CQLite: Coverage-biased Q-Learning Lite for Efficient Multi-Robot Exploration

Ehsan Latif Ramviyas Parasuraman

***Abstract*— In practical applications, mobile robots must be able to autonomously navigate and explore complex surroundings. Prior approaches to autonomous exploration involved having the robots keep an internal map of the environment, which they would subsequently navigate using localization and planning techniques. However, these methods do not consider failures. Contrarily, learning-based approaches enhance exploration performance as the robot interacts with its environment. Still, they take longer to converge to the best solution, have high communication and update costs, and are therefore challenging to implement in practical settings due to their considerable sample complexity. In this paper, we propose a new cooperative Q-learning-based approach, termed CQLite, which implements a coverage-weighted reward function considering light communication overhead to address the need for fast convergence. We theoretically analyze the approach to guaranteeing full coverage, fast convergence, and reduced computational complexity. We also experimentally validate our method in simulated indoor map exploration using multiple robots. Our proposed CQLite significantly outperformed the state-of-the-art multi-robot exploration techniques (such as the Rapidly-exploring Random Trees (RRT) and the Deep Reinforcement Learning (DRL) based approaches) in terms of mapping performance (coverage and distance traveled) with more than 2x reductions in computation and communication.**

I. Introduction

Map-based exploration is a significant problem of interest in robotics and multi-robot systems (MRS) [1], [2]. In this problem, robots move continuously to create the full environmental map in a new bounded environment without any prior information. It can be useful in various applications, including search and rescue, domestic service, survey and operations, field robotics, etc. The autonomous exploration solutions can also demonstrate how adaptable the robotic system or MRS is since robots could carry out the exploration and mapping mission in a different and uncharted area.

Some recent work, like frontier-based [3] and information-based [4] methods, have been effective in realizing an efficient exploration objective. Information-based methods typically use the Shannon entropy to describe the uncertainty of the environmental map and construct the optimization problems such that the robot's control variable is continuously optimized during the exploration process. Frontier-based methods involve deciding the robot's next move (or path) by searching the frontier points on the border of free and unknown points. Often, these methods only produce approximate solutions due to the nature of optimization.

Integrating learning with planning solutions holds great promise, especially for robot exploration [5]. In the reinforcement learning (RL) paradigm, robots can continuously improve competence and adapt to the dynamics of natural surroundings by observing the results of navigational choices made in the actual world [6]. On the other hand, cooperation among robots in an MRS can help achieve a complex mission through simple distributed approaches [7].

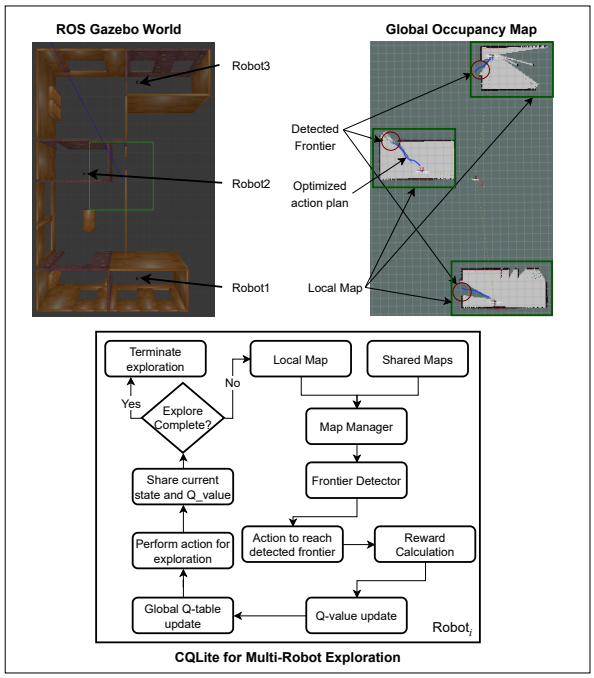

Fig. 1. Overview of the proposed CQLite for efficient multi-robot exploration shown with the Gazebo simulation and the mapping process.

This paper explores the intersection between learning and cooperation, designs a combined solution to achieve efficient map exploration, and provides theoretical support for coverage control, fast convergence, and time complexity. We leverage the benefits of learning-based paradigms for joint exploration. We aim to create a distributed algorithm that gains knowledge through robot-robot information sharing while minimizing communication and computing overheads. Specifically, we have utilized a cooperative Q-learning methodology with a coverage-biased reward function with a light-communication and information fusion strategy. In our approach, we reduce communication complexity by sharing only the current state information, i.e., Q-value, instead of the complete Q-table as done in [8], [9], [10], [11]. Fig. 1 provides an overview of the proposed method implemented in the Robot Operating System (ROS [12]) framework.

The main contributions of our work are the following:

- We introduce a novel coverage-biased Q-learning ap-

Heterogeneous Robotics Research Lab, School of Computing, University of Georgia, Athens, GA 30602, USA
Author emails: {ehsan.latif;ramviyas}@uga.edu

proach (termed as *CQLite*) for efficient multi-robot map exploration with limited data exchanges.
- We substantiate our method with theoretical analysis of convergence and computational complexity.
- We validate through simulation experiments and performance comparison with state-of-the-art multi-robot exploration methods: RRT [13] and DRL [14].
- We open source[1] the CQLite as a ROS package for use and further development by the robotics community.

The key idea behind the CQLite is utilizing a coverage-biased reward function and achievement of fast convergence by sharing limited information among robots. Our method achieves fast convergence with the highest coverage performance and reduced communication costs. A real-world robot demonstration of CQLite is available in the attached video[2].

II. Related Work

The two most common approaches to the map exploration problem focus on frontiers and information. In a recent study, the agent was greedily pushed in an occupancy grip-map to the closest boundaries [15] or the most uncertain regions [16]. For instance, Tang et al. [17] adopted an information potential field to indicate the joint path and map entropy; then, based on the gradient of this field, an exploration path was efficiently computed. Walker et al. [18] proposed a DRL, wherein a utility function was formalized as a criterion to measure the information gain of each candidate waypoint.

Most learning-based autonomous navigation approaches rely mainly on supervised training data. Examples include learning route planners, obstacle detectors, reactive controllers over a map-based planner [19], and driving affordances [20]. However, many approaches are based on human supervision, which is fundamentally bound by the amount of human data available. The capabilities of intense and expressive models like deep neural networks are frequently constrained significantly by the data available.

Researchers exploited self-supervised learning with generalized graph computation to achieve autonomous map exploration. Gregory et al.[21] provided a generalized computation graph that includes value-based and model-based techniques, with specific instantiations interpolating between model-free and model-based, to address the need to learn complicated policies with few samples. They then created an instance of this network to create a sample-efficient navigation model that can be taught from unprocessed photos. In the case of a complex environment, graph computation can become a complicated problem to solve and is not feasible for real-time exploration with resource-constrained robots.

In research [22], Q-learning is expanded, and an RL for multi-robot cooperative tasks is proposed. The standard reward is transformed into an "internal reward" when robots on the map cannot communicate with one another. This allows the robots to learn following this "internal reward" type and achieve task collaboration under such challenging circumstances. Another study [23] suggests that robots can use the knowledge acquired through learning more effectively by mixing RL with traditional deep neural networks. Compared to classical RL, the algorithm can help the robot find the best way to get out of the map more quickly.

A unique RL map navigation technique is proposed in research [24] to address the unknown map navigation problem for autonomous ground vehicles. First, the picture data of the random map is collected using the quadrotor's bottom-mounted camera. Then the virtual map is created in the simulation environment using an image processing technique. An enhanced Q-learning algorithm is suggested to address the issue that the original greedy strategy repeatedly forces the robot to linger in the past state.

A recent study [25] looked at the issue of automatic exploration in an unfamiliar setting, which is crucial when using a robotic system to perform specific social duties. By breaking down the exploration process into the decision, planning, and mapping modules, research created a general exploration framework that increases the adaptability of the robotic system. They proposed a DRL-based decision algorithm based on that framework, which uses a deep neural network to learn an exploration strategy from a partial map. Similarly, for exploration region selection in the decision-making process, a study [26] used DRL. The agent can forecast a long-term visitation order for unknown subregions thanks to the presentation of exploration knowledge over office designs. Based on this method, they suggest an exploration architecture that, to enhance exploration performance further, combines a DRL model, a next-best-view selection strategy, and a structural integrity measurement.

Though the DRL and Q-learning approaches are popular and solve specific research questions, they provide sub-optimal solutions outside the task space (or domain). They cannot converge in cases of an infinite horizon. DRL approaches address traditional map exploration issues but provides solutions with high computation, communication, and update costs. When it comes to RL, to solve the problems of slow learning speed and non-convergence of traditional Q-learning, many scholars have improved RL in different aspects. Although some methods cannot accelerate the convergence speed of Q-learning, they provide us with interesting ideas. To address these gaps in the literature, our approach considers an efficient information transfer mechanism combined with Q-learning for multi-robot cooperation to solve map exploration tasks.

III. Proposed Approach

The problem formulation employed here deploys many robots at random starting locations in an unknown environment. The robots must navigate towards the frontier position detected by local sensing information as a standard map exploration strategy [15]. To accomplish this efficiently, a robot must decide which frontier to navigate after leaving its current explored region. In doing so, it is hoped to reduce the number of steps to take and the size of data exchanges with

[1] https://github.com/herolab-uga/cqlite

[2] Video of simulation experiments and real-world demonstrations are also available at https://hero.uga.edu/research/cqlite/

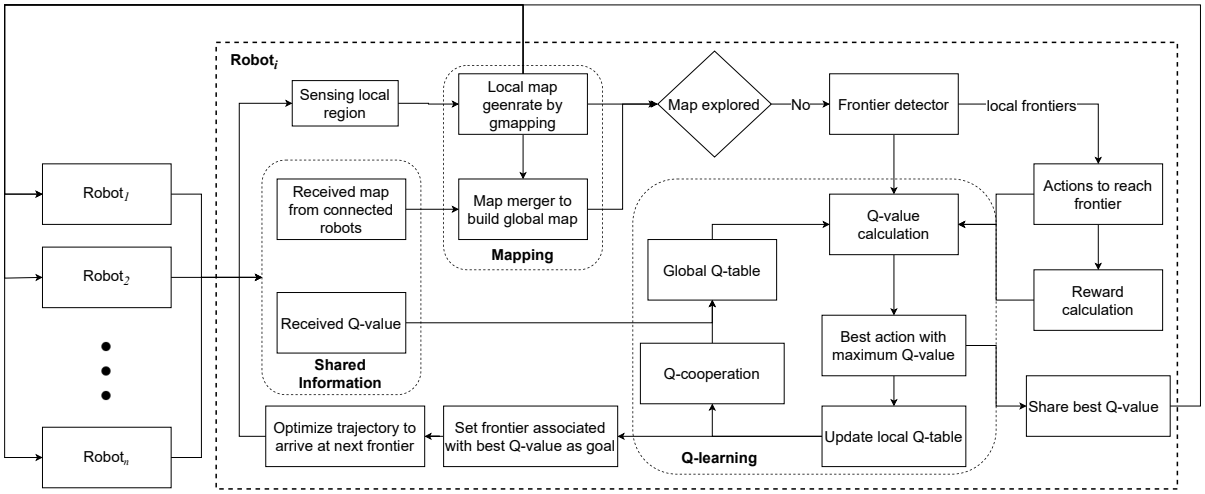

Fig. 2. System architecture of CQLite distributed across several robots. It shows the Robot$i$'s process showing the mapping, frontier detection, and Q-learning operations along with the communication of local map and updated Q-value information to n connected robots.

connected robots while considerably enhancing the effectiveness of each robot's random exploration. Robots continue to generate local maps and share the newly developed map after arriving at a new frontier. Each robot merges a locally generated map and receives a map from peers to build a global map using the feature similarity-based map merging technique [27]. The robot's decisions regarding an action plan are based on the shared information and Q-learning computation, as discussed below.

A. Q-learning

Markov decision processes frequently model the robot's interaction with the environment. A robot can transition from state $s_t \in S$ to state s_{t+1} due to carrying out an action $a_t \in A$ based on its state at time t. The transition probability is defined as $T : S \times A \times S \rightarrow [0, 1]$. The robot will receive a reward for each action using a reward function $R : S \times A \times S \rightarrow R$ specific to the task. The robot will have learned the course of action to take in each state and will be able to maximize the reward of the entire interaction process.

In Q-learning, all possible states and actions are created using the Q table, which then updates each value through iterative learning. The robot then chooses the best course of action for each state based on the values in the table. This approach is frequently utilized in path planning, chess, card games, and other activities.

Assumptions: The environment has a flat ground terrain throughout all available space to perform the task.

- The robots have an omnidirectional sensory system that can detect the boundary of an obstruction within the maximum sensing range r_s and provides a description of the open space around the robot.
- Each robot has a communication range $r_c >> r_s$ that it can use at any time to broadcast the data stored in its memory. The robot is constantly available to receive information about its relative position from a neighbor robot inside the r_c communication range.

The Q-learning algorithm updates the Q value as

$$Q_{t+1}(s_t, a_t) = (1-\alpha)Q_t(s_t, a_t) + \alpha\left[r_t + \gamma \max Q_t(s_{t+1}, a_t)\right] \quad (1)$$

where s_t and s_{t+1} are the current states and the next state, respectively. It is the action executed, $\alpha \in (0, 1]$ controls the balance between the coverage and delay, and γ is the discount factor to prioritize present vs. future rewards.

The objective of the CQLite is to perform maximum coverage in less time and avoid overlapping exploration, which can be numerically defined as

$$max_\pi\{P_a^\pi(t) - \lambda E_t(a|\pi)\}, \quad (2)$$

where $P_a^\pi(t)$ is the probability to cover the region using for action a using policy π at time t, $E_t(a|\pi)$ is average number of steps to cover associated with action a at time t in policy π and λ is the cost associated with each step.

At each discrete time step t, the agent acquires an observation s_t from the environment, selects a corresponding action a_t, then receives feedback from the environment in the form of a reward $r_t = R(s_t, a_t)$ as shown below

$$r_t = \begin{cases} -\lambda & \text{if } s_t \text{ with } a_t \text{ unexplored} \\ \lambda - Q_t & \text{if } s_t \text{ with } a_t \text{ explored} \end{cases} \quad (3)$$

Then the state information is updated s_{t+1}. The goal of RL agent is to select policy π that maximizes the discounted sum of future rewards, i.e., $Q_\pi(s_1) = \sum_\tau^{t=1} \gamma^t R(s_t, a_t)$, which according to the Bellman optimality principle satisfies.

This reward function in Eq. (3) produces a negative reward whenever the agent has looped back and no reward otherwise.

B. Multi-Robot Lite Cooperation

The next issue we address is the communication amongst individual exploration-capable robots in our MRS. CQLite exploration system is designed for a single robot, allowing each robot to make independent decisions based on local information and with little interaction from other robots. In

Algorithm 1: CQLite Algorithm

Data: Reward matrix R, learning rate α, discount factor γ, step cost λ;
$Q(s,a) = 0$, Number of iterations: $k = 0$;
Global Q table as Q_{cop};
while $(k \leq k_{max})$ *and (Q table still changes)* **do**
  $S_t \rightarrow S_{start}$, step = 0;
  Local Q table as Q ;
  while $(step \leq step_{max})$ *and* $(s \neq terminal)$ **do**
    Receive Q-value for S_t from other robots;
    Update Q_{cop} and Q;
    Initialize empty frontier list F_t;
    Q_{temp} = list(0);
    for *each frontier* f *in* F_t **do**
      calculate Q_f for actions a_f to reach frontier f using Eq. (1);
      Append Q_{temp} with Q_f for action a_f;
    end
    Update Q_{t+1} with $max_a(Q_{temp})$ Take action a_t associated with Q_{t+1};
    Share Q_{t+1} with other robots;
    $S_t = S_{t+1}$;
    Find new frontiers at new S_t and update F_t;
    step = step + 1;
  end
  k = k + 1;
end

our lite version of Q-learning, only the current state and Q-value are communicated amongst nearby robots to encourage cooperation. When another robot receives the information, it will update the received Q value in its Q table and update the local map. We develop a discovery approach based on the distance between simulated robots to replicate the network range in which we only share the current position of a robot i, its Q-Value for each direction, and mark the current situation as explored to avoid repetitive exploration.

C. Overall Procedure

In the traditional Q-learning process, the state action space is too large due to the complex map environment used to solve map-aided problems. This also results in an unmanageably high number of iterations and takes a long time in the entire learning process. To avoid this issue, we propose a multi-robot (lite) cooperation strategy based on Q-learning. The whole procedure concerning robot i can be visualized in Fig. 2. The precise stages in this procedure are

- Given a sensing range of a robot, it is only allowed to perform Q-learning on the visible frontiers in the explorable region.
- Multiple robots use Q-learning to explore these small maps at the same time. Because the sub-map is much simpler than the target map, it only needs a few iterations to complete the learning.
- Each robot updates its Q table at every iteration based on Q-value update equation 1. After learning, we can get a set of Q tables, where $Q = \{Q_1, Q_2, Q_3, ..., Qn\}$.
- Combine these Q tables as the global Q table of the global map, called Q_{cop}.
- Finally, based on Q_{cop}, the robot efficiently completes the target map's exploration.

Algorithm 1 provides the pseudo-code description of CQLite for efficient map exploration.

D. Coverage Control

Robots create a global Q table for each cell and action after searching the map and experiencing several experiences. The Q table is then turned into a weighted graph $G = (S, E, C)$, where $S = \{s_1, s_2, ..., s_n\}$ denotes the set of states, and $E \in |S| \times |S|$ signifies the set of edges whose elements indicate whether or not a path exists between the center points of each pair of states. Furthermore, C is the weight matrix indicating the edge metric cost. The primary goal of discovering this study's reduced graph and significant states is to optimally disperse robots over the coverage region by minimizing the relevant cost function. Because robots move at varying speeds, we formulate the cost as a function of the defined traveling time as $t_{(s_{p_i}, s_q)} = \frac{d_{(s_{p_i}, s_q)}}{v_i}$, where v_i is the i^{th} robot's speed, and $d_{(s_{p_i}, s_q)} \in C$ is the Euclidean distance between the i^{th} robot's current state p_i and state q. Furthermore, knowing the optimal path from state p_i to state s, the overall optimal traveling time for each robot is the sum of the trip times (costs) from state p_i to state s. In this study, the shortest path between each pair of states is computed using the A* algorithm [28]. Then the total time τ is calculated by knowing path $P = \{p_1, p_2, ..., p_n\}$ as:

$$\tau_{(S_p, S_q)} = t_{(s_{p_1}, s_q)} + t_{(s_{p_2}, s_q)} + ... + t_{(s_{p_n}, s_q)} \quad (4)$$

After determining the shortest time between each pair of states, the field is partitioned into M Voronoi subgraphs g_{r_i} for $i \in \{1, 2, ..., M\}$ to distribute work proportionally among M robots. To that aim, the ideal Voronoi diagram g_{r_i} for i^th robot, according to Lloyd's algorithm [29], is a split of the area determined as

$$g_{r_i} = \{s_q \in S | \tau_{(s_{p_i}, s_q)} \leq t_{(s_{p_j}, s_q)}, \forall i \neq j\}. \quad (5)$$

The i^{th} robot is responsible for covering the state s (associated robot) in its sub-graph g_{r_i} using the Voronoi partitioning result. The entire cost is then calculated as

$$\Lambda_{(p, g_r)} = \sum_{i=1}^{m} \sum_{q \in g_{r_i}} \tau_{(s_{p_i}, s_q)} \phi_q, \quad (6)$$

where ϕ_q is the priority value associated with state s_q. As the map turns into a graph, higher priority values are assigned to target states, while lower priority values are assigned to states far and already explored from the current state. The entire travel time (cost) will therefore be minimized, and an optimal solution will be obtained only when the distance between the current state of the vehicle and the state $d_{(s_{p_i}, s_q)}$ converges to zero.

IV. Theoretical Analysis

We first analyze the convergence of the target Q value update function Eq. (1). We denote the error ratio $\delta_t = \frac{MSE(Q_t)}{E_t(a|\pi)}$.

Theorem 1 (Convergence). Using Eq. (1) for Q-value updates, then if $0 \leq \delta_t \leq 1$, with probability $1-e$, we have:

$$E_t(a|\pi) \leq \alpha E_1(a|\pi) + \sqrt{\frac{\ln(1/e)\sum_{i=0}^{t-1}\psi_i^2(\delta_{t-i:t})}{2}} \quad (7)$$

Here, $\psi_i(\delta_{t-1:t}) = \frac{\prod_{j=t-i}^{t-1}(j+\gamma\delta_j)}{\prod_{j=t-i}^{t} j}$ and $\alpha_t = \frac{\prod_{j=1}^{t-1}(j+\gamma\delta_j)}{\prod_{j=2}^{t} j}$.

Proof: our analysis is derived based on the subsequent synchronous Q-learning. In contrast to the conventional synchronous Q-learning, we swap out the current Qt for the independent Q-function $Q'(s,a)$ for the target $Q_t(s_t,a_t)$.

$$\begin{aligned} &Q_t\left(s_t,a_t\right) \\ &= \left(\frac{t-1}{t}\right) Q_{t-1}(s,a) + \frac{1}{t}\left(r_t + \gamma max_{a'} Q'_{t-1}(s',a')\right) \end{aligned}$$

Note that if $Q'_t(s,a) = Q^*_{source}$, we know that $0 \leq \delta_t \leq 1$. First, we break down the update role into:

$$\begin{aligned} &Q_t\left(s_t,a_t\right) \\ &= \left(\frac{t-1}{t}\right) Q_{t-1}(s,a) + \frac{1}{t}\left(r_t + \gamma max_{a'} Q'_{t-1}(s',a')\right) \\ &= \left(\frac{t-1}{t}\right) Q_{t-1}(s,a) + \frac{1}{t}\big(r_t + \gamma max_{a'} Q'_{t-1}(s',a') \\ &+ \gamma max_{a'} Q^*_{t-1}(s',a') - \gamma max_{a'} Q^*_{t-1}(s',a')\big) \end{aligned}$$

Let $\varepsilon_t(s,a) = Q_t(s,a) - Q^*(s,a)$ and $\xi(s') = \gamma \times max_{a'}\left(Q^*_{t-1}(s',a')\right)$ then recall the definition of δ_t, we will have:

$$\begin{aligned} &\varepsilon_t(s,a) \\ &\leq \frac{t-1}{t}\varepsilon_{t-1}(s,a) + \frac{1}{t}\big(\xi(s') - E_{s'}\xi(s')\big) + \frac{1}{t}\gamma\delta_t\varepsilon_{t-1}(s',a') \\ &\leq \frac{t-1}{t}\varepsilon_{t-1}(s,a) + \frac{1}{t}\big(\xi(s') - E_{s'}\xi(s')\big) + \frac{1}{t}\gamma\delta_t E_{t-1} \end{aligned}$$

As we know $\varepsilon_t(s,a) \leq E_t$, by applying maximization and recursion of E, we will have:

$$\begin{aligned} E_t &\leq \frac{t-1+\gamma\delta_t}{t}E_{t-1} + \frac{1}{t}\big(\xi(s') - E_{s'}\xi(s')\big) \\ &\leq \frac{\prod_{j=1}^{t-1}(j+\gamma\delta_j)}{\prod_{j=2}^{t} j}E_1 + \sum_{i=1}^{t-1}\frac{\prod_{j=t-i}^{t-1}(j+\gamma\delta_j)}{\prod_{j=t-i}^{t} j} \\ &\times \big(\xi(s') - E_{s'}\xi(s')\big) \\ &= \alpha_t E_1 + \sum_{i=1}^{t-1}\psi_i(\delta)\big(\xi(s') - E_{s'}\xi(s')\big) \end{aligned}$$

According to weighted Hoeffding inequality [30], with probability $1-e$, we can prove Eq (7) for theorem 1. ■

The convergence result demonstrates the influence of the error ratio on the convergence rate. In other words, learning will go more quickly for our chosen Q value update function.

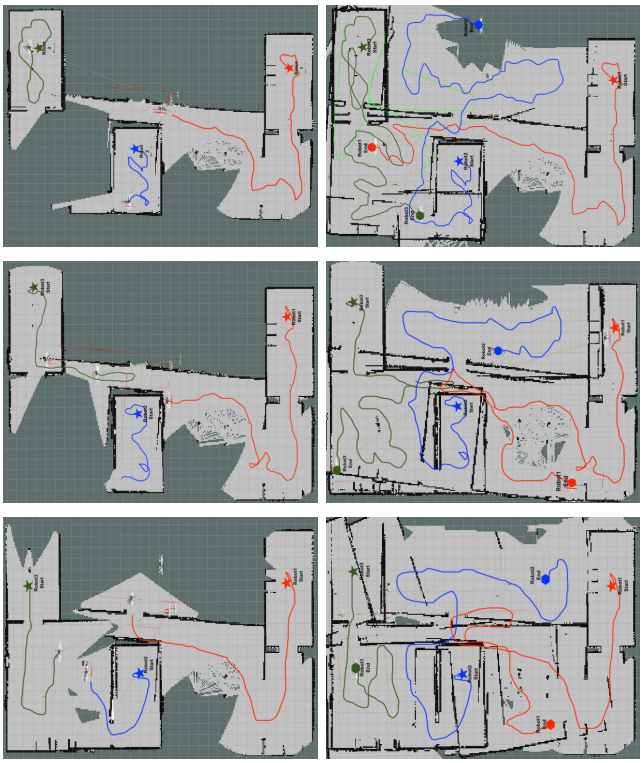

Fig. 3. A depiction of the outcome in a sample trial. It shows the partial map generated at 150s elapsed time (left column) and the final explored map (right column) created by the three compared approaches; RRT (top), DRL (center), and CQLite (bottom), with three robots moving in a simulated Gazebo world along with the following trajectories, start and end points.

A. Time Complexity

Assume that the grid factor is k_g and that the target submap is $k \times l$ in size. The grid map's size is then $k_g k \times k_g l$, and the total number of points is $k_g^2 kl$. The operations to find Q values must be carried out cyclically $k_g^2 kl$ times. kl can calculate and represent the CQLite's state space size. M_i is used in the computation as a history of the agents' positions, and the size of the information map and the number of agents determine the state's capacity. After all the states have been trained, the best policy can be found by choosing the action space with the highest Q-value. This is done by querying a list of length (kl) to determine the Q-value of the current state. As a result, searching a list and locating the highest Q-value in the action space constitute the computational complexity of the Q-Traversal algorithm. In actuality, both full search and list query comprise various comparison processes. The list length and the comparison data's nature influence how many comparisons are made. The length of the Q-value table is significantly less than (kl) through selecting the training process, and the computing complexity of the algorithm is considerably less than $O(kl)$.

V. Experimental Validation

Three Turtlebot3 robots are used to carry out the exploration plan, implemented using the ROS framework. The open-source *openslam-gmapping*[3] technique of the ROS

[3] https://openslam-org.github.io/gmapping.html

TABLE I

PERFORMANCE RESULTS OF RRT, DRL, AND THE PROPOSED CQLITE.

* INDICATES THE BEST PERFORMER.

Approaches	RRT [13]	DRL [14]	CQLite (ours)
Mapping Time (s)	1208 ± 52	**624*** ± 67	1029 ± 59
Total distance (m)	592 ± 11	604 ± 19	**543*** ± 9
Mapping Efficiency(%)	87 ± 4	91 ± 3	**95*** ± 3
CPU Utilization (%)	112 ± 22	79 ± 18	**42*** ± 8
RAM (MB)	824 ± 19	1264 ± 41	**665*** ± 24
Communication Payload (MB)	2.2 ± 0.08	1.4 ± 0.06	**0.6*** ± 0.01
Time Complexity Per Robot	$O(n \times \log(n))$ n-size of map,	$O(n)$ n-size of map,	$O(kl)$* k-width of sub-map l-height of sub-map

gmapping package is used to create 2D maps. It uses odometer data and a particle filter method as its foundation. The local maps created by each robot are combined to create the global map. Frame conversion between the local map frames is necessary for map merger. The coordinates transformation correlation between the robots must be calibrated before combining local maps. In the current work, the global frame is one robot's frame, and the relative positions and orientations of the robots are initialized to a known state.

The ROS *movebase*[4] package allows the robot to move toward the goal point while securely avoiding barriers between robots. The Dijkstra algorithm for global path planning and the Dynamic Window Approach (DWA) for local dynamic obstacle avoidance are both implemented in this package. It should be noted that in this study, the units of time and distance are in seconds and meters, respectively.

A. Evaluation

The exploration approach and the method put forward by RRT [13] and DRL [14] are compared in our experiments. We use the below metrics for evaluation: 1) **Mapping Time**: The amount of time spent mapping is a gauge of how effectively a multi-robot map is explored; 2) **Total Distance Traveled:** This term refers to the length of the robot's trajectory as a whole, which is necessary for a multi-robot system to explore the entire map. The entire trajectory length gives an idea of the robot's energy usage while subtly describing its investigation's effectiveness; 3) **Mapping Efficiency:** The percentage of the generated map with time elapsed; 4) **CPU Utilization:** The maximum percentage consumption of the processor of a robot throughout the trajectory; 5) **Memory Consumption (RAM):** The maximum occupied memory by the robot throughout the trajectory; 6) **Communication payload:** The maximum size of shared data by a robot.

B. Simulation

The simulation environment is created using the Gazebo simulator. Gazebo offers a robust collection of sensor models that may be utilized immediately, including camera, depth camera, laser, IMU, and other sensors frequently used by robots. A closed indoor environment with about $250m^2$ area has been established on the gazebo simulator with the house world. The robots may quickly finish the map investigation in a closed environment. Each robot has a laser scanner to gather data about its surroundings. The robot's trajectory length can be determined based on the odometer information.

The following parameters are used in the experiments in the simulated environment. The laser scanner's range and r_{area} are set to $15m$ and $1m$, respectively. Additionally, the robot's maximum linear and angular speeds in the simulation are set to $0.5ms^{-1}$ and $\frac{\pi}{4}rads^{-1}$, respectively. Additionally, the global detector's growth factor η and the local detector's growth factor η_1 in the RRT detector are set to $5m$ and $3m$, respectively. The weight parameters, $\alpha = 0.6$, $\gamma = 0.95$ and $\lambda = 2$ for $1m$ distant step.

C. Results

We have reported the average performance of each approach after performing multiple trials to reduce the measurement noise. In a three-robot system, numerous simulations are carried out. The mapping outcome of each approach with the trajectories followed by three robots in the simulated environment is shown in Fig. 3. It should be stated that the outcome considers the average mapping time, distance traveled, and mapping efficiency. Mapping efficiency is determined by comparing with the original map, and reported percentages are normalized with gazebo world dimensions. Table I compares the three methods in terms of different evaluation metrics used. It also lists the theoretical (algorithmic) computational complexity. In the proposed strategy, CQLite can cover the largest possible area in less time than DRL and comparable to RRT. However, CQLite can achieve 10% higher mapping efficiency by traveling 22m less than RRT. It is clear that CEQlite consumes approximately half of the memory and has less than half the communication payload size as DQL. However, the memory consumption of CQLite is still comparable with RRT, as the RRT approach applies pruning techniques to reduce memory consumption.

Overall, the proposed CQLite shows significant potential to outperform the state-of-the-art techniques and creates promising avenues to further research. It applies well to resource-limited and communication-limited applications.

VI. CONCLUSION

Learning-based strategies offer promising solutions to multi-robot exploration problems. However, because of their substantial sample complexity, they take longer to converge to the optimal solution, incur high communication and update costs, and are consequently difficult to apply in real-world situations. To satisfy the demand for quick convergence, we devise CQLite, a cooperative Q-learning over a coverage-weighted reward function with little communication overhead. We evaluate the strategy using empirical data to ensure complete coverage, quick convergence, and minimal computational cost. Additionally, we compared our method to widely-used RRT and DRL strategies for multi-robot map exploration. These methods included simulation and real-world indoor exploration with several robots. Our future work includes validating the scalability of the approach and increasing the generality of reward for a heterogeneous team of multi-robot systems.

[4] https://github.com/ros-planning/navigation

REFERENCES

[1] W. Burgard, M. Moors, C. Stachniss, and F. E. Schneider, "Coordinated multi-robot exploration," *IEEE Transactions on robotics*, vol. 21, no. 3, pp. 376–386, 2005.

[2] Y. Rizk, M. Awad, and E. W. Tunstel, "Cooperative heterogeneous multi-robot systems: A survey," *ACM Computing Surveys (CSUR)*, vol. 52, no. 2, pp. 1–31, 2019.

[3] A. Dai, S. Papatheodorou, N. Funk, D. Tzoumanikas, and S. Leutenegger, "Fast frontier-based information-driven autonomous exploration with an mav," in *2020 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2020, pp. 9570–9576.

[4] B. Fang, J. Ding, and Z. Wang, "Autonomous robotic exploration based on frontier point optimization and multistep path planning," *IEEE Access*, vol. 7, pp. 46 104–46 113, 2019.

[5] R. Shrestha, F.-P. Tian, W. Feng, P. Tan, and R. Vaughan, "Learned map prediction for enhanced mobile robot exploration," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 1197–1204.

[6] B. Singh, R. Kumar, and V. P. Singh, "Reinforcement learning in robotic applications: a comprehensive survey," *Artificial Intelligence Review*, pp. 1–46, 2021.

[7] R. Fierro, L. Chaimowicz, and V. Kumar, "Multi-robot cooperation," in *Autonomous Mobile Robots*. CRC Press, 2018, pp. 417–460.

[8] A. K. Sadhu and A. Konar, "An efficient computing of correlated equilibrium for cooperative q-learning-based multi-robot planning," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 50, no. 8, pp. 2779–2794, 2018.

[9] J. Wang, Z. Ren, T. Liu, Y. Yu, and C. Zhang, "Qplex: Duplex dueling multi-agent q-learning," *arXiv preprint arXiv:2008.01062*, 2020.

[10] H. Ge, Y. Song, C. Wu, J. Ren, and G. Tan, "Cooperative deep q-learning with q-value transfer for multi-intersection signal control," *IEEE Access*, vol. 7, pp. 40 797–40 809, 2019.

[11] A. Mahajan, T. Rashid, M. Samvelyan, and S. Whiteson, "Maven: Multi-agent variational exploration," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[12] M. Quigley, B. Gerkey, and W. D. Smart, *Programming Robots with ROS: a practical introduction to the Robot Operating System*. " O'Reilly Media, Inc.", 2015.

[13] L. Zhang, Z. Lin, J. Wang, and B. He, "Rapidly-exploring random trees multi-robot map exploration under optimization framework," *Robotics and Autonomous Systems*, vol. 131, p. 103565, 2020.

[14] J. Hu, H. Niu, J. Carrasco, B. Lennox, and F. Arvin, "Voronoi-based multi-robot autonomous exploration in unknown environments via deep reinforcement learning," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 12, pp. 14 413–14 423, 2020.

[15] W. Gao, M. Booker, A. Adiwahono, M. Yuan, J. Wang, and Y. W. Yun, "An improved frontier-based approach for autonomous exploration," in *2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*. IEEE, 2018, pp. 292–297.

[16] A. Bouman, M. F. Ginting, N. Alatur, M. Palieri, D. D. Fan, T. Touma, T. Pailevanian, S.-K. Kim, K. Otsu, J. Burdick *et al.*, "Autonomous spot: Long-range autonomous exploration of extreme environments with legged locomotion," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 2518–2525.

[17] C. Tang, R. Sun, S. Yu, L. Chen, and J. Zheng, "Autonomous indoor mobile robot exploration based on wavefront algorithm," in *International Conference on Intelligent Robotics and Applications*. Springer, 2019, pp. 338–348.

[18] O. Walker, F. Vanegas, F. Gonzalez, and S. Koenig, "A deep reinforcement learning framework for uav navigation in indoor environments," in *2019 IEEE Aerospace Conference*. IEEE, 2019, pp. 1–14.

[19] C. Richter and N. Roy, "Safe visual navigation via deep learning and novelty detection," in *Proceedings of Robotics: Science and Systems (RSS)*. Robotics: Science and Systems Foundation, July 2017.

[20] D. Wang, C. Devin, Q.-Z. Cai, F. Yu, and T. Darrell, "Deep object-centric policies for autonomous driving," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8853–8859.

[21] G. Kahn, A. Villaflor, B. Ding, P. Abbeel, and S. Levine, "Self-supervised deep reinforcement learning with generalized computation graphs for robot navigation," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 5129–5136.

[22] F. Uwano and K. Takadama, "Communication-less cooperative q-learning agents in maze problem," in *Intelligent and Evolutionary Systems*. Springer, 2017, pp. 453–467.

[23] N. D. Pozza, L. Buffoni, S. Martina, and F. Caruso, "Quantum reinforcement learning: the maze problem," *arXiv preprint arXiv:2108.04490*, 2021.

[24] X. Yu, Y. Wu, and X.-M. Sun, "A navigation scheme for a random maze using reinforcement learning with quadrotor vision," in *2019 18th European Control Conference (ECC)*. IEEE, 2019, pp. 518–523.

[25] H. Li, Q. Zhang, and D. Zhao, "Deep reinforcement learning-based automatic exploration for navigation in unknown environment," *IEEE transactions on neural networks and learning systems*, vol. 31, no. 6, pp. 2064–2076, 2019.

[26] D. Zhu, T. Li, D. Ho, C. Wang, and M. Q.-H. Meng, "Deep reinforcement learning supervised autonomous exploration in office environments," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 7548–7555.

[27] J. G. Mangelson, D. Dominic, R. M. Eustice, and R. Vasudevan, "Pairwise consistent measurement set maximization for robust multi-robot map merging," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 2916–2923.

[28] S. Tjiharjadi, M. Wijaya, and E. Setiawan, "Optimization maze robot using a* and flood fill algorithm," *International Journal of Mechanical Engineering and Robotics Research*, vol. 6, no. 5, pp. 366–372, 2017.

[29] M. Xu, Z. Jia, J. Wang, L. A. Campos, and G.-K. Chang, "A novel data-compression technology for digital mobile fronthaul with lloyd algorithm and differential coding," in *Optical Fiber Communication Conference*. Optica Publishing Group, 2018, pp. Tu2K–2.

[30] P. Duda, M. Jaworski, L. Pietruczuk, and L. Rutkowski, "A novel application of hoeffding's inequality to decision trees construction for data streams," in *2014 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2014, pp. 3324–3330.