environments effectively.

VII. Results and Discussions

To comprehensively assess the performance and behavior of the proposed path-planning algorithms, a series of rigorous tests were conducted. Twenty distinct simulations were run across four diverse environments, each serving as a unique case study. These environments were meticulously chosen to observe and analyze the algorithms' effectiveness and adaptability. They include:

A. *Sparsely Populated Dynamic Environment*

In this specific case study, a total of twenty simulations were conducted, positioning 5 dynamic obstacles within the grid environment. Importantly, these obstacles were designed to move at a speed comparable to that of the MR i.e. both the MR and obstacles were set to have a maximum speed of 1 unit per iteration. The outcomes of this study are presented in Table I, detailing the results derived from these simulations. When navigating dynamic obstacles, there are notable differences in performance between the two approaches. Most notable is the crash rate; the A* algorithm failed 10 times out of the 20 simulation runs. Based on this data, this means it has a 50% chance of failing in a dynamic environment populated by 5 obstacles. $\%$ probability of failure $= \frac{10}{20} \times 100 = 50\%$. On the other hand, the optimized A* algorithm failed once out of the 20 simulation runs. This translates to a 5% chance of failure. So, the $\%$ probability of failure $= \frac{1}{20} \times 100 = 5\%$. To deepen the comprehension of the performance of A* algorithm

TABLE I: Sparsely populated dynamic obstacles (obstacle max velocity = 1, MR max velocity = 1, number of obstacles = 5).

			Performance			
Simulation_No	A_Iterations	A_Distance	A_Crashed	B_Iterations	B_Distance	B_Crashed
1	75	98.0244	1	126	165.8356	0
2	99	137.1787	0	99	137.1787	0
3	69	92.4386	1	222	296.2153	0
4	102	139.3503	0	139	184.2203	0
5	106	141.6934	0	106	141.6934	0
6	32	43.4264	1	199	247.9483	0
7	139	160.6102	0	115	154.0071	0
8	14	17.9706	1	143	192.3625	0
9	24	32.5269	1	179	218.4214	0
10	55	75.5391	1	126	169.1493	0
11	99	137.1787	0	99	137.1787	0
12	102	139.3503	0	102	139.3503	0
13	101	138.7645	0	105	142.7645	0
14	67	91.267	1	176	217.4924	0
15	60	82.6102	1	82	112.4802	1
16	99	137.1787	0	101	139.5929	0
17	186	188.5563	0	145	197.6762	0
18	62	85.8528	1	145	199.333	0
19	99	137.1787	0	105	143.5929	0
20	42	57.1543	1	113	156.1493	0
MEAN/ AVERAGE	81.6	106.6925	0.5	131.35	174.6321	0.05

variants, a thorough and inclusive comparison was undertaken, drawing valuable insights from the existing literature on state-of-the-art path-planning algorithms. The consideration of notable algorithms extends beyond A* to encompass D* [24], RRT* (Rapidly Exploring Random Tree) [25], and MPC (Model Predictive Control) [26]. The widespread recognition of the A* algorithm for its simplicity and efficiency in static environments set a foundational drawback for this comparison. However, its limitations become apparent in dynamic scenarios, exemplified by the 50% probability of failure observed in algorithm A within our study. This observation underscores the necessity for innovation to address the challenges posed by dynamic environments. The introduction of the optimized A* algorithm (algorithm B) marks a significant departure from traditional approaches. Drawing inspiration from concepts presented by [1], the algorithm B exhibits a substantial improvement, dramatically reducing the probability of failure to 5%. This optimization signifies a noteworthy stride in enhancing the adaptability and robustness of A* in the face of dynamic obstacles. In the broader context, comparative studies, such as the evaluation of D* [24], [27] and the analysis of RRT* [25] contribute valuable perspectives on algorithm adaptability in dynamic environments. These studies emphasize the significance of continuous advancements in pathfinding algorithms to address the complexities of dynamic scenarios effectively. Among the 20 simulations conducted, algorithm B showcased the longest path in navigating the dynamic environment. As depicted in Fig. 4 on the right, this scenario highlights one of the limitations of the optimized approach.

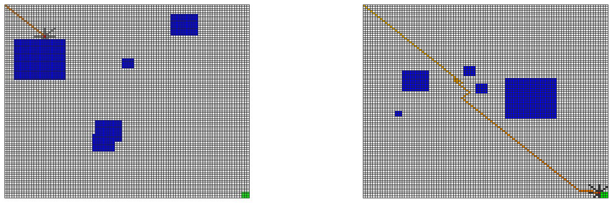

Fig. 4: In the visual representation of the test simulation in algorithms A and B for sparsely populated dynamic obstacles. The figure on the left demonstrates that MR did not successfully reach the target. However, in the figure on the right, MR successfully reached the target.

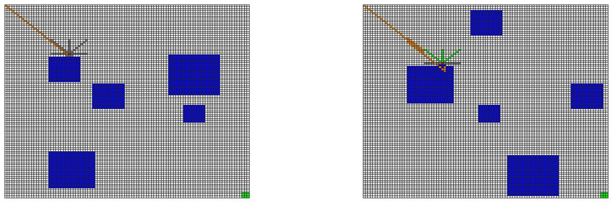

Fig. 5: In the visual representation of the test simulation, algorithms A and B perform faster than MR in navigating sparsely populated dynamic obstacles. As evident from both the figures on the right and left, MR is unable to reach the target successfully in this particular test scenario.

B. *Sparsely Populated Dynamic Obstacles Faster than MR*

This test mirrors the one outlined in subsection VII.A. However, in this study, the velocity of the obstacles was doubled to 2 units per iteration to examine the impact of the velocity disparity between the MR and the obstacles on the performance of the path planning algorithms. The outcomes from 20 simulations conducted in this environment are depicted in Table II.

In Fig. 5, depicting simulation, both algorithms failed to establish a collision-free path to the target, despite the repelling mechanism activated by the optimized algorithm B (highlighted by the green shading in the MR's local scope). This occurrence underscores a fundamental limitation of reactive algorithms when contrasted with predictive methodologies such as velocity obstacles. Notably, due to obstacles moving at velocities twice that of the mobile robot (MR), the MR struggles to respond promptly to obstacles on an impending collision course. This observation gains further significance when analyzing the outcomes from all twenty simulations conducted under these specific conditions. Algorithm A exhibited a 65% failure rate, while algorithm B demonstrated a 60% failure rate. This implies that the optimization integrated into the original algorithm had a minor impact on performance, approximately 5%, within this constrained context. Moreover,in terms of iterations, algorithm B demonstrates a higher average (mean B = 99.55), as compared to algorithm A (mean A = 61.65). This suggests that algorithm B requires a more extensive computational process, involving a greater number of

TABLE II: Faster than MR sparsely populated dynamic obstacles (obstacle max velocity = 2, MR max velocity = 1, number of obstacles = 5).

Performance						
Simulaion_No	A_Iterations	A_Distance	A_Crashed	B_Iterations	B_Distance	B_Crashed
1	23	31.1127	1	127	175.1198	0
2	60	83.0244	1	63	87.6812	1
3	110	144.0366	0	68	92.6812	1
4	99	137.1787	0	99	137.1787	0
5	99	137.1787	0	99	137.1787	0
6	13	16.1421	1	15	19.799	1
7	72	99.5807	1	75	104.6518	1
8	100	137.7645	0	107	146.8356	0
9	68	93.9239	1	86	113.9949	1
10	47	65.0538	1	133	176.1493	0
11	16	20.799	1	190	260.2447	1
12	19	24.2132	1	145	195.6051	0
13	101	138.7645	0	129	167.5929	0
14	8	9.0711	1	120	164.5635	1
15	36	47.8406	1	85	110.5097	1
16	21	27.0416	1	60	81.7817	1
17	28	37.3553	1	104	143.5929	1
18	185	187.5563	0	85	112.1665	1
19	20	26.4558	1	111	149.1787	0
20	108	143.6934	0	90	120.066	1
MEAN/ AVERAGE	61.65	80.3894	0.65	99.55	134.8286	0.6

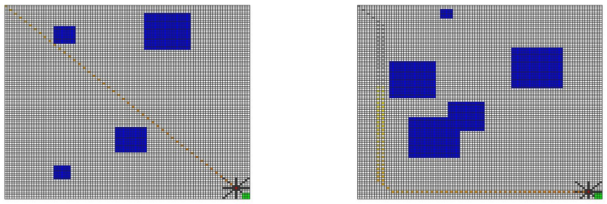

Fig. 6: In the visual representation of the test simulation in algorithms A and B in slower than MR sparsely populated dynamic obstacles. In both cases, MR successfully reaches the target. However, in the figure on the right, the path taken by MR is longer than the one depicted in the figure on the left.

iterations to navigate the environment. Similarly, the analysis of the distance metric reveals a substantial disparity between the two algorithms. Algorithm B covers a significantly greater distance (mean B = 134.8286), as compared to the algorithm A (mean A = 80.3894). This implies that algorithm B adopts a more extensive path in its planning process.

C. *Sparsely Populated Dynamic Obstacles Slower than MR*

This test replicates the conditions outlined in (A) above. However, in this study, the mobile robot's (MR) velocity was doubled to 2 units per iteration to assess the influence of the velocity disparity between the MR and the obstacles on the performance of the path planning algorithms (see Fig. 6). The results from 20 simulations conducted in this environment are presented in Table III.

TABLE III: Slower than MR sparsely populated dynamic obstacles (obstacle max velocity = 1, MR max velocity = 2, number of obstacles = 5).

Performance						
Simulaion_No	A_Iterations	A_Distance	A_Crashed	B_Iterations	B_Distance	B_Crashed
1	50	135.7645	0	50	135.7645	0
2	33	88.0244	1	65	170.5635	0
3	50	135.7645	0	52	139.1787	0
4	54	141.2792	0	54	141.2792	0
5	52	138.9361	0	144	363.7838	0
6	52	138.9361	0	52	138.9361	0
7	50	135.7645	0	50	135.7645	0
8	51	136.9361	0	123	304.3747	0
9	50	135.7645	0	50	135.7645	0
10	52	138.9361	0	104	247.9066	0
11	22	58.5685	1	122	325.328	0
12	52	138.9361	0	74	194.7767	0
13	14	35.1127	1	138	292.6762	0
14	40	108.6518	1	87	238.7595	0
15	50	135.7645	0	52	140.5929	0
16	21	55.7401	1	93	244.0315	0
17	14	35.9411	1	69	182.5341	0
18	50	135.7645	0	50	135.7645	0
19	50	135.7645	0	50	135.7645	0
20	20	40.4853	1	174	400.0315	0
MEAN/ AVERAGE	41.35	110.3418	0.35	82.65	210.1788	0

TABLE IV: Densely populated with dynamic obstacles (obstacle max velocity = 1, MR max velocity = 1, number of obstacles = 10).

Performance						
Simulaion_No	A_Iterations	A_Distance	A_Crashed	B_Iterations	B_Distance	B_Crashed
1	106	141.6934	0	127	168.9066	0
2	66	91.0955	1	51	70.2965	1
3	81	94.9117	1	113	151.1787	0
4	28	36.5269	1	172	225.6762	1
5	35	42.2843	1	54	67.4975	1
6	22	28.8701	1	158	216.061	0
7	10	12.3137	1	230	295.931	0
8	87	120.7939	1	120	163.9777	0
9	70	96.7523	1	103	140.5219	1
10	37	50.0833	1	90	124.6224	1
11	37	50.0833	1	104	141.1076	1
12	18	24.0416	1	136	179.9777	0
13	28	37.3553	1	401	534.2052	0
14	56	77.3675	1	86	118.5513	1
15	100	138.1787	0	140	186.0488	0
16	54	74.9533	1	180	241.5462	1
17	30	40.598	1	229	300.0732	1
18	25	33.5269	1	86	117.7229	1
19	12	14.7279	1	198	266.1737	1
20	54	74.9533	1	188	253.1026	0
MEAN/ AVERAGE	47.8	64.0555	0.9	148.3	198.1589	0.55

When the MR is set to travel faster than obstacles, there is an increase in both algorithms' performance. This is with respect on the data based on the test (A) results above. Algorithm A has a failure rate of 35% from 50% while algorithm B has a 0% failure rate from 5%. This results to a 15% and 5% decrease in failure rate of the approaches. However, despite a 15% in the failure rate between the two algorithms, there is an approximately 50% difference in the iteration count and distance metrics using the optimized algorithm B, as shown below:

$$\frac{\text{Algorithm A average iterations}}{\text{Algorithm B average iterations}} \times 100$$
$$\frac{41.35}{82.65} \times 100 = 50\%$$
$$\frac{\text{Algorithm A average distance}}{\text{Algorithm B average distance}} \times 100$$
$$\frac{110.3418}{210.1788} \times 100 = 52.5\%.$$

This indicates that the optimized algorithm B is not tuned for shorter paths and instead favors success rate over path length.

D. Densely Populated with Dynamic Obstacles

In this scenario, the environment was populated with 10 dynamic obstacles, each possessing relatively similar speeds. The objective of this test was to assess the influence of the obstacle population on the algorithm's performance (see Fig. 7). The outcomes of the simulations conducted for this specific case study are presented in Table IV.

The outcomes of this test align with expectations i.e. both algorithms exhibited reduced performance as compared to the reference data set from test (A) above. This decline in performance can be attributed to the higher probability of collisions with obstacles in densely populated environments as compared to the sparsely populated ones. This inference is derived from the failure rate metric. Algorithm A's failure rate escalated from 50% to 90%, whereas algorithm B's failure rate increased from 5% to 55%. To comprehensively address the scalability of the proposed algorithms, the findings presented in a densely populated setting shed light on the intricate challenges faced, revealing a substantial escalation in failure rates for both algorithm A and the optimised algorithm B. In the context of larger and more intricate environments, the scalability of path-planning algorithms emerges as a critical determinant of their practical applicability. The observed performance decline in densely populated scenarios hints at potential limitations in

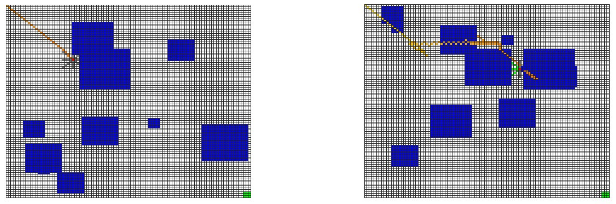
Fig. 7: In the visual representation of the test simulation for algorithms A and B in a densely populated environment with dynamic obstacles, in both figures, MR faces challenges in reaching the target. In the figure on the right, the magnetic resonance (MR) feature is activated, which pushes away obstacles. However, despite this feature, MR was ultimately unable to successfully reach the target within this particular environment.

the scalability, indicating that as the environmental complexity increases, encompassing larger spatial dimensions and higher obstacle densities, the algorithms may encounter more challenges in sustaining efficiency and avoiding collisions. In this study, increasing the grid size to 200 has some cost associated with it. One is that the computational efficiency is reduced, requiring a considerable amount of resources to complete the simulation. On the upside, there are few collisions with randomly moving obstacles.

VIII. Conclusions and Future Works

In conclusion, the comparative analysis of algorithms A and B, as evaluated through rigorous simulations, provides a deeper understanding of their performance in dynamic environments. When subjected to dynamic environments with moving obstacles, algorithm B displayed a marginal advantage over algorithm A, attributed to its optimized features. Despite this improvement, both algorithms faced challenges in densely populated environments, leading to an escalation in failure rates. Examining the metrics, it is evident that the introduction of moving obstacles with double the velocity of the mobile robot significantly impacted the algorithms' performance. While algorithm B showed minor enhancements, it still grappled with swift-moving obstacles. Notably, algorithm B's repelling mechanism occasionally resulted in MR entrapment between obstacles and boundaries. While both algorithms exhibited commendable performance under specific conditions, they were not entirely immune to challenges. The identified metrics emphasize the pressing need for further research, particularly in the realm of dynamic obstacle avoidance strategies. Enhancing the algorithms' adaptability and reliability is crucial for their practical implementation in real-world scenarios. Incorporating random movements within the obstacles may better emulate real-world uncertainties. Additionally, it is advisable to integrate acceleration and inertia constraints into the model [28] in future research. This enhancement would contribute to a more precise representation of motion dynamics in the physical world, thereby augmenting the accuracy and realism of the simulation.

Acknowledgment

MA thanks King Khalid University and Saudi Arabia Cultural Bureau in the UK, London for supporting this research.

References

[1] A. K. Guruji, H. Agarwal, and D. Parsediya, "Time-efficient a* algorithm for robot path planning," *Procedia Technology*, vol. 23, pp. 144–149, 2016.

[2] L. Liu, X. Wang, X. Yang, H. Liu, J. Li, and P. Wang, "Path planning techniques for mobile robots: Review and prospect," *Expert Systems with Applications*, p. 120254, 2023.

[3] R. B. Tilove, "Local obstacle avoidance for mobile robots based on the method of artificial potentials," in *Proceedings., IEEE International Conference on Robotics and Automation*. IEEE, 1990, pp. 566–571.

[4] A. Ismail, A. Sheta, and M. Al-Weshah, "A mobile robot path planning using genetic algorithm in static environment," *Journal of Computer Science*, vol. 4, no. 4, pp. 341–344, 2008.

[5] R. S. Pol and M. Murugan, "A review on indoor human aware autonomous mobile robot navigation through a dynamic environment survey of different path planning algorithm and methods," in *2015 International Conference on Industrial Instrumentation and Control (ICIC)*. IEEE, 2015, pp. 1339–1344.

[6] P. Raja and S. Pugazhenthi, "Path planning for mobile robots in dynamic environments using particle swarm optimization," in *2009 International Conference on Advances in Recent Technologies in Communication and Computing*. IEEE, 2009, pp. 401–405.

[7] C. Blum, "Ant colony optimization: Introduction and recent trends," *Physics of Life reviews*, vol. 2, no. 4, pp. 353–373, 2005.

[8] D. Karaboga and B. Akay, "A comparative study of artificial bee colony algorithm," *Applied Mathematics and Computation*, vol. 214, no. 1, pp. 108–132, 2009.

[9] 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.

[10] P. Raja and S. Pugazhenthi, "Optimal path planning of mobile robots: A review," *International Journal of Physical Sciences*, vol. 7, no. 9, pp. 1314–1320, 2012.

[11] B. Patle, A. Pandey, D. Parhi, A. Jagadeesh *et al.*, "A review: On path planning strategies for navigation of mobile robot," *Defence Technology*, vol. 15, no. 4, pp. 582–606, 2019.

[12] S. M. LaValle, *Planning algorithms*. Cambridge University Press, 2006.

[13] 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.

[14] P. Yap, "Grid-based path-finding," in *Advances in Artificial Intelligence: 15th Conference of the Canadian Society for Computational Studies of Intelligence, AI 2002 Calgary, Canada, May 27–29, 2002 Proceedings 15*. Springer, 2002, pp. 44–55.

[15] X. Fan, Y. Guo, H. Liu, B. Wei, and W. Lyu, "Improved artificial potential field method applied for auv path planning," *Mathematical Problems in Engineering*, vol. 2020, pp. 1–21, 2020.

[16] R. Gonzalez, M. Kloetzer, and C. Mahulea, "Comparative study of trajectories resulted from cell decomposition path planning approaches," in *2017 21st International Conference on System Theory, Control and Computing (ICSTCC)*. IEEE, 2017, pp. 49–54.

[17] 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.

[18] D. Wilkie, J. Van Den Berg, and D. Manocha, "Generalized velocity obstacles," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2009, pp. 5573–5578.

[19] Z. A. Algfoor, M. S. Sunar, and H. Kolivand, "A comprehensive study on pathfinding techniques for robotics and video games," *International Journal of Computer Games Technology*, vol. 2015, pp. 7–7, 2015.

[20] X. Huang, X. Dong, J. Ma, K. Liu, S. Ahmed, J. Lin, and B. Qiu, "The improved a* obstacle avoidance algorithm for the plant protection uav with millimeter wave radar and monocular camera data fusion," *Remote Sensing*, vol. 13, no. 17, p. 3364, 2021.

[21] F. Duchoň, A. Babinec, M. Kajan, P. Beňo, M. Florek, T. Fico, and L. Jurišica, "Path planning with modified a star algorithm for a mobile robot," *Procedia Engineering*, vol. 96, pp. 59–69, 2014.

[22] X. Wang, Z. Liu, and J. Liu, "Mobile robot path planning based on an improved a* algorithm," in *International Conference on Computer Graphics, Artificial Intelligence, and Data Processing (ICCAID 2022)*, vol. 12604. SPIE, 2023, pp. 1093–1098.

[23] H. Wang, S. Lou, J. Jing, Y. Wang, W. Liu, and T. Liu, "The ebs-a* algorithm: An improved a* algorithm for path planning," *PloS One*, vol. 17, no. 2, p. e0263841, 2022.

[24] C. Saranya, M. Unnikrishnan, S. A. Ali, D. Sheela, and V. Lalithambika, "Terrain based d algorithm for path planning," *IFAC-PapersOnLine*, vol. 49, no. 1, pp. 178–182, 2016.

[25] T. Zeng and B. Si, "Mobile robot exploration based on rapidly-exploring random trees and dynamic window approach," in *2019 5th International Conference on Control, Automation and Robotics (ICCAR)*. IEEE, 2019, pp. 51–57.

[26] H. Bai, J. Gao, X. Sun, and W. Yan, "Model predictive visual trajectory-tracking control of wheeled mobile robots," in *2019 IEEE 28th International Symposium on Industrial Electronics (ISIE)*. IEEE, 2019, pp. 569–574.

[27] F. A. Raheem, U. I. Hameed *et al.*, "Path planning algorithm using d* heuristic method based on pso in dynamic environment," *American Scientific Research Journal for Engineering, Technology, and Sciences*, vol. 49, no. 1, pp. 257–271, 2018.

[28] M. Ali and S. Das, "Motion of mobile robots in environments with dynamic obstacles and arbitrary directions," in *2023 7th International Conference on Electronics, Materials Engineering Nano-Technology (IEMENTech)*, 2023, pp. 1–6.