Hybrid Quantum-Classical Algorithms for Path Planning: QAOA-A* and CQRW-A*

Asif Aktab Ronggon[1], Tuhin Hossain[2], Md. Jakir Hossen[3], M. F. Mridha[4]
[1]Dept. of Electrical and Electronic Engineering, Bangladesh University of Engineering and Technology, Dhaka, Bangladesh
[2]Dept. of Computer Science and Engineering, Jahangirnagar University, Savar, Dhaka, Bangladesh
[3]Center for Advanced Analytics (CAA), COE for Artificial Intelligence Faculty of Engineering and Technology (FET), Multimedia University, 75450 Melaka, Malaysia.
[4]Dept. of Computer Science, American International University-Bangladesh, Dhaka, Bangladesh
Corresponding author: (email:jakir.hossen@mmu.edu.my and firoz.mridha@aiub.edu)

ABSTRACT– Efficient path planning for unmanned vehicles in complex, semi-structured environments remains a formidable challenge, primarily because of vast planning spaces and intricate environmental constraints. Classical algorithms, such as A*, often struggle with large search spaces and local optima, which limits their effectiveness in dynamic scenarios such as evacuation planning. One promising approach to overcome these challenges is to integrate quantum techniques into a traditional system to boost the performance of pathfiniding algorithms. In this study, we used quantum methods to study the robot navigation. The simulation experiment used 20x20 and 30x30 grids to create a two-dimensional plane space with obstacle densities of 17%, 30%, and 35%. Next, we generated a heuristic function that combines quantum-inspired computing with the classical A* algorithm. This approach suite contains quantum-inspired models, such as the Quantum Approximate Optimization Algorithm (QAOA) and Continuous Quantum Random Walk (CQRW), which outperform the traditional search methods. The experimental results showed that the shortest path lengths were consistently achieved by CQRW-enhanced A* on a 20×20 grid with a 17% obstacle density, a 30×30 grid with a 35% obstacle density, and path lengths of 26.09, 40.70, and 39.30 units, respectively. However, the CQRW-A* took 65.74s, 197.60s, and 515.52s to execute, respectively. In contrast, the 20×20 grid showed that QAOA-A* generated the shortest path with obstacle densities of 30% and 35%, yielding 21.71 and 27.45 units, respectively, but at a high computational cost of 90.09s and 96.89s. However, Classical-A* showed substantially faster execution times, ranging from 0.07 to 1.91s, although it provided larger pathways in most instances (32.57 units, 27.73 units, and 34.30 units). Thus, quantum-enhanced approaches, such as CQRW-A* and QAOA-A*, demonstrate superior performance in path optimization.

***Index Terms*—Quantum Approximate Optimization Algorithm (QAOA), Continuous Quantum Random Walk (CQRW), Classical A*, Quantum-Inspired Heuristics, Hybrid Quantum-Classical Algorithms, Global Path Planning.**

I. INTRODUCTION

Path planning has recently emerged as a prominent issue, particularly in the context of robot navigation and evacuation [1]. This remains a critical challenge in robotics. Fundamental to motion planning is based on two concepts: motion refers to movement from one place to another, and planning refers to the formulation of a strategy to accomplish the intended motion. In motion planning, a robot follows a predefined path through a known environment to achieve a specific goal [2], [3]. The coverage path planning (CPP) issue is determining the best, most efficient, and safest way to traverse a given region.

Despite advancements in classical path-planning techniques, traditional algorithms such as A* face significant limitations in dynamic, large-scale environments, where the search space is vast and computational costs increase exponentially. These algorithms often become trapped in local optima, rendering them suboptimal in real-world scenarios in which the environment frequently changes. By contrast, quantum-inspired algorithms exploit the principles of superposition and entanglement, allowing them to explore multiple solution paths concurrently. This potentially provides a significant advantage in overcoming these local minima and computational bottlenecks, thereby allowing for more efficient path optimization in complex environments.

Although quantum computing holds great promise, it is still in its early stages and faces scalability challenges. Therefore, hybrid quantum-classical approaches have emerged as a practical solution, where the robustness of classical algorithms provides a quantum advantage for specific tasks. In the context of path planning, the hybridization of quantum heuristics with classical search strategies allows for a scalable, optimized, and robust solution, capable of navigating complex, obstacle-dense environments while benefiting from quantum speedup in global optimization.

The optimization of robot path planning is distinct from that of the conventional path planning optimization. Intelligent optimization algorithms, particularly heuristic intelligent optimization algorithms [4], are crucial path planning algorithms for robots navigating dynamic situations. Particularly prominent algorithms for path planning include the bee colony algorithm [5], ant colony algorithm [6], genetic algorithm [7], A* algorithm [8], particle swarm optimization algorithm [9], and artificial potential field algorithm [10]. Using optimization features such as minimum energy consumption, shortest time, and shortest distance, the main goal of path planning is to map the safest and most efficient route from the origin to the destination [11], [12], [13]. Recent technological advances have led to the application of path-planning approaches based on quantum optimization algorithms and deep learning [14], [15], [16]. To accelerate computations within a classical algorithm, a new area of research called hybrid (quantum-classical) algorithms combines a central processing unit (CPU) with a quantum processing unit (QPU) [17].

Quantum optimization methods utilize quantum bits (qubits) and quantum gates [18] to enhance solution sets by simultaneously exploring multiple possibilities through quantum superposition and interference. These quantum properties allow algorithms to navigate complex solution spaces more efficiently, providing faster computation times and more reliable global optimality detection than traditional evolutionary algorithms. Quantum Genetic Algorithms (QGA) and Quantum Approximate Optimization Algorithms (QAOA) have been widely applied in fields such as production scheduling, data mining, and combinatorial optimization, demonstrating their broad utility and theoretical significance.

In this study, we demonstrate the implementation of an approach that combines classical robotics with quantum computing (QC). We investigated the motion challenges of robots in a quantum decision-making setting. The production rules inform the quantum rule-processing algorithm's processing of the robot's activity planning in a recognize-act cycle. Systems equipped with artificial intelligence (AI) can benefit significantly from the ability of this method to boost their computational performance. The interconnected nature of quantum theory and human reasoning, particularly its multitasking capabilities, motivated the deployment of quantum algorithms in this system. The main contributions of this study are as follows:

i. We present a novel hybrid framework that integrates QAOA and CQRW-based heuristics into a classical A* search, enabling efficient navigation through highly complex, obstacle-rich environments characterized by dense polygonal obstacles and large configuration spaces.
ii. We rigorously designed and mathematically formulated a custom cost Hamiltonian for the QAOA that incorporates obstacle proximity and path cost, along with a weighted Laplacian-based continuous quantum walk model that captures global spatial connectivity, allowing the heuristic to reflect both local and nonlocal environmental structures.
iii. We theoretically prove that these quantum-inspired heuristics maintain admissibility and consistency, ensuring the preservation of A*'s optimality and completeness guarantees, even in high-dimensional, cluttered search spaces, thereby facilitating more informed and globally aware search strategies in challenging robotic navigation scenarios.
iv. Through extensive experiments on 20x20 and 30x30 grids with varying obstacle densities, we demonstrate that the proposed quantum-enhanced algorithms, QAOA-A* and CQRW-A*, consistently outperform classical A* in path optimization, highlighting their potential for real-world applications despite increased computational time.

A review of the intended study is provided in section 2, and the methodology is described in section 3. Section 4 specifies the environment setting, conducts a thorough review of the experimental results, and provides the performance evaluation metrics for the specific experimental configuration. Finally, Section 5 concludes the study by proposing ideas for future work and discussing its shortcomings.

II. Related Work

A. Traditional Path Planning Approach

Liu et al. [19] described an Enhanced Genetic Algorithm (EGA) for robot path planning, aimed at optimizing navigation within dynamic and complex environments. The IGA incorporates adaptive functions for crossover and mutation probabilities, along with a reference population, which allows the algorithm to efficiently manage obstacles and enhance path optimization. Their simulation results showed that IGA performs better than regular genetic algorithms by reducing the path length by 20%, reducing the planning time by 25%, and successfully navigating complex environments 98% of the time.

Li et al. [20] proposed an IACA for mobile robot path planning. This algorithm fixes problems with the classical ACA, including delayed convergence, vulnerability to local optima, and large path lengths. Their method improves the search efficiency and resilience by enhancing pheromone initialization, updating, and volatilization rules. To replicate realistic obstacle circumstances, the system was evaluated in a grid-based environment by using 20×20 and 30×30 maps. Their modified algorithm shortens the path length from 30 to 29.01, reduces the number of iterations from 200 to 45, and shortens the execution time from 70.91 to 63.28 s, according to comparative modeling.

Jiachen et al. [21] suggested a hybrid of the Dynamic Window Approach (DWA) and Improved Dung Beetle Optimizer (IDBO) for better path planning in static and dynamic situations for robots. To replicate real-life navigational difficulties, the system was evaluated on grid-based maps with different sizes (15×15 and 20×20) and obstacle densities. IDBO outperformed more conventional algorithms such as DBO, MFO, FPPOA, and MSDBO in several respects. It reached a path length of 24.48 on average, reduced iterations to 29, and maintained a success rate of 100% in static settings. Despite a somewhat longer average computation time of 0.1726 s, it outperformed all of its competitors in terms of correctness and resilience. Furthermore, IDBO exhibited improved convergence speed and stability in benchmark testing in 23 classical functions, consistently reaching or approaching the global optimum.

Chen et al. [22] suggested an RRT Dijkstra-based path-planning method for self-driving cars that operate in partially structured and complicated environments. This approach leverages a Dijkstra optimizer to determine the best course of action in a refined, weighted graph and a Rapidly Exploring Random Tree (RRT) to shrink the enormous search space. All impediments, both static and dynamic, were included in the simulations using occupancy and cost maps. The grid-based map had dimensions of 60 m × 20 m. While maintaining excellent planning accuracy, the proposed approach improved real-time performance by 22% over the Lattice Planner.

Lim et al. [23] introduced the Class-Ordered A* (COA*) algorithm, which is an extension of the standard A* algorithm that uses weighted graph-based path planning with semantic classification (colors). A 2D mobile robot and robotic manipulators were used in the 3D and 5D areas to test COA* in partially visible settings, which are represented by ternary

colored graphs. The graphs encode vertices and edges as feasible, unknown, or infeasible. Their tests showed that COA* routinely chose less risky paths by assigning more weight to semantic information, even when the path lengths were significantly longer than those of traditional A*. In particular, the results demonstrated that COA* significantly outperformed traditional A* in reducing the uncertainty ratio, which is defined as the proportion of unknown edges within the planned paths, particularly in cases of higher dimensions.

Liu et al. [24] developed a Dynamic Genetic Algorithm with Ant Colony Binary Iterative Optimization (DGA-ACBIO) to improve UAV spray routes in hilly areas, focusing on several tea fields. To accurately replicate the environmental conditions, they used the Ovital 3D software to construct 20 polygon-shaped tea fields, each measuring an average of 0.45 hectares. The algorithm incorporates a dynamic genetic algorithm (DGA) that utilizes adaptive crossover and mutation operators, along with an ant colony binary iterative optimization (ACBIO) method characterized by enhanced pheromone update mechanisms. Simulation comparisons demonstrated that DGA-ACBIO exhibited superior performance compared to traditional algorithms, achieving a reduction in the optimal flight path of up to 715.8 meters compared to DGA and a minimum reduction of 287.6 meters in relation to PSO, AFSA, and standalone ACBIO. Furthermore, DGA-ACBIO achieved a reduction in computational time exceeding 50%, demonstrating its enhanced efficiency and effectiveness for UAV route planning in intricate agricultural contexts.

Li et al. [25] presented the IACO algorithm, which can be used to plan global autonomous vehicle routes under complicated road conditions with different types of surface, friction, and turning restrictions. The team utilized a 20×20 grid map to model realistic road conditions, numerically representing road conditions and obstacles to direct the path planning process. The IACO algorithm improves the traditional ACO by incorporating an A*-based heuristic, adaptive pheromone initialization, a modified transition probability function, and a path-trimming mechanism. Based on the simulation results, IACO outperformed regular ACO and A* in generating shorter pathways, achieving faster convergence, and avoiding poor road conditions. The results showed that the convergence curves stabilized quickly and that the time consumption decreased to approximately 23% of that of the default technique, indicating enhanced efficiency and resilience.

Khalidi et al. [26] proposed the T* algorithm, a heuristic-based path planning approach for complex Linear Temporal Logic (LTL) constraints in robotic systems. The algorithm was tested in grid-based environments where robots had to perform tasks in a precise order, avoid obstacles, and follow goals with a time component. To efficiently synthesize pathways that correspond to LTL specifications, T* combines automated planning with traditional A* search. According to the simulation results, T* performed better in terms of path validity, execution duration, and planning success rate than both classical A* and sampling-based techniques.

B. Quantum-Inspired Path Planning Approach

The coverage path planning (CPP) problem is NP-hard; hence, many academics have developed heuristics to find near-optimal solutions, especially for multirobot systems. Evolutionary algorithms, spanning trees, and potential artificial fields have long been used as tools to optimize robot coverage efficiency and coordination. Nevertheless, these methods are far from ideal and often involve issues, such as path overlap. Combinatorial optimization problems such as CPP may provide solutions for new QC techniques, such as quantum annealing (QA) and quantum alternating operator ansatz.

Rao et al. [27] proposed an approach that maximizes multirobot CPP path optimization and exploration using QAOA's quantum benefits of QAOA. The authors showed that quantum heuristics improve runtime efficiency over a depth-first search.

Chella et al. [28] used Grover's algorithm with quantum-based motion path planning to improve grid-based robot navigation. This concept uses quantum search to find the optimal paths faster and minimize the number of iterations. During the experiments, they carefully modified the path length, grid size (10x10), and number of iterations (200). The convergence time and path length are 30% faster and 15% shorter, respectively, than those of the conventional search methods.

Maity et al. [29] examined path planning in autonomous flying robots in a closed room using ACO, Particle Swarm Optimization (PSO), Quantum PSO (QPSO), and a hybrid ABC-PSO strategy. The authors' quantum-inspired PSO trajectory optimization approaches focus on hyperparameters, such as execution time, path length, and iteration count (150 iterations). Multiple-criterion decision making (MCDM) and TOPSIS performance evaluations show that Quantum PSO has a 3.4-unit cost function and a 4.75-second execution time. In the difficult trajectory-planning problem, the QPSO is the most efficient algorithm.

Fan et al. [30] introduced an Improved Quantum Genetic Algorithm (IQGA) to optimize a Linear Quadratic Regulator (LQR) for use in autonomous wheeled tractors and trajectory-tracking control systems. This study refines the state weighting matrix (Q) and control weighting matrix (R) by using IQGA to improve traceability. The simulation results demonstrated that IQGA outperformed GA, PSO, and QGA. IQGA reduced the lateral displacement error to 0.2714 m, longitudinal displacement error to 0.1253 m, and heading angle error to 0.0099 rad when following a circular trajectory at 5 m/s. Using a double-shift trajectory, we obtained lateral displacement errors of 0.1134 m, longitudinal displacement errors of 0.0043 m, and heading angle errors of 0.0004 rad. This study demonstrates how IQGA improves the management accuracy and stability.

Mannone et al. [31] used quantum mechanics to simulate and improve robotic swarms by mimicking ant foraging behavior. This technique depicts local robotic interactions by using quantum gates and circuits. It uses nests (starting locations), food (goal), pathfinding errors (travel time), and the target time. Compared to traditional methods, quantum pathfinding reduces the errors by 18% and the target time by 22%. The swarms had a 30% faster convergence speed to the target than conventional path planning methods.

Schuetz et al. [32] utilized quantum annealing and biased random key genetic algorithms (BRKGA) to optimize the robot trajectory planning in complex industrial situations. We optimized the initial trajectory using BRKGA, altering hyperparameters such as population size (100), elite set size (20), and mutation rate (0.05) for more accurate answers. The quantum annealing component optimizes the original solutions using hyperparameters such as the qubit connectivity and annealing period (1000-5000 μs). Comparing quantum annealing with classical techniques reduces the path cost by 35% and improves execution speed.

Jiao et al. [33] introduced the Improved Quantum Particle Swarm Optimization (IQPSO) algorithm, which optimizes the path planning of escort robots in challenging, obstacle-filled situations. By fusing PSO with QC principles, namely quantum behaviors, the proposed IQPSO improves the search space exploration and exploitation capabilities of the algorithm. Exact optimizations were performed on the critical hyperparameters of the algorithm to guarantee an effective path planning. These included the particle population size (20 particles), maximum iterations (1000), and obstruction penalty coefficients. IQPSO achieved a 20.421 path length and a 172 ms execution time, compared to PSO's 25.182 and 213 ms, respectively, according to the experimental data. IQPSO also outperformed Quantum-PSO (QPSO). This study proves that the IQPSO algorithm can successfully guide escort robots through ever-changing surroundings without causing accidents.

In addition, Atchade-Adelomou et al. [34] introduced qRobot, a hybrid quantum-classical method for warehouse order picking and batch optimization using mobile robots. To reduce the trip distance and enhance the batching procedure, the authors used a variational quantum eigensolver (VQE) algorithm in conjunction with quantum annealing. The method was fine-tuned by testing hyperparameters such as robot capacity (from 1 to 45 items) and the number of qubits (from 10 to 188). QUBO was used to model the problem and convert it into an Ising model for quantum annealing. D-Waves offer faster execution times, particularly for larger problems. Execution times showed better performance than classical approaches for large-scale scenarios, ranging from 1.92 s for two items to 53.28 s for twelve items.

Du et al. [35] presented two algorithms that solve problems with traditional optimization algorithms, namely slow convergence and local optima: the Bloch Spherical Quantum Bee Colony Algorithm (QABC) for multirobot path planning and the Bloch Spherical Quantum Genetic Algorithm (BQGA) for single-robot path planning. To increase the diversity of search space and global search capabilities, quantum algorithms use spherical Bloch coordinates. We ran simulations using the following critical hyperparameters: population size (100-200), number of quantum bits (n=2), and evolutionary generations (100). The estimated path length was 12.628 m, the actual path length was 12.682 m, and the execution time was 1.153 s for the QABC, which is a considerable improvement over conventional algorithms such as ACA and Genetic Algorithm (GA). These findings demonstrate the potential of quantum optimization methods for improving multi-robot route planning in complicated settings.

Lathrop et al. [36] put forward quantum search methods to improve sampling-based motion planning, namely q-RRT and quantum full path search (q-FPS). The authors used QAA to improve the pathfinding efficiency by increasing the likelihood of finding solutions without collisions in less time. Some of the hyperparameters of the q-RRT algorithm that were investigated include the expansion steps, number of iterations, and oracle calls. Compared with classical RRT, the q-RRT method showed a quadratic speedup and, reduced execution time by half while minimizing oracle calls. These enhancements illustrate that quantum methods work well for optimizing pathfinding in complicated settings while simultaneously decreasing computing complexity.

Li et al. [37] presented the application of a Quantum Genetic Algorithm (QGA) to optimize the climb trajectory of Airbus A-320, with the objective of reducing flight time and fuel consumption. QGA improves the convergence speed and search capabilities by integrating quantum principles, surpassing the performance of classical genetic algorithms. The algorithm underwent fine tuning through the adjustment of critical hyperparameters, including a population size of 300 and 150 evolutionary generations, crossover probability of 0.8, and mutation probability of 0.05. The results indicate notable enhancements, with a climb time recorded at 270 s and fuel consumption measured at 200 kg, highlighting the capabilities of quantum optimization within the aviation sector.

Chella et al. [38] developed a quantum-based path-planning system for swarm robotics that uses quantum gates and Grover's search algorithm to optimize a robot's interaction and mobility. Robots in a swarm exchange data and redirect themselves to reach their objectives faster and more accurately, similar to ants. The quantum approach converges 30% faster than classical algorithms, finding the optimal solution 30% faster. By minimizing the path length, the trajectory length was reduced by 18%, making the movement more efficient. By reducing the convergence time and extending the path length, QC improves swarm robots under complex circumstances.

III. RESEARCH METHODOLOGY

This section describes a rigorous methodology for a unified pathfinding framework that integrates two quantum-enhanced algorithms, the QAOA-based pathfinder and the CQRW-based pathfinder, with the classical A* algorithm. The framework was designed to navigate a two-dimensional (2D) environment with polygonal obstacles, allowing a systematic comparison of quantum and classical pathfinding strategies in terms of path quality, computational efficiency, and robustness. The methodology encompasses environmental modeling, graph construction, algorithmic formulations, path optimization, and performance evaluation, supported by precise mathematical representations and a robust theoretical foundation. This work was executed using a Python-based implementation, leveraging libraries such as PennyLane for quantum simulations, NetworkX for graph operations, and Matplotlib for visualization, ensuring reproducibility and scalability.

A. Problem Formulation

Let the environment be represented by a compact, bounded subset of two-dimensional Euclidean space:

$$\mathcal{C} \subset \mathbb{R}^2$$

divided into a free configuration space $\mathcal{C}_{\text{free}}$ and an obstacle space $\mathcal{C}_{\text{obs}}$, such that:

$$\mathcal{C} = \mathcal{C}_{\text{free}} \cup \mathcal{C}_{\text{obs}}, \quad \mathcal{C}_{\text{free}} \cap \mathcal{C}_{\text{obs}} = \emptyset.$$

Given a start position $s \in \mathcal{C}_{\text{free}}$, a goal position $g \in \mathcal{C}_{\text{free}}$ and a finite set of polygonal obstacles $\mathcal{O} = \{\mathcal{O}_1, \mathcal{O}_2, \ldots, \mathcal{O}_m\}$, where $\mathcal{O}_i \subset \mathcal{C}_{\text{obs}}$. The objective is to compute a continuous and collision-free trajectory $\pi : [0,1] \to \mathcal{C}_{\text{free}}$ that connects s to g and minimizes the cost function. Formally, the optimal trajectory $\pi^*(t)$ satisfies the following:

$$\pi^*(t) = \arg\min_{\pi} \int_0^1 \mathcal{L}(\pi(t), \dot{\pi}(t))\, dt \tag{1}$$

subject to:

$$\pi(0) = s, \quad \pi(1) = g, \quad \pi(t) \in \mathcal{C}_{\text{free}} \quad \forall t \in [0,1],$$

where $\mathcal{L}$ is the Lagrangian that captures the path length and proximity of the obstacle to the obstacles. Because direct optimization in a continuous space is computationally intractable, we discretize the domain and solve the graph-based approximation of the problem.

B. Environment Modeling

The environment is modeled as a 2D rectangular domain:

$$\mathcal{C} = [x_{\min}, x_{\max}] \times [y_{\min}, y_{\max}] \tag{2}$$

Each obstacle $\mathcal{O}_i \in \mathcal{O}$ is a closed polygon defined by an ordered sequence of vertices:

$$\begin{aligned} \mathcal{O}_i = &\{(x_1, y_1), (x_2, y_2), \ldots, (x_{n_i}, y_{n_i})\}, \\ &\text{with } (x_{n_i+1}, y_{n_i+1}) = (x_1, y_1) \end{aligned} \tag{3}$$

The configuration space was discretized using a uniform grid with a resolution $\delta > 0$, producing the following candidate positions:

$$\begin{aligned} \mathcal{P} = &\{(x, y) \in \mathcal{C} \mid x = x_{\min} + i\delta, \\ &y = y_{\min} + j\delta,\ i, j \in \mathbb{Z}_{\geq 0}\} \end{aligned} \tag{4}$$

Using a point-in-polygon algorithm, we eliminate all points inside the obstacles to yield a free point set as follows:

$$\mathcal{P}_{\text{free}} = \mathcal{P} \setminus \left(\bigcup_{i=1}^{m} \mathcal{O}_i \right) \tag{5}$$

This discrete set of obstacle-free point sets forms the basis for constructing a visibility graph.

C. Visibility Graph Construction

We construct a visibility graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where:

$$\mathcal{V} = \mathcal{P}_{\text{free}} \cup \{s, g\} \tag{6}$$

includes all the valid free-space points along the start and goal locations.

An edge $(u, v) \in \mathcal{E}$ exists between two vertices if the straight-line segment connecting them lies entirely within $\mathcal{C}_{\text{free}}$:

$$(u, v) \in \mathcal{E} \iff \text{LineSegment}(u, v) \subset \mathcal{C}_{\text{free}} \tag{7}$$

For computational efficiency, each node is connected to its k-nearest visible neighbors using a KD tree for neighbor queries. The weight of each edge is the Euclidean distance between the corresponding vertices:

$$w(u, v) = \|u - v\|_2 = \sqrt{(x_u - x_v)^2 + (y_u - y_v)^2} \tag{8}$$

This visibility graph serves as the common search space across all three algorithms evaluated in this study.

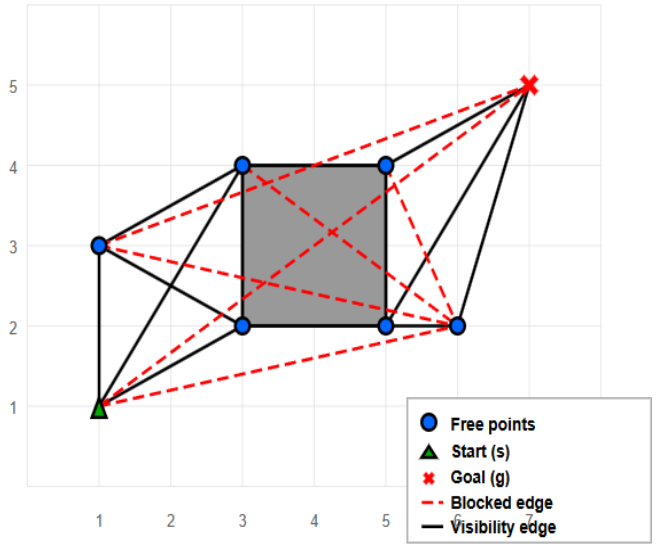

Fig. 1: Example of visibility graph construction in a polygonal environment. Blue dots represent free-space points $\mathcal{P}_{\text{free}}$, with the start point s shown as a green triangle and the goal point g as a red cross. Solid black lines indicate visibility edges where line-of-sight is unobstructed within the free space $\mathcal{C}_{\text{free}}$, while red dashed lines represent blocked edges that intersect obstacles (shaded regions).

As shown in Figure 1, the visibility graph effectively captures all feasible direct connections between points within free space, excluding edges that intersect obstacles, which are illustrated as blocked edges (red dashed lines). This structure provides an efficient search space for path planning algorithms.

D. A* Pathfinding Algorithm

The A* algorithm is a widely used pathfinding method that efficiently determines the optimal path in graph-based representations of continuous environments, such as the visibility

graph constructed in this study. It combines the advantages of both uniform cost search and greedy best-first search, allowing a systematic exploration of a search space by considering both the accumulated cost to reach a node and a heuristic estimate of the remaining cost to reach the goal. This section formalizes the A* algorithm and describes the components and procedures used to determine the optimal path.

Given start node s and goal node g in a visibility graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, the A* algorithm searches for an optimal path that minimizes the total cost $f(n)$, which is the sum of two terms:

$$f(n) = g(n) + h(n) \tag{9}$$

where $g(n)$ is the cost to reach node n from start node s and $h(n)$ is the heuristic estimate of the minimum cost from node n to goal node g. The objective is to find the path π^* that minimizes $f(n)$ while ensuring that the path from s to g remains within the valid region $\mathcal{C}_{\text{free}}$ (i.e. avoiding obstacles). The optimal path is computed as follows:

$$\pi^* = \arg\min_{\pi} \left(g(\pi) + h(\pi)\right) \tag{10}$$

where $g(\pi)$ represents the cost of the path from the start to any node $n \in \mathcal{V}$, and $h(\pi)$ represents the heuristic cost to reach the goal node g. The heuristic function is a critical component of the A* algorithm, as it guides the search towards the goal. A common choice for the heuristic in pathfinding problems is the Euclidean distance between two points $n = (x_n, y_n)$ and $G = (x_G, y_G)$, which is given by:

$$h_{\text{euclidean}}(n, G) = \sqrt{(x_n - x_G)^2 + (y_n - y_G)^2} \tag{11}$$

However, we improve this heuristic by incorporating the influence of obstacles into the pathfinding process. The obstacle influence function is introduced to account for the additional cost imposed by obstacles in the pathfinding process. The influence was computed based on the distance between a given node and its nearest obstacle. For a point $p = (x_p, y_p)$, the obstacle influence is given by:

$$I(p) = \sum_{i=1}^{m} \frac{1}{d(p, O_i)^2} \tag{12}$$

where $d(p, O_i)$ is the distance from point p to the nearest point on obstacleO_i, and m is the number of obstacles in the environment. The total heuristic $h(n)$ was adjusted as follows:

$$h(n) = h_{\text{euclidean}}(n, G) \times (1 + I(n)) \tag{13}$$

where $h_{\text{euclidean}}(n, G)$ is the standard Euclidean heuristic and $I(n)$ is the obstacle influence. This adjustment increases the heuristic value for nodes near obstacles, steering the search away from such areas and towards clear spaces. The A* algorithm identifies the optimal path by iteratively selecting the node with the smallest $f(n)$ value from the open list, exploring its neighbors and updating the $g(n)$ and $f(n)$ values using heuristic $h(n)$, which accounts for both Euclidean distance and obstacle influences. Once the goal is reached, the path is reconstructed by backtracking the nodes using predecessor links, that store the parent node for each node in the path. This ensures the exploration of the most efficient path to the goal.

E. QAOA-A for Pathfinding in Continuous Spaces with Obstacles*

The Quantum Approximate Optimization Algorithm (QAOA) is a hybrid quantum-classical variational method that finds approximate solutions to discrete combinatorial optimization problems. The QAOA prepares a parameterized quantum state whose expected energy with respect to a carefully designed cost Hamiltonian corresponds to the objective function of the problem. Through classical optimization of the variational parameters, the algorithm aims to approximate the ground state of this Hamiltonian, thereby identifying near-optimal solutions.

In the context of the pathfinding problem addressed in the provided code, QAOA was adapted to select the optimal next move from the current node within a discretized navigation graph constructed using a visibility graph approach. Specifically, at each step of the search, given a set of possible next states (neighbors) $\{n_1, n_2, \ldots, n_k\}$ from the current position, QAOA helps determine the optimal next state by minimizing the expectation value of the problem Hamiltonian H_P.

The QAOA circuit consists of alternating applications of two Unitary operators:

i) Unitary $U_P(\gamma) = e^{-i\gamma H_P}$
ii) Mixer unitary $U_M(\beta) = e^{-i\beta H_M}$

For p layers, the QAOA state is given by:

$$|\psi(\boldsymbol{\gamma}, \boldsymbol{\beta})\rangle = U_M(\beta_p)U_P(\gamma_p)\ldots U_M(\beta_1)U_P(\gamma_1)|\psi_0\rangle \tag{14}$$

where $|\psi_0\rangle$ is the initial state and $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_p)$, $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)$ are the variational parameters to be optimized. Unlike the standard QAOA approach, where the initial state $|\psi_0\rangle$ is typically the uniform superposition $|+\rangle^{\otimes n}$, here the initial state is explicitly prepared by applying parameterized single-qubit rotations encoding classical problem data. This data-driven initialization embeds spatial and directional information into the quantum register before the alternating problem and mixer unitaries are applied, thereby enabling the QAOA circuit to leverage domain-specific knowledge.

1) Problem Hamiltonian Construction

The problem Hamiltonian H_P is the cornerstone of the QAOA framework. It encodes the optimization objective, that is, the cost of selecting each possible next node in the pathfinding graph, directly into a quantum operator acting on the Hilbert space of n qubits. It is constructed as follows:

$$H_P = \sum_{i=1}^{k} C(n_i)|i\rangle\langle i| \tag{15}$$

where $C(n_i)$ is the cost associated with selecting neighbor n_i, defined as:

$$C(n_i) = w_1 \cdot d(n_i, \mathbf{g}) + w_2 \cdot P_{\mathcal{O}}(n_i) + w_3 \cdot d(c, n_i) \tag{16}$$

where $d(n_i, \mathbf{g})$ is the Euclidean distance from n_i to goal $\mathbf{g}$, $P_{\mathcal{O}}(n_i)$ is an obstacle proximity penalty, $d(c, n_i)$ is the distance from the current node c to n_i and w_1, w_2, w_3 are the weighting factors.

*2) **Mixer Hamiltonian***

Whereas the problem Hamiltonian H_P encodes the cost landscape, the mixer Hamiltonian H_M enables the quantum state to explore the solution space beyond the fixed initial configuration. This facilitates transitions between candidate solutions, allowing the QAOA to escape local minima and leverage quantum interference to obtain better approximations. The mixer Hamiltonian facilitates transitions between different basis states and is given by:

$$H_M = \sum_{i=0}^{n-1} \sigma_x^{(i)} \tag{17}$$

where $\sigma_x^{(i)}$ is the Pauli-X operator acting on the i th qubit, and n is the number of qubits used.

State Encoding

We mapped spatial information and goal orientation onto the quantum state to encode the state of the pathfinding problem into a quantum circuit. For an n-qubit system, we use the following equation:

$$|\psi_{\text{init}}\rangle = \bigotimes_{i=0}^{n-1} |\psi_i\rangle \tag{18}$$

where the first four qubits encode the normalized current position: $|\psi_{0-3}\rangle = f_{\text{pos}}(\mathbf{c})$, next two qubits encode the direction to the goal: $|\psi_{4-5}\rangle = f_{\text{dir}}(\mathbf{g}-\mathbf{c})$ and the last two qubits encode the normalized distance to the goal: $|\psi_{6-7}\rangle = f_{\text{dist}}(\|\mathbf{g}-\mathbf{c}\|_2)$.

The encoding functions f_{pos}, f_{dir}, and f_{dist} map their respective inputs to appropriate qubit rotations:

$$f_{\text{pos}}(\mathbf{c}) = R_Y(\pi \cdot \tilde{x}) \otimes R_X(\pi \cdot \tilde{x}) \otimes R_Y(\pi \cdot \tilde{y}) \otimes R_X(\pi \cdot \tilde{y}) |0000\rangle \tag{19}$$

$$f_{\text{dir}}(\vec{v}) = R_Y(\pi \cdot \tilde{\theta}) \otimes R_Z(\pi \cdot \tilde{\theta}) |00\rangle \tag{20}$$

$$f_{\text{dist}}(d) = R_Y(\pi \cdot \tilde{d}) \otimes R_Z(\pi \cdot \tilde{d}) |00\rangle \tag{21}$$

where $\tilde{x}$, $\tilde{y}$, $\tilde{\theta}$, and $\tilde{d}$ are normalized values between 0 and 1.

*3) **QAOA Parameter Optimization***

The QAOA parameters $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ are optimized by minimizing the expectation value of the problem Hamiltonian as follows:

$$\min_{\boldsymbol{\gamma},\boldsymbol{\beta}} \langle \psi(\boldsymbol{\gamma},\boldsymbol{\beta}) | H_P | \psi(\boldsymbol{\gamma},\boldsymbol{\beta}) \rangle \tag{22}$$

This optimization is performed using classical methods such as COBYLA with the objective function:

$$F(\boldsymbol{\gamma},\boldsymbol{\beta}) = \langle \psi(\boldsymbol{\gamma},\boldsymbol{\beta}) | H_P | \psi(\boldsymbol{\gamma},\boldsymbol{\beta}) \rangle \tag{23}$$

*4) **QAOA-based Heuristic***

Once optimized, the quantum circuit produces a parameterized state that amplifies the amplitudes of low-cost neighboring states. By measuring the expectation values of the Pauli-Z operators on each qubit,

$$\langle Z_j \rangle = \langle \psi | Z_j | \psi \rangle \tag{24}$$

A quantum heuristic value is computed by combining the quantum solution structure with the classical distance to the goal as follows:

$$h_{\text{QAOA}}(v_c) = \left(1 + \alpha \sum_{j=1}^{n} |\langle Z_j \rangle|\right) \cdot d(v_c, v_{\text{goal}}) \tag{25}$$

where α is a weighting parameter and $d(v_c, v_{\text{goal}})$ denotes the classical distance from the current node v_c to the goal node v_{goal}.

The QAOA-based heuristic framework transforms the classical pathfinding cost function into a quantum energy landscape, enabling the simultaneous exploration of multiple potential next-generation options through quantum superposition. By leveraging quantum interference effects during parameterized unitary evolutions, the algorithm inherently amplifies the states corresponding to low-cost, obstacle-avoiding paths. The variational optimization of QAOA parameters systematically tunes the quantum circuit to emphasize these promising directions, thereby encoding a sophisticated, non-classical cost evaluation that accounts for obstacle proximity and path length. Consequently, this quantum-enhanced heuristic surpasses the conventional Euclidean distance estimates in terms of sensitivity to complex environmental constraints, thereby effectively pruning the search space. This enhanced guidance mechanism can lead to a more efficient exploration and faster convergence in pathfinding tasks within intricate continuous domains featuring polygonal obstacles.

F. CQRW-A for Pathfinding in Continuous Spaces with Obstacles*

The CQRW-A* pathfinding algorithm leverages the computational advantages of quantum walks to augment the heuristic function of the A* search, thereby enabling an efficient navigation in complex environments. Quantum walks, a quantum analog of classical random walks, exploit superposition, interference, and entanglement to explore multiple paths simultaneously, thereby offering potential speedups in optimization problems [5], [39]. In contrast to discrete quantum walks, continuous quantum walks evolve under a time-independent Hamiltonian, making them particularly suitable for modeling continuous-space dynamics [40]. The A* algorithm, which is the cornerstone of heuristic-based pathfinding, guarantees optimality when equipped with an admissible heuristic [41]. However, classical heuristics often fail to capture complex environmental features, such as obstacle proximity or global connectivity. The CQRW addresses this limitation by providing a quantum-inspired heuristic that evaluates the probabilistic flow toward the goal and improves the efficiency and

Algorithm 1 QAOA-A* algorithm for Pathfinding

function QAOAPATHFINDING(G, s, g)
  Input: Graph $G = (V, E)$, start s, goal g
  Output: Optimal path $\mathcal{P}$
  Initialize: $Q \leftarrow \{(h_Q(s, g), s)\}$, visited $\leftarrow \emptyset$
  $\boldsymbol{\theta}^* \leftarrow$ OPTIMIZEQAOA(s, g)
  while $Q \neq \emptyset$ **do**
  end
  $(f, u) \leftarrow$ POP(Q) **if** $u = g$ **then**
  return
  end
  RECONSTRUCTPATH(u)
  visited $\leftarrow$ visited $\cup \{u\}$ **forall** $v \in$ *Neighbors*(u) $\setminus$ *visited* **do**
  end
  $g_{new} \leftarrow g(u) + w(u, v)$
  $f_{new} \leftarrow g_{new} + h_Q(v, g; \boldsymbol{\theta}^*)$
  PUSH(Q, (f_{new}, v))
end function

function $h_Q(u, g; \boldsymbol{\theta})$
  Input: Current node u, goal g, QAOA parameters $\boldsymbol{\theta}$
  Output: Quantum heuristic value
  $\rho_0 \leftarrow$ ENCODESTATE(u, g) ▷ Encode problem state
  $H_P \leftarrow \sum_i c_i |i\rangle\langle i|$ ▷ Problem Hamiltonian
  $H_M \leftarrow \sum_j X_j$ ▷ Mixer Hamiltonian
  $\rho \leftarrow \rho_0$ **for** $p = 1$ ***to*** P **do**
  end
  $\rho \leftarrow e^{-i\gamma_p H_P} \rho e^{i\gamma_p H_P}$
  $\rho \leftarrow e^{-i\beta_p H_M} \rho e^{i\beta_p H_M}$
  return $(1 + \alpha \cdot \mathrm{Tr}(\rho Z)) \cdot \|u - g\|_2$
end function

function OPTIMIZEQAOA(s, g)
  $\boldsymbol{\theta}_0 \sim \mathcal{U}(0, 2\pi)^{2P}$ ▷ Random initialization
  $\mathcal{L}(\boldsymbol{\theta}) = \langle \psi(\boldsymbol{\theta}) | H_P | \psi(\boldsymbol{\theta}) \rangle$
  return $\arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta})$ ▷ Classical optimization
end function

robustness of the A* search. Following the construction of a visibility graph representation $\mathcal{G}$ of a continuous environment, we introduce a novel pathfinding paradigm that exploits the intrinsic advantages of CQRW to navigate complex spatial domains with enhanced efficiency.

1) Continuous Quantum Random Walk (CQRW)

The foundation of our approach is the integration of quantum walk dynamics with the classical A* search algorithm. We reformulate the canonical A* framework using a quantum-mechanical lens, where the evaluation function is expressed as:

$$f(n) = g(n) + h_{\text{quantum}}(n) \tag{26}$$

Here, $g(n) : V \to \mathbb{R}^+$ represents the accumulated cost from start node s to node $n \in V$ and $h_{\text{quantum}}(n) : V \to \mathbb{R}^+$ is our quantum-derived heuristic that estimates the cost from n to goal g. The classical A* algorithm guarantees the optimal paths when the heuristic function is admissible and consistent. Our quantum heuristic maintains these properties while leveraging the quantum interference effects to explore the solution space more efficiently. The optimality criterion for our quantum-enhanced A* search can be formulated as:

$$\mathcal{P}^* = \arg\min_{\mathcal{P} \in \Pi} \sum_{(v_i, v_j) \in \mathcal{P}} w(v_i, v_j) \tag{27}$$

where Π represents the set of all feasible paths from s to g in the graph G. While classical random walks perform random steps, quantum walks exhibit non-local behavior owing to quantum superposition and interference, allowing the system to explore multiple paths simultaneously. In this algorithm, a CQRW is employed over a discretized visibility graph representation of the environment to evaluate potential paths. The quantum walk evolves according to the Schrödinger equation:

$$i\frac{d}{dt}|\psi(t)\rangle = H|\psi(t)\rangle \tag{28}$$

where the Hamiltonian H is the weighted Laplacian matrix L of the discretized visibility graph, which represents the connectivity of the environment. The Laplacian matrix is the difference between the degree matrix (D) and the adjacency matrix (A) of the graph, where:

$$L = D - A, \tag{29}$$

Here, weighted adjacency matrix A and degree matrix D were used, where $D_{ii} = \sum_j A_{ij}$. The entries in the adjacency matrix A_{ij} for an edge (v_i, v_j) are calculated based on a distance-based weight, goal direction bias, and obstacle proximity penalty to prioritize shorter, goal-directed paths while avoiding obstacles.

In the context of the CQRW, a Laplacian matrix is used to describe the evolution operator that governs the dynamics of the quantum walk. The Hamiltonian governing the system is related to the negative Laplacian as follows:

$$H = -\gamma \mathbf{L} \tag{30}$$

where γ is a coupling constant, which is often set to unity in discrete implementations for simplicity.

The state evolves over a total time T over an evolution operator, discretized into N steps as follows:

$$|\psi(T)\rangle = \exp(-iHT)|\psi(0)\rangle.$$

To simulate the quantum walk, we used a series of discretized time steps Δt, evolving the quantum state step-by-step over the graph. The number of time steps n determines the precision of the quantum walk simulation and is a key parameter in controlling the resolution of the pathfinding

process. The evolution is approximated using a Trotter-like discretization:

$$|\psi(t+\Delta t)\rangle \approx \exp(-iH\Delta t)|\psi(t)\rangle, \quad \Delta t = \frac{T}{N}. \tag{31}$$

Each time step t leads to an updated quantum state, and the system evolves towards a probability distribution that can be used to estimate the likelihood of reaching various nodes. This is computed efficiently by using sparse matrix exponentiation. The final state yields the following probabilities:

$$P_i = |\langle i|\psi(T)\rangle|^2, \quad i \in V. \tag{32}$$

These probabilities reflect the probability of reaching node i after time T, with interference effects that amplify paths that align with the goal and avoid obstacles. The CQRW is grounded in quantum mechanics, where the Laplacian Hamiltonian ensures that the walk respects the topology of the graph [40]. The weighted adjacency matrix incorporates environmental priors (goal direction and, obstacle avoidance), making the walk context-aware. Unlike classical random walks, which diffuse uniformly, the interference of quantum walks amplifies the probabilities along favorable paths, providing a richer heuristic signal [5]. The continuous-time formulation is ideal for continuous-space pathfinding because it avoids the lattice constraints of discrete walks, and the local subgraph approach balances the computational cost with global exploration.

*2) **Quantum-Enhanced A* Heuristic***

A quantum walk is used to calculate a quantum heuristic that estimates the likelihood of reaching a goal from a given state. This heuristic is based on the probability distribution generated by the quantum walk and is used to guide the pathfinding algorithm (A*) by assigning a higher priority to paths that have a higher probability of reaching the goal. For a given node i in the graph, the quantum walk heuristic $h_q(i)$ is calculated as:

$$h_quantum(i) = \sum_j |\psi_j(t)|^2 \cdot \frac{1}{1+d(i,G)} \tag{33}$$

where $\psi_j(t)$ represents the amplitude of the j-th node at time t, and $d(i,G)$ is the distance from node i to goal node G. The heuristic is designed to encourage paths with high quantum probabilities and those that are closer to the goal, thereby allowing for more efficient exploration of the solution space. The quantum heuristic was combined with a classical Euclidean heuristic (straight-line distance between nodes) to balance quantum pathfinding exploration with classical search strategies. The final heuristic used for the A* search algorithm is the weighted sum of quantum and classical heuristics:

$$h(i) = \alpha \cdot h_q(i) + (1-\alpha) \cdot \text{Euclidean}(i,G) \tag{34}$$

where α is a tuning parameter that balances the influence of the quantum and classical heuristics.

The heuristic is computed for each expansion of the node in the A* search, which maintains a priority queue (open list) of nodes ordered by $f(v)$. For each node v, the neighbors are evaluated, and $g(v')$, $h(v')$, and $f(v')$ are updated if a better path is found. The path is reconstructed when g is reached. The quantum heuristic outperforms classical heuristics by capturing the global connectivity through quantum interference and reducing the number of nodes explored [39]. The probability P_g reflects the preference for the quantum walk for the paths leading to the goal, whereas the fallback case accounts for scenarios where g is outside the local subgraph, using proximity-weighted probabilities to approximate path quality. The goal direction bias and obstacle penalty in the Laplacian ensure that the heuristic is informed by the environmental constraints, thereby guiding the search toward feasible and efficient paths. Admissibility guarantees optimality, whereas the probabilistic nature of quantum walk enhances robustness against local minima, particularly in environments with complex obstacle configurations.

Algorithm 2 CQRW-A* algorithm for Pathfinding

Input: Graph $G = (V, E)$, start S, goal G_{goal}
Output: Optimal path P from S to G_{goal}
Initialize: $\mathcal{O} \leftarrow \{S\}$, $\mathcal{C} \leftarrow \emptyset$, $g(n) \leftarrow \infty$, $f(n) \leftarrow \infty\ \forall n$
Set $g(S) \leftarrow 0$, $f(S) \leftarrow$ QUANTUMHEURISTIC(S, G_{goal}, G)
while $\mathcal{O} \neq \emptyset$ **do**
end
$c \leftarrow \arg\min_{n \in \mathcal{O}} f(n)$ **if** $c = G_{goal}$ **then**
end
return reconstructed path P from S to G_{goal}
$\mathcal{O} \leftarrow \mathcal{O} \setminus \{c\}$; $\mathcal{C} \leftarrow \mathcal{C} \cup \{c\}$ **for** *each neighbor* n *of* c *in* G **do**
end
$n \in \mathcal{C}$ **continue**
$g_{\text{tent}} \leftarrow g(c) + w(c,n)$ **if** $g_{tent} < g(n)$ **then**
end
$g(n) \leftarrow g_{\text{tent}}$
$f(n) \leftarrow g(n) +$ QUANTUMHEURISTIC(n, G_{goal}, G)
Add n to $\mathcal{O}$ if not present
return null (no path found)
function QUANTUMHEURISTIC(n, G_{goal}, G)
Extract local subgraph $\mathcal{L}$ from G centered at n
Build weighted Laplacian L for $\mathcal{L}$ with goal-bias and obstacle-penalty
Initialize quantum state ψ_0 localized at n
Evolve: $\psi(t) \leftarrow \exp(-iLt)\psi_0$
Compute probabilities $p_i = |\psi_i(t)|^2$
Return $h(n) \leftarrow \alpha / p_{\text{goal}} + (1-\alpha) \cdot d(n, G_{goal})$
end function

In Figure 3, a workflow diagram is presented that provides a framework for evaluating quantum-enhanced path planning algorithms in comparison to the classical A* method.

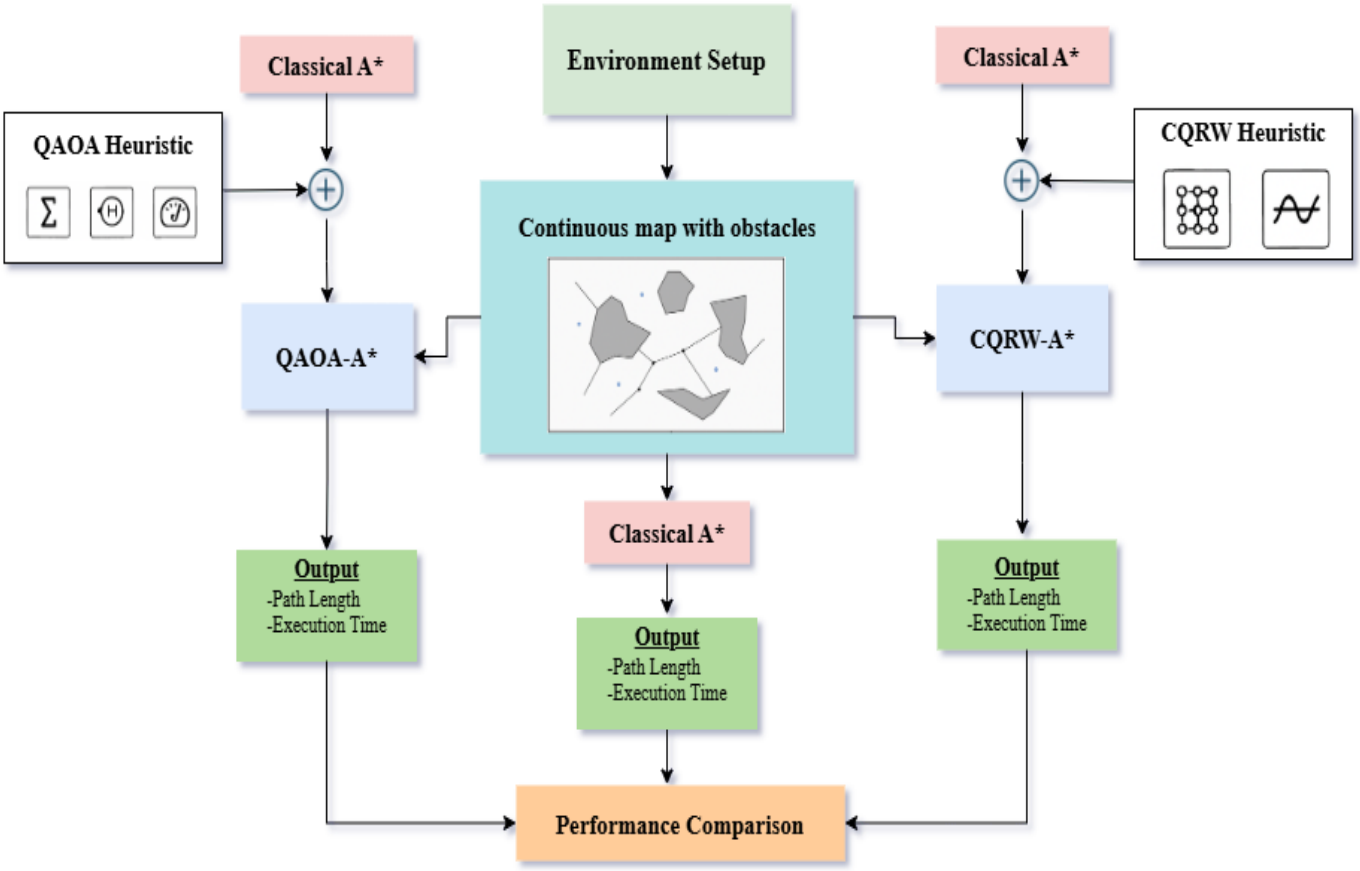

Fig. 3: Workflow methodology.

IV. Experiment Design And Result Analysis

The performances of three different pathfinding algorithms, namely Classical A*, QAOA-A*, and CQRW-A*, were evaluated and compared. For the purpose of evaluating these algorithms, grids of dimensions 20×20 and 30×30 were utilized. Additionally, the algorithms were assessed using various obstacle densities. The performance was evaluated based on a set of criteria that included the actual length of the path and the amount of was required for the computation.

A. *Experiment Setting*

The experiment was conducted on two different grid sizes, 20×20 and 30×30, with obstacle densities of 17%, 30%, and 35% that as shown in Figure 4. According to the obstacle density, the 20×20 grid had 4-13 obstacles for each scenario, whereas the 30×30 grid had 7-32 obstacles. For consistency, all configurations had the same movement step size and direction set to eight. In both the 20×20 and 30×30 grids, the starting point was always (2,2), and the goal points were (18,18) and (28,28), respectively. The simulations were run on a classical computer system with 8GB of RAM and an 8th-generation Core i5 processor using the latest version of PennyLane (0.41.1). Finally, we evaluate the performance of each algorithm by comparing its path length with its execution time.

TABLE I: Comparison of Classical-A*, QAOA-A*, and CQRW-A* for obstacle density of 17% with grid size 20×20.

Algorithm	Actual Path Length	Time/s
Classical-A*	32.57	0.073
QAOA-A*	28.67	121.66
CQRW-A*	**26.09**	65.74

Table I presents a comparative analysis of the three pathfinding algorithms. The classical A* algorithm serves as the baseline with a path length of 32.57 units and a computation time of 0.073s. Both the quantum-enhanced algorithms demonstrated improvements in path optimization, albeit at the cost of increased computation time. QAOA-A* achieves a significant reduction in the path length to 28.67 units, representing approximately a 12% improvement over the classical approach, but requires a substantially longer computation time of 121.66s. Similarly, the CQRW-A* algorithm provides the shortest path length of 26.09 units, outperforming both the classical and the QAOA-enhanced algorithms in path efficiency by approximately 20% and 9%, respectively. The computation time of 65.74s is notably shorter than that of the QAOA-enhanced method, although it is still considerably higher than that of the classical A*. In this setup, figures 5, 6, and 7 illustrate the path planning visualizations for classical A*, QAOA-A*, and CQRW-A*, respectively, on a 20x20 grid with an obstacle density of 17%.

Algorithm	Parameter								
	Grid size	Density	Sorted numbers	Obstacle numbers	Obstacle blocks	Moving direction	Moving step size	Start Point	Goal Point
Classical A*	20*20	17%	400	4	20	8	1	(2,2)	(18,18)
		30%		8					
		35%		13					
	30*30	17%	900	7	30				(28,28)
		30%		10					
		35%		32					
QAOA enhanced A*	20*20	17%	400	4	20				(18,18)
		30%		8					
		35%		13					
	30*30	17%	900	7	30				(28,28)
		30%		10					
		35%		32					
CQRW enhanced A*	20*20	17%	400	4	20				(18,18)
		30%		8					
		35%		13					
	30*30	17%	900	7	30				(28,28)
		30%		10					
		35%		32					

Fig. 4: Illustration of the experiment setup.

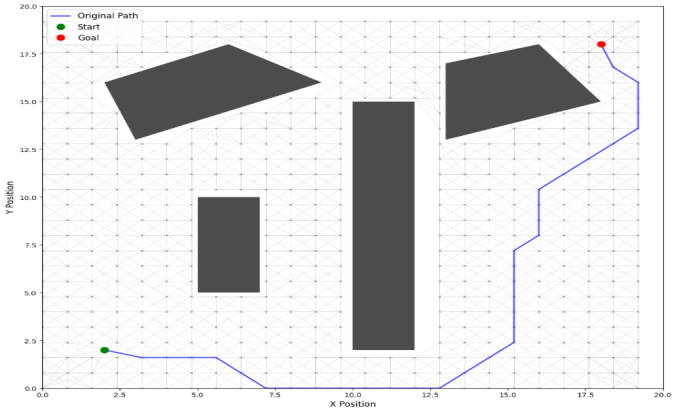

Fig. 5: Schematic diagram of path planning using 20×20 grid with 17% obstacle density for the classical A* algorithm.

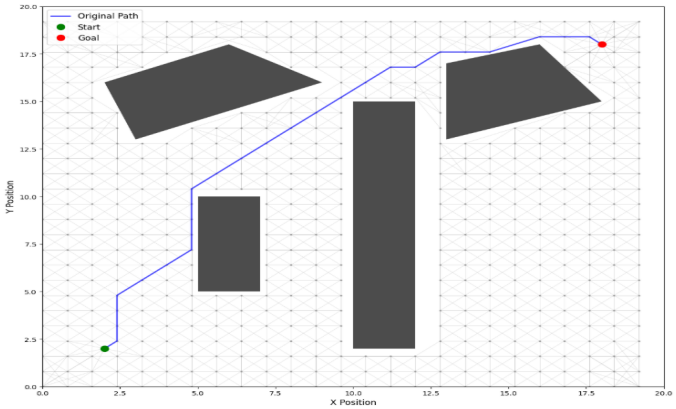

Fig. 6: Schematic diagram of path planning using 20×20 grid with 17% obstacle density for the CQRW-A* algorithm.

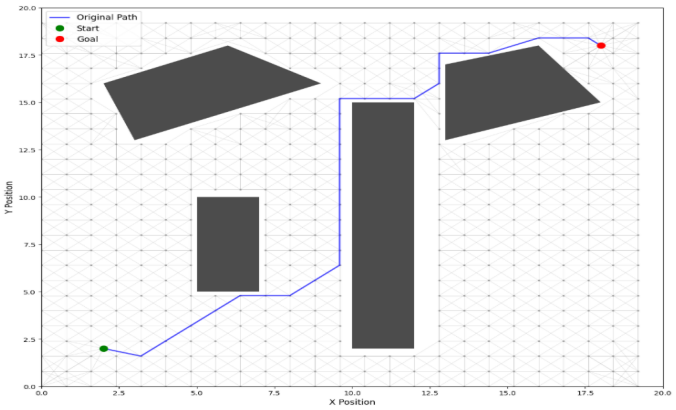

Fig. 7: Schematic diagram of path planning using 20×20 grid with 17% obstacle density for the QAOA-A* algorithm.

TABLE II: Comparison of Classical-A*, QAOA-A*, and CQRW-A* for obstacle density of 30% with grid size 20×20.

Algorithm	Actual Path Length	Time/s
Classical-A*	27.73	0.64
QAOA-A*	**21.71**	90.09
CQRW-A*	26.37	66.26

Table II shows the performance of the Classical-A*, QAOA-A*, and CQRW-A* algorithms on a 20×20 grid with eight obstacles covering 30% of the area. Classical A* and CQRW-A* found paths that were 27.73 and 26.37 units long, respectively, but QAOA-A* found the shortest path at 21.71 units, showing that it works better. This experiment indicates that QAOA may improve its pathfinding in complicated contexts by utilizing its quantum optimization capability. However, this

benefit comes at a large cost in terms of calculation time: QAOA-A* took 90.09s to do the work, while CQRW-A* took 66.26s, and Classical-A* took only 0.64s. Although it sacrifices path optimality for speed, the Classical-A* algorithm provides the fastest response.

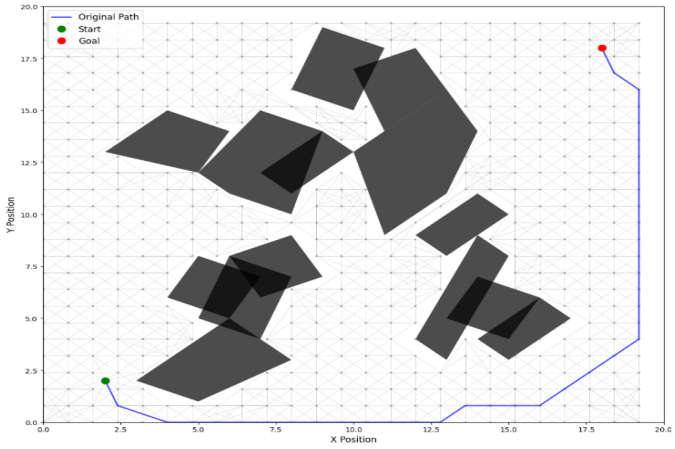

Fig. 8: Schematic diagram of path planning using 20×20 grid with 30% obstacle density for the classical A* algorithm.

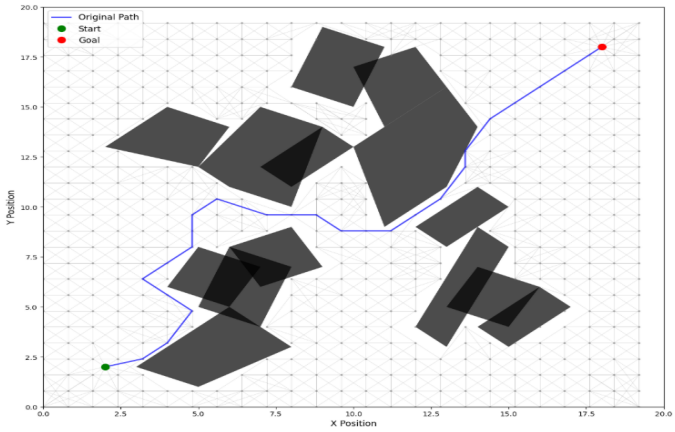

Fig. 9: Schematic diagram of path planning using 20×20 grid with 30% obstacle density for the QAOA-A* algorithm.

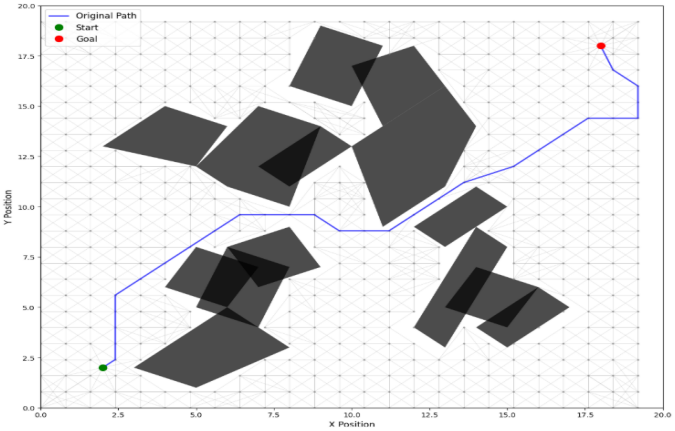

Fig. 10: Schematic diagram of path planning using 20×20 grid with 30% obstacle density for the CQRW-A* algorithm.

The route planning for this setup is shown in Figures 8, 9, and 10 for classical A*, CQRW-A*, and QAOA-A*, respectively, on a 20x20 grid with 30% obstacles.

TABLE III: Comparison of Classical-A*, QAOA-A*, and CQRW-A* for obstacle density of 35% with grid size 20×20.

Algorithm	Actual Path Length	Time/s
Classical-A*	34.30	0.24
QAOA-A*	**27.45**	96.89
CQRW-A*	28.72	83.69

On a 20×20 grid, Table III presents a comparison of the performances of Classical-A*, QAOA-A*, and CQRW-A*. The grid contained 13 obstacles, covering 35% of the area. With QAOA-A* producing 27.45 units, CQRW-A* producing 28.72 units, and Classical-A* producing 34.30 units, both quantum-enhanced methods are able to find shorter pathways than the classical strategy. However, this comes at the expense of somewhat longer computation times: 96.89s for QAOA-A*, 83.69s for CQRW-A*, and as little as 0.24s for classical-A*.

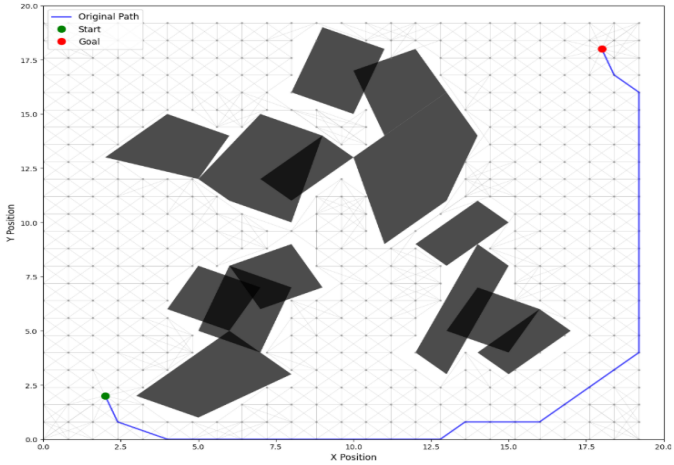

Fig. 11: Schematic diagram of path planning using 20×20 grid with 35% obstacle density for the classical A* algorithm.

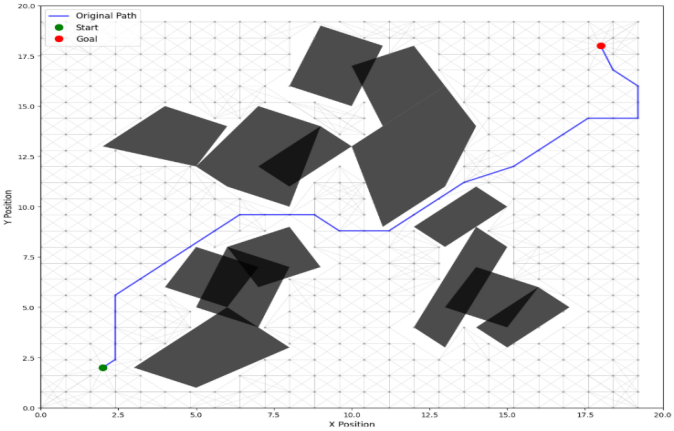

Fig. 12: Schematic diagram of path planning using 20×20 grid with 35% obstacle density for the CQRW-A* algorithm.

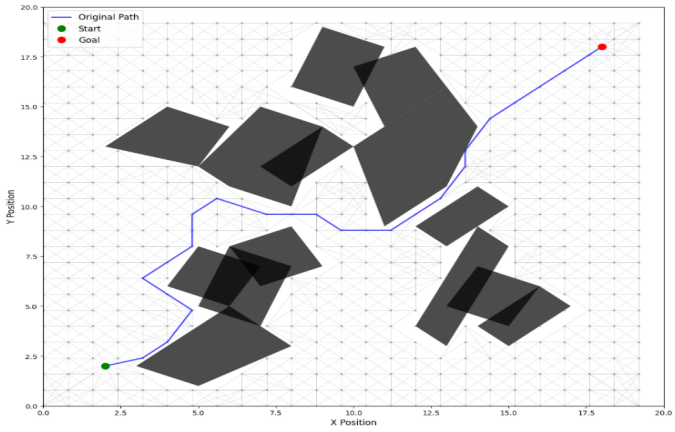

Fig. 13: Schematic diagram of path planning using 20×20 grid with 35% obstacle density for the QAOA-A* algorithm.

Using a grid of 20X20, with 35% obstacles, the route planning for this configuration is depicted in Figures 11, 12, and 13, respectively, for traditional A*, CQRW-A*, and QAOA-A*, respectively.

TABLE IV: Comparison of Classical-A*, QAOA-A*, and CQRW-A* for obstacle density of 17% with grid size 30×30.

Algorithm	Actual Path Length	Time/s
Classical-A*	43.66	0.28
QAOA-A*	41.78	178.02
CQRW-A*	**40.70**	197.60

In Table IV, for a 30×30 grid with 13 obstacles, which covers 17% of the grid, the quantum-enhanced methods outperform the classical method in terms of the path length. QAOA-A* reaches the second-shortest path at 41.78 units, after CQRW-A* at 40.70 units, and classical A* reaches 43.66 units. Even in less highly blocked situations, these results show that techniques influenced by quantum mechanics provide better path optimization. However, there is a substantial processing expense associated with the enhancement of the path quality. Classical-A* finishes in 0.28s, QAOA-A* takes 178.02s, and CQRW-A* takes 197.60s.

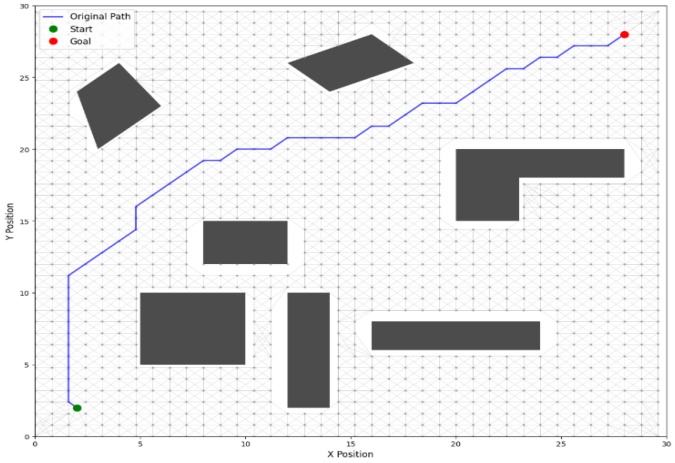

Fig. 14: Schematic diagram of path planning using 30×30 grid with 17% obstacle density for the classical A* algorithm.

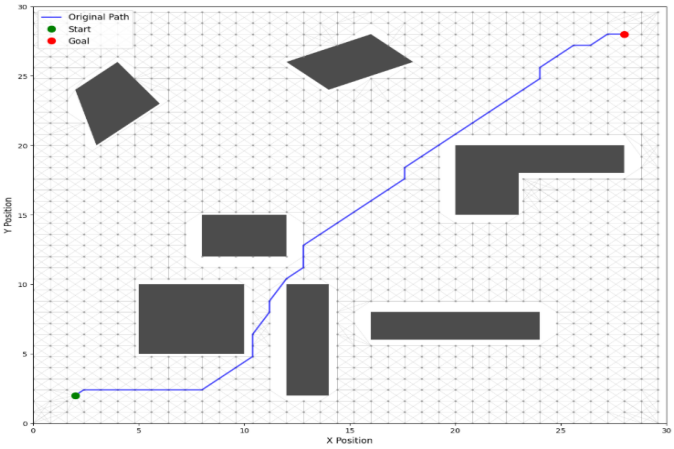

Fig. 15: Schematic diagram of path planning using 30×30 grid with 17% obstacle density for the CQRW-A* algorithm.

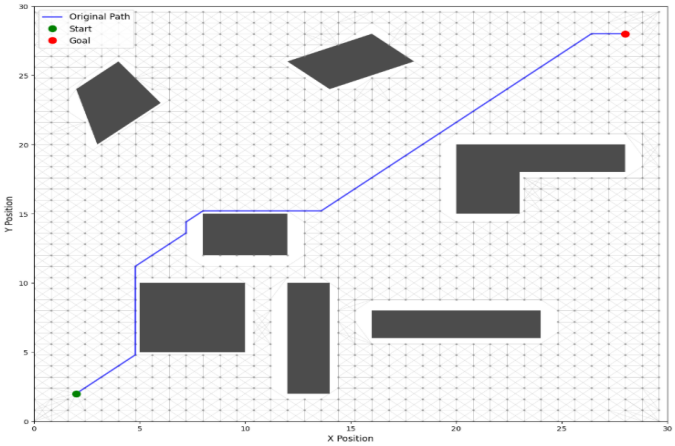

Fig. 16: Schematic diagram of path planning using 30×30 grid with 17% obstacle density for the QAOA-A* algorithm.

A 30x30 grid with 17% obstacles was used to illustrate the route planning for this configuration, as shown in Figures 14, 15, and 16 for classical A*, CQRW-A*, and QAOA-A*, respectively.

TABLE V: Comparison of Classical-A*, QAOA-A*, and CQRW-A* for obstacle density of 30% with grid size 30×30.

Algorithm	Actual Path Length	Time/s
Classical-A*	50.00	1.91
QAOA-A*	48.17	186.86
CQRW-A*	**46.29**	211.80

According to Table V, the quantum-enhanced algorithms provide better results than the classical A* algorithm in terms of path length on a 30×30 grid with 30% obstacle density. In contrast to the QAOA-A* and CQRW-A* algorithms, which produced shorter paths of 48.17 and 46.29 units, respectively, the classical A* algorithm generated paths with an average of 50.00 units. However, these improvements in path optimality are accompanied by significantly longer computational durations. The classical A* requires 1.91s, whereas the QAOA-

A* and CQRW A* approaches require 186.86s and 211.80s, respectively.

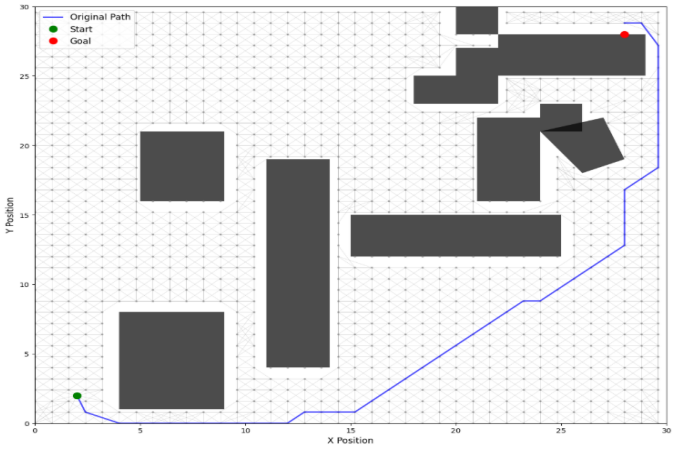

Fig. 17: Schematic diagram of path planning using 30×30 grid with 30% obstacle density for the classical A* algorithm.

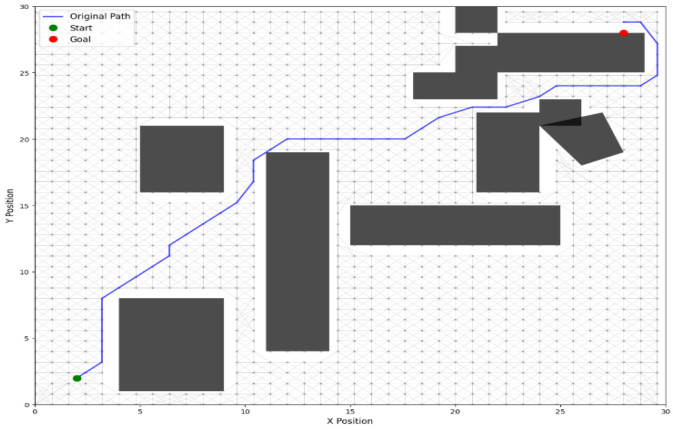

Fig. 18: Schematic diagram of path planning using 30×30 grid with 30% obstacle density for the CQRW-A* algorithm.

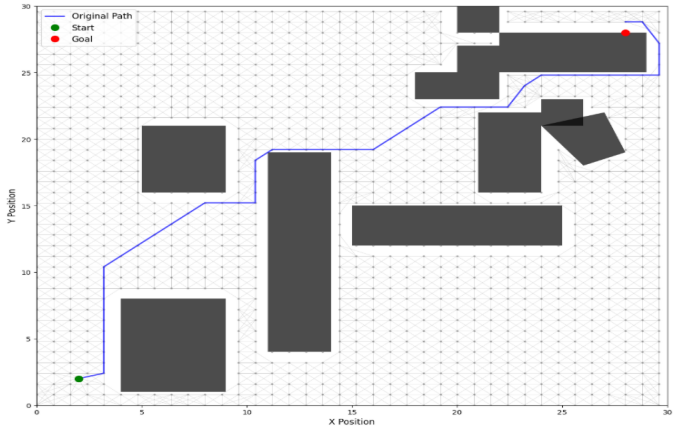

Fig. 19: Schematic diagram of path planning using 30×30 grid with 30% obstacle density for the QAOA-A* algorithm.

The route planning for this configuration is illustrated in Figures 17, 18, and 19 for classical A*, CQRW-A*, and QAOA-A*, respectively, on a 30x30 grid containing 30% obstacles.

TABLE VI: Comparison of Classical-A*, QAOA-A*, and CQRW-A* for obstacle density of 35% with grid size 30×30.

Algorithm	Actual Path Length	Time (s)
Classical-A*	48.18	0.92
QAOA-A*	42.10	169.69
CQRW-A*	**39.30**	515.52

In Table VI, the results of comparing the performance of the Classical A*, QAOA-A*, and CQRW-A* algorithms on a 30×30 grid with a 35% obstacle density are shown. The classical-A* algorithm produced a path length of 48.18 units and a computation time of 0.92s. The path length was reduced to 42.10s by QAOA-A*; however, the computation time increased to 169.69s. CQRW-A* had the longest computation time of 515.52s, but the shortest path length of 39.30s. The route plan for this configuration is illustrated in Figures 20, 21, and 22 for classical A*, CQRW-A*, and QAOA-A*, respectively, on a 30x30 grid containing 35% of obstacles.

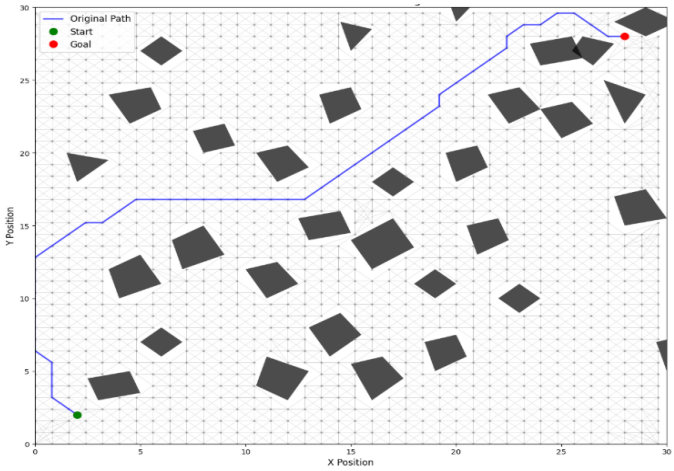

Fig. 20: Schematic diagram of path planning using 30×30 grid with 35% obstacle density for the classical A* algorithm.

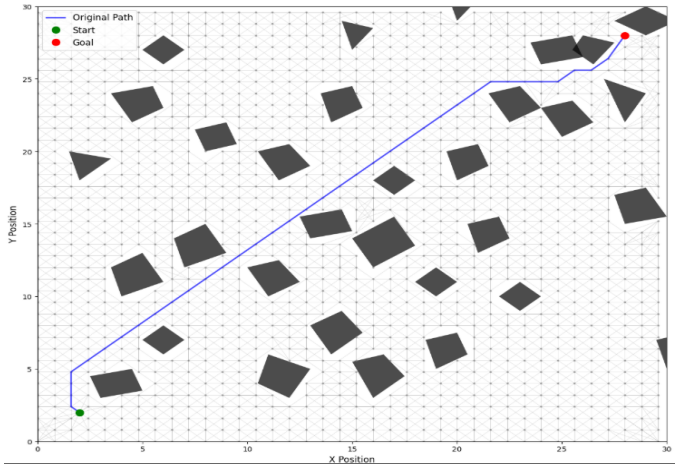

Fig. 21: Schematic diagram of path planning using 30×30 grid with 35% obstacle density for the CQRW-A* algorithm.

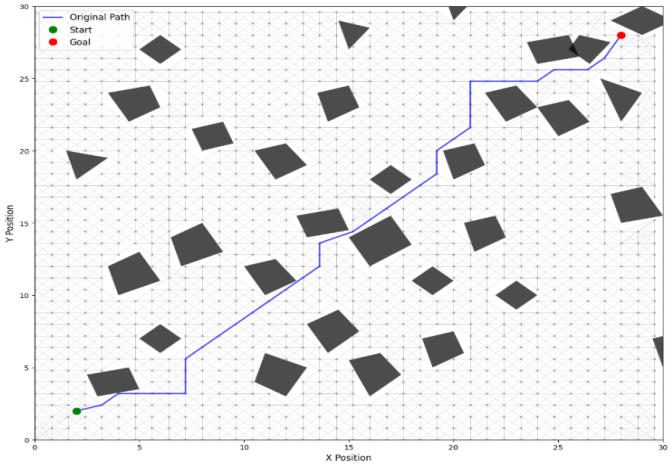

Fig. 22: Schematic diagram of path planning using 30×30 grid with 35% obstacle density for the QAOA-A* algorithm.

V. Conclusion and Future Work

This paper presents a comprehensive evaluation of hybrid quantum-classical algorithms for path planning and analyzes the performance of the Classical A*, QAOA-A*, and CQRW-A* algorithms. The experiments, conducted on grids of sizes 20×20 and 30×30 with obstacle densities of 17%, 30%, and 35%, demonstrate that quantum-inspired approaches consistently outperform classical A* in terms of path optimization. Specifically, for a 20×20 grid with 17% obstacle density, Classical A* generated a path length of 32.57 units in 0.073s. In comparison, QAOA-A* reduced the path length to 28.67 units, albeit at the cost of an increased computation time of 121.66s, while CQRW-A* achieved the shortest path length of 26.09 units in 65.74s. For more complex environments, such as a 30×30 grid with 35% obstacle density, Classical A* produced a path length of 48.18 units in 0.92s. QAOA-A* reduced the path length to 42.10 units, but the computation time increased to 169.69s, while CQRW-A* provided the shortest path at 39.30 units, albeit with the longest computation time of 515.52s. These results highlight the potential of quantum-enhanced algorithms to achieve superior path optimization, particularly in complex and obstacle-laden environments. However, they also underscore the trade-off between optimization quality and computational efficiency. Quantum-inspired algorithms such as QAOA-A* and CQRW-A* show promise in scenarios where pathfinding performance takes precedence over execution time, making them suitable for applications requiring high-quality solutions rather than real-time constraints.

Despite these advancements, the significant computational overhead associated with simulating quantum processes on classical hardware limits the scalability and real-time applicability of these algorithms in practical robotics scenarios. To address this, future research should focus on optimizing the quantum modules to reduce the computational burden, enhancing the simulation efficiency by refining the integration of quantum and classical techniques, and exploring real-world implementations on quantum hardware to achieve faster and more scalable solutions. Additionally, it is essential to investigate the integration of these quantum algorithms with autonomous decision-making systems and evaluate their performance in dynamic, real-world environments to advance their practical utility.

References

[1] L. Xu, K. Huang, J. Liu, D. Li, and Y. F. Chen, "Intelligent planning of fire evacuation routes using an improved ant colony optimization algorithm," *Journal of Building Engineering*, vol. 61, p. 105208, 2022.

[2] M. M. Costa and M. F. Silva, "A survey on path planning algorithms for mobile robots," in *2019 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*. IEEE, 2019, pp. 1–7.

[3] K. Karur, N. Sharma, C. Dharmatti, and J. E. Siegel, "A survey of path planning algorithms for mobile robots," *Vehicles*, vol. 3, no. 3, pp. 448–468, 2021.

[4] T. T. Mac, C. Copot, D. T. Tran, and R. De Keyser, "Heuristic approaches in robot path planning: A survey," *Robotics and Autonomous Systems*, vol. 86, pp. 13–28, 2016.

[5] A. M. Childs, "Universal computation by quantum walk," *Physical review letters*, vol. 102, no. 18, p. 180501, 2009.

[6] Q. Luo, H. Wang, Y. Zheng, and J. He, "Research on path planning of mobile robot based on improved ant colony algorithm," *Neural Computing and Applications*, vol. 32, pp. 1555–1566, 2020.

[7] X. Xu, X. Yu, Y. Zhao, C. Liu, and X. Wu, "Global path planning of mobile robot based on improved genetic algorithm," *Comput. Integr. Manuf. Syst*, vol. 28, pp. 1659–1672, 2022.

[8] C. Li, X. Huang, J. Ding, K. Song, and S. Lu, "Global path planning based on a bidirectional alternating search a* algorithm for mobile robots," *Computers & Industrial Engineering*, vol. 168, p. 108123, 2022.

[9] L. Weng, Z. Ji, M. Xia, and A. Wang, "Robot path planning based on improved multi-objective particle swarm," *Journal of System Simulation*, vol. 26, no. 12, pp. 2892–2898, 2014.

[10] M. Zha, Z. Wang, J. Feng, and X. Cao, "Unmanned vehicle route planning based on improved artificial potential field method," in *Journal of Physics: Conference Series*, vol. 1453, no. 1. IOP Publishing, 2020, p. 012059.

[11] F. Gul, I. Mir, L. Abualigah, P. Sumari, and A. Forestiero, "A consolidated review of path planning and optimization techniques: Technical perspectives and future directions," *Electronics*, vol. 10, no. 18, p. 2250, 2021.

[12] R. Chen, J. Hu, and W. Xu, "An rrt-dijkstra-based path planning strategy for autonomous vehicles," *Applied Sciences*, vol. 12, no. 23, p. 11982, 2022.

[13] S. Roy and Z. Zhang, "Route planning for automatic indoor driving of smart cars," in *2020 IEEE 7th International Conference on Industrial Engineering and Applications (ICIEA)*. IEEE, 2020, pp. 743–750.

[14] Y. Liu, C. Vogiatzis, R. Yoshida, and E. Morman, "Solving reward-collecting problems with uavs: A comparison of online optimization and q-learning," *Journal of Intelligent & Robotic Systems*, vol. 104, no. 2, p. 35, 2022.

[15] C. Hu, Y. Xia, and J. Zhang, "Adaptive operator quantum-behaved pigeon-inspired optimization algorithm with application to uav path planning," *Algorithms*, vol. 12, no. 1, p. 3, 2018.

[16] J. Li, B. Xu, Y. Yang, and H. Wu, "Quantum ant colony optimization algorithm for agvs path planning based on bloch coordinates of pheromones," *Natural Computing*, vol. 19, pp. 673–682, 2020.

[17] P. J. Karalekas, N. A. Tezak, E. C. Peterson, C. A. Ryan, M. P. Da Silva, and R. S. Smith, "A quantum-classical cloud platform optimized for variational hybrid algorithms," *Quantum Science and Technology*, vol. 5, no. 2, p. 024003, 2020.

[18] C. P. Williams and C. P. Williams, "Quantum gates," *Explorations in quantum computing*, pp. 51–122, 2011.

[19] J. Liu, Y. Cai, and Y. Cao, "A robot path-planning method based on an improved genetic algorithm," *Transactions of FAMENA*, vol. 48, no. 3, pp. 141–154, 2024.

[20] Y. Li and F. Liu, "Path planning algorithm for mobile robot based on improved ant colony algorithm," in *Journal of Physics: Conference Series*, vol. 2083, no. 4. IOP Publishing, 2021, p. 042033.

[21] H. Jiachen and F. Li-hui, "Robot path planning based on improved dung beetle optimizer algorithm," *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, vol. 46, no. 4, p. 235, 2024.

[22] R. Chen, J. Hu, and W. Xu, "An rrt-dijkstra-based path planning strategy for autonomous vehicles," *Applied Sciences*, vol. 12, no. 23, p. 11982, 2022.

[23] J. Lim and P. Tsiotras, "A generalized a* algorithm for finding globally optimal paths in weighted colored graphs," dec 2020.
[24] Y. Liu, P. Zhang, Y. Ru, D. Wu, S. Wang, N. Yin, F. Meng, and Z. Liu, "A scheduling route planning algorithm based on the dynamic genetic algorithm with ant colony binary iterative optimization for unmanned aerial vehicle spraying in multiple tea fields," *Frontiers in Plant Science*, vol. 13, p. 998962, 2022.
[25] X. Li, Q. Li, and J. Zhang, "Research on global path planning of unmanned vehicles based on improved ant colony algorithm in the complex road environment," *Measurement and Control*, vol. 55, no. 9-10, pp. 945–959, 2022.
[26] D. Khalidi, D. Gujarathi, and I. Saha, "T: A heuristic search based path planning algorithm for temporal logic specifications," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 8476–8482.
[27] P. U. Rao, F. Speelman, B. Sodhi, and S. Kinge, "A quantum computing approach for multi-robot coverage path planning," *arXiv preprint arXiv:2407.08767*, 2024.
[28] A. Chella, S. Gaglio, G. Pilato, F. Vella, and S. Zammuto, "A quantum planner for robot motion," *Mathematics*, vol. 10, no. 14, p. 2475, 2022.
[29] R. Maity, R. Mishra, and P. K. Pattnaik, "Analysis of path finding techniques for flying robots through intelligent decision-making algorithms in quantum inspired computing environment," *Wireless Personal Communications*, vol. 135, no. 3, pp. 1561–1580, 2024.
[30] X. Fan, J. Wang, H. Wang, L. Yang, and C. Xia, "Lqr trajectory tracking control of unmanned wheeled tractor based on improved quantum genetic algorithm," *Machines*, vol. 11, no. 1, p. 62, 2023.
[31] M. Mannone, V. Seidita, and A. Chella, "Modeling and designing a robotic swarm: A quantum computing approach," *Swarm and Evolutionary Computation*, vol. 79, p. 101297, 2023.
[32] M. J. Schuetz, J. K. Brubaker, H. Montagu, Y. van Dijk, J. Klepsch, P. Ross, A. Luckow, M. G. Resende, and H. G. Katzgraber, "Optimization of robot-trajectory planning with nature-inspired and hybrid quantum algorithms," *Physical Review Applied*, vol. 18, no. 5, p. 054045, 2022.
[33] M.-h. Jiao, H.-x. Wei, B.-w. Zhang, J.-q. Jin, Z.-q. Jia, and J.-l. Yan, "Path planning of escort robot based on improved quantum particle swarm optimization," in *2019 Chinese Control And Decision Conference (CCDC)*. IEEE, 2019, pp. 3730–3735.
[34] P. Atchade-Adelomou, G. Alonso-Linaje, J. Albo-Canals, and D. Casado-Fauli, "qrobot: A quantum computing approach in mobile robot order picking and batching problem solver optimization," *Algorithms*, vol. 14, no. 7, p. 194, 2021.
[35] Z. Du and H. Li, "Research on application of improved quantum optimization algorithm in path planning," *Applied Sciences*, vol. 14, no. 11, p. 4613, 2024.
[36] P. Lathrop, B. Boardman, and S. Martínez, "Quantum search approaches to sampling-based motion planning," *IEEE Access*, vol. 11, pp. 89 506–89 519, 2023.
[37] Y. Li, S. Qin, and L. Jing, "Research on flight trajectory optimization based on quantum genetic algorithm," in *Journal of Physics: Conference Series*, vol. 1549, no. 2. IOP Publishing, 2020, p. 022074.
[38] A. Chella, S. Gaglio, M. Mannone, G. Pilato, V. Seidita, F. Vella, and S. Zammuto, "Quantum planning for swarm robotics," *Robotics and Autonomous Systems*, vol. 161, p. 104362, 2023.
[39] S. E. Venegas-Andraca, "Quantum walks: a comprehensive review," *Quantum Information Processing*, vol. 11, no. 5, pp. 1015–1106, 2012.
[40] E. Farhi and S. Gutmann, "Quantum computation and decision trees," *Physical Review A*, vol. 58, no. 2, p. 915, 1998.
[41] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.