IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 4 October 2023, accepted 23 October 2023, date of publication 25 October 2023, date of current version 7 November 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3327559

RESEARCH ARTICLE

USV Port Oil Spill Cleanup Using Hybrid Multi-Destination RL-CPP

OREN ELMAKIS[1] **AND AMIR DEGANI**[1,2]**, (Member, IEEE)**
[1]Technion Autonomous Systems Program, Technion—Israel Institute of Technology, Haifa 3200003, Israel
[2]Civil and Environmental Engineering, Technion—Israel Institute of Technology, Haifa 3200003, Israel

Corresponding author: Amir Degani (adegani@technion.ac.il)

This work was supported by the Technion Autonomous Systems Program.

ABSTRACT Human activities are the principal contributors to oil pollution in marine ecosystems, thereby causing severe ecological damage. The high volume of vessel traffic operating in these areas contributes to the rapid contamination of the marine ecosystem, leading to frequent oil spill events, particularly near ports where congestion is prevalent. Addressing this issue today necessitates the involvement of numerous skilled personnel committed to the task. This team undertakes the repetitive and tedious work of surveying the area, detecting spills, and employing various techniques to address each oil slick. The emergence of Unmanned Surface Vehicle (USV) technology has introduced a promising alternative capable of alleviating the process of continuous monitoring and cleaning operations in proximal shoreline areas. This paper addresses the problem of USV cleaning operations near the port. The proposed method synthesizes a hierarchical architecture that integrates traditional global path planning for multi-destination oil spills, along with coverage path planning based on reinforcement learning, to adapt to dynamically changing oil spills. This combined architecture results in a comprehensive solution, allowing navigation within the port's vicinity to address each occurrence of oil pollution. To evaluate the effectiveness of this approach, we conducted an elaborate simulation designed to replicate port activities. The findings of this paper indicate a significant reduction in pollution levels due to USV operation and underscore the ability to acquire complex policies for dynamic coverage planning through the use of a reinforcement learning framework.

INDEX TERMS Autonomous agents, marine navigation, oil pollution, path planning, reinforcement learning.

I. INTRODUCTION

Over the past decade, public awareness and interest have grown regarding the negative consequences of oil spills caused by human operations. This increasing concern has motivated researchers to seek innovative solutions to environmental problems [1].

Marine oil pollution, predominantly caused by human activities, is one such problem with severe repercussions for ecosystems. Oil spills strip birds of their insulation, poison sea turtles, and inflict severe damage on the respiratory systems of dolphins and whales. Oil spill incidents fall into two categories: the frequent ones, usually small scale, and the rare large spills. Although events such as operational discharges, natural seepage, or pipe leakages are common, their detection, monitoring, and management are still notably challenging.

Addressing and mitigating marine oil spills is a complex process that demands comprehensive logistical operation, which can only be managed by highly experienced personnel. This operation necessitates expertise in incident management, monitoring methodologies, deployment of barrier-installation vessels, and specialized oil recovery techniques, such as the utilization of vacuums and absorbent materials [2].

One way to deal with oil spills is the utilization of dispersants, substances designed to break down oil slicks into smaller, more manageable droplets. These dispersants are usually applied from aerial platforms or small vessels capable of handling substantial payloads [3]. However, deploying

The associate editor coordinating the review of this manuscript and approving it for publication was Angelo Trotta.

these platforms requires skilled teams capable of accurately targeting oil slicks. Thus, the primary focus is addressing large oil spills since they pose an immediate danger, often overlooking the smaller-scale incidents.

Addressing these incidents often requires human operators to embark on prolonged, exhausting voyages that involve repetitive and laborious tasks. This, in turn, leads to dedicating numerous human resources to patrolling vessels near oil pipelines, anchoring ships, and monitoring predetermined areas, especially in the absence of automated methods and platforms. Autonomous systems have established a new path for confronting these problems, by facilitating the creation of methods and tools that make the process of observing and controlling water pollution more efficient. Although Remote sensing permits the automated identification of oil spills, the process of handling them continues to be an unresolved matter.

An Unmanned Surface Vehicle (USV) demonstrates considerable potential for advancing environmental disaster mitigation, specifically in providing a continuous response to oil spills of varying scales. These systems, designed to carry substantial payloads and to operate in marine environments, prove particularly effective when integrated with remote sensing platforms. Such integration enables them to tackle long-range tasks proficiently and accurately.

While a USV offers promise in mitigating environmental disasters such as oil spills, two key challenges remain unaddressed:

1. The absence of an efficient method for multi-destination path planning that ensures safe navigation near shores.

2. The lack of an adaptive Coverage Path Planning (CPP) method that effectively cleans dynamic oil spills in a marine environment.

Aiming to bridge these gaps, this study introduces a novel, hybrid path-planning strategy, specifically designed for a USV platform tackling multiple oil spill events near ports. Utilizing real-time remote sensing information, our method implements a visibility graph representation that enables us to determine secure paths within oil spills. These routes then serve to link separate spill groups by deploying classical path planning techniques.

Subsequently, the method addresses the challenge of internal path planning within each distinct group of spills. This problem is formulated as an Asymmetric Traveling Salesman Problem (ATSP), facilitating the identification of the shortest path among the routes. Lastly, given that each spill presents a dynamic coverage path planning challenge, the local planner based on Reinforcement Learning (RL) is employed to overcome this challenge and guide the USV cleaning operation.

The novelty of this paper is as follows:

1. Resolving the multi-destination ATSP among obstacles to secure an optimal path.

2. Integrating traditional methodologies with the Proximal Policy Optimization (PPO) RL technique in the context of USV path planning.

3. Implementing CPP to address the oil spill's motion and extension.

II. RELATED WORK

In this section, we provide an overview of the state-of-the-art research related to the problem of USV cleaning operations near ports, as addressed in this paper. Our study focuses on a hybrid model integrating multi-destination global path planning with an RL coverage path planning method for a USV platform.

A. UNMANNED SURFACE VEHICLES

In recent years, there has been a growing interest in the development of USV technology, spurring intensive progress in the creation of new applications. These applications encompass various fields, such as disaster management [1], search and rescue [4], and network infrastructure [5]. As illustrated by Jung et al. [6], a USV was employed to confront harmful algal blooms that disseminate across the lake's surface, allowing for prolonged deployments. Another illustrative example highlights the USV's capacity to engage in prolonged tasks, such as examining water quality, a challenge that requires continuous monitoring to ensure the safety of the water reservoir. Jo et al. [7] presented a USV platform capable of inspecting and reporting water conditions, thus acting as a maneuverable sensor. Similarly, Osen et al. [8] presented the use of a USV as a maneuverable sensor to address the problem of aquafarming, allowing for the automation of farm inspections.

B. USV OIL SPILL CLEANING

Emerging technologies in autonomous systems, specifically in the field of USVs, provide platforms that can significantly facilitate the process of oil spill cleaning. Studies addressing the oil spill challenge present USV platforms designed for various tasks, such as detection and sampling [9], boom deployment around anchoring ships [10], or boom maneuvering with two USVs for oil spill recovery [11]. Special cleaning approaches based on CPP have been utilized in studies focusing on large spills [12]. Furthermore, researchers have addressed the challenge posed by the movement and extension of oil spills over time. To tackle such dynamic scenarios [13], they propose dynamic path-planning techniques capable of adapting accordingly.

Despite substantial advancements in addressing key issues, a gap remained in a solution that utilizes USV for continuously managing scattered oil spills in congested maritime environments near ports.

C. MULTI-DESTINATION PATH PLANNING

Multi-destination path planning is a complex combinatorial problem that requires determining the sequence in which multiple targets or destinations should be reached. This involves devising an efficient and often optimal route that

guides the agent to each goal, ensuring every target is visited while minimizing cost measures.

A frequently encountered formulation of this problem is the Traveling Salesman Problem (TSP). This problem is centered around determining the most efficient, or shortest, route that an agent can take to visit a collection of destinations. Given that this challenge falls into the category of NP-Hard problems, it's often tackled using stochastic optimization strategies, such as simulated annealing [14], genetic algorithms [15], or the ant colony optimization [16].

Even though these strategies may not consistently yield the global optimum solution, precise methods such as the Held-Karp algorithm, rooted in dynamic programming, are available. However, the complexity of this method is $O(2^n n^2)$, which makes it intractable in many practical problems. To speed up the computation process, other methods that yield high-quality solutions have been devised, such as the heuristic 2-opt approach [17], and the approximation method known as the 1.5-approximation algorithm [18]. While the TSP has been thoroughly examined, there remains a significant challenge in dealing with multi-destination path planning in contexts featuring obstacles. In these scenarios, many of the destinations are obstructed by obstacles, which cause the line-of-sight path to become unfeasible. Consequently, determining the minimal distance between each pair of these destinations relies on methods such as Dijkstra's algorithm, A*, or Rapidly exploring Random Trees (RRT) [19]. However, due to their complexity, using these methods for each destination pair soon becomes computationally impracticable.

D. COVERAGE PATH PLANNING

The objective of a CPP algorithm is to design a path for an agent that covers an entire target area. Typically, CPP methods are designated for monitoring static environments with obstacles. The problem bears similarities to the TSP, in which the agent is required to visit every cell in the task area, a solution to which is NP-Hard [20]. Given the complexity of this challenge, studies have suggested employing multiresolution partitioning to divide the task area into distinct segments [13], or alternatively, structuring the site into convex parts [21]. In line with these approaches, Sheny et al. [22] presented a method based on a convolutional neural network approach to solve this problem and overcome the computational hurdle. Furthermore, Incorporating RL with classic methods can address CPP problems by breaking the task area into smaller segments and training RL agents to handle these smaller task regions [23].

This approach enables near-optimal solutions while substantially reducing computational time. Operations to clean oil spills necessitate a coverage path that encompasses the entirety of the oil slick area, enabling a USV to perform in-situ operations. Leveraging Image Processing and Self-Organizing Map methodologies, a USV tasked with spill removal can effectively detect and thoroughly cover the entire oil spill zone [12]. While CPP solutions have effectively tackled the issue of static coverage, there remains a necessity for real-time adaptive strategies to manage oil spill coverage, as the oil travels in the ocean medium.

E. REINFORCEMENT LEARNING

The swift progress in RL techniques in recent decades has unlocked new possibilities for tackling complex problems. Among these techniques, Deep learning-based RL architectures like deep Q-Network [24], deep deterministic policy gradient [25], PPO [26], and Soft Actor-Critic [27] have demonstrated exceptional capabilities in handling complex stochastic tasks. These methods have proven to be effective in areas such as cooperative path planning [28], navigating constrained environments [29], and patrolling water resources [30]. However, challenges arise in training tasks that require prolonged processes due to the inherent behavior of RL agents that rely on a specific, dense, and short-term reward system [31], [32]. This situation prompts the adoption of hybrid and hierarchical techniques that merge classical methods' stability and resilience with RL's efficiency and generality [33].

This way, the multi-tasking problem can be addressed by a hierarchical structure of classic planning for long-range tasks while the RL agent handles the sub-tasks [34].

This paper presents a solution for navigating congested marine environments during oil spill clean-up operations. The method tackles the ATSP by pinpointing oil spill locations and deploying an RL agent for their cleanup. A layered framework is used, supplemented with an RL agent specifically trained to map out the CPP route for the dynamically changing area.

According to this literature review, the research advancements of this proposed method are:

1. An adaptive CPP method that enables the addressing of dynamically changing areas.
2. The efficient path planning method for the multi-destination problem.
3. A hybrid method integrating classic and RL techniques to overcome the challenges posed by the sparse reward structure of the global problem.

III. METHODOLOGY

The main aim of this hierarchical system is to address the navigation of a USV for the purpose of cleaning up oil spill pollution in complex and congested environments. This system uses remote sensing data and the local sensing capabilities of the USV itself to operate in real-time and dynamically adjust the planned path as needed.

A. FRAMEWORK

In the study outlined in this paper, we have constructed a simulation framework replicating a congested marine port scenario Fig. 1. The framework is built around four fundamental components: (1) the shoreline and breakwater, which

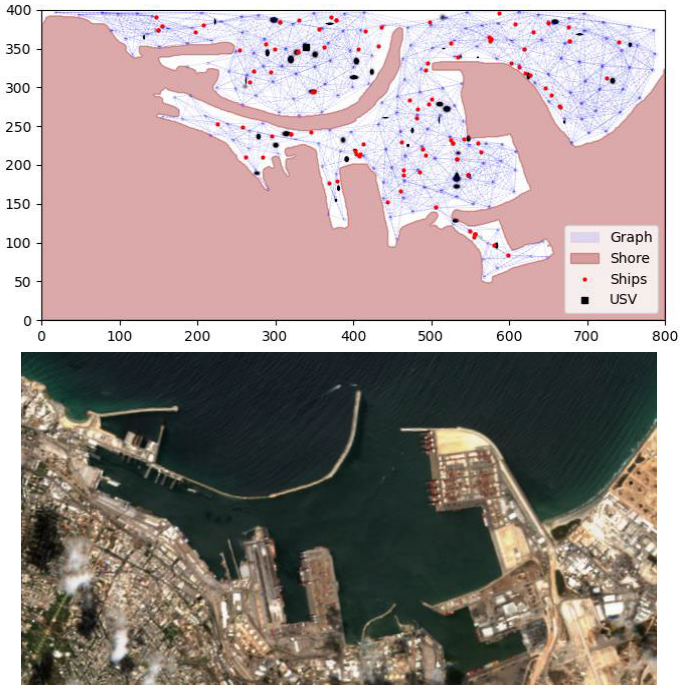


FIGURE 1. A simulation framework that replicates the Haifa port coastline and presents an initial stage scenario of a crowded port.

serve as obstacles, (2) ship routes, which function as a roadmap, (3) artificial ships that maneuver within the port region, and (4) unpredictably occurring spills originating from these vessels. We utilized Google Maps to accurately capture the shoreline and breakwater shape of various ports, presented in Fig. 1 as the brown area. Subsequently, employing the RRT algorithm, we established a comprehensive roadmap spanning the marine area to simulate vessel routes near the ports, shown as a blue graph in Fig. 1.

Following that, we devised arbitrary trajectories for each vessel and implemented a kinematic model [28], represented by the coordinates (X, Y, θ), in which only axial velocity was assumed, in accordance with the following model:

$$\begin{aligned} X^{i+1} &= X^i + \Delta t \cdot u_v^i \cdot cos\,(\theta) \\ Y^{i+1} &= Y^i + \Delta t \cdot u_v^i \cdot sin\,(\theta) \\ \theta^{i+1} &= \theta^i + u_\delta^i \cdot \Delta t \end{aligned} \tag{1}$$

In this model, X and Y stand for Cartesian coordinates, θ indicates the direction, (u_v^i, u_δ^i) are the specific actions taken, and Δt represents the time step. This is used to simulate the movement of each vessel during the scenario. Each of these vessels is presented as a red dot on the roadmap in Fig. 1.

During each iteration, the probability of a vessel spilling oil is determined by the equation:

$$P\left(x^{i+1}\right) = P\left(x^i\right) + \epsilon^{prob} \tag{2}$$

where $P(x^i)$ represents the probability of a vessel spilling oil in the current iteration, $P(x^{i+1})$ represents the probability in the following iteration, and ϵ^{prob} represents the change of likelihood resulting from the elapse of time. The oil spill is produced as a collection of Gaussian pollutants combined, each forming a distinct shape and adhering to a particular equation:

$$S^i = \sum_n \frac{1}{2\pi\sigma_x\sigma_y} e^{-\left(\frac{(x-\mu_x)^2}{2\sigma_x^2} + \frac{(y-\mu_y)^2}{2\sigma_y^2}\right)}. \tag{3}$$

S^i represents the cumulative pollution, constituted of n Gaussian pollutants. Here, σ_x and σ_y denote the dispersal of the function, while μ_x and μ_y signify the location of the spill peak, which is in the vicinity of the vessel region.

Lastly, the USV, shown as the black cube in Fig. 1, employs the same kinematic model as outlined in (1). The cleaning capacity area of the USV is denoted by C_l, and the USV tackles the spills by covering the polluted regions affected by these spills.

B. METHOD OVERVIEW

This methodology employs a hierarchical structure incorporating both global and local planning mechanisms to devise an oil cleanup strategy within a port-like region. The initial stage of global planning relies on the utilization of remote sensing imagery, deploying a visibility graph to link each oil spill with adjacent, visible spillages through a secure pathway. Consequently, these oil spills are grouped into interconnected subsets that have safe paths between them. These subsets are then connected by identifying a pair of nodes that offer the shortest route. Each interconnected subset is ultimately addressed by formulating the scenario as an ATSP. Upon

detecting each new oil spill, an RL-based local planner is deployed. This planner uses local vision data to efficiently guide the USV toward cleaning the spill.

C. GLOBAL VISIBLE ATSP

The process of global path planning utilizes remote sensing data to create the path for the USV. Assuming we can detect oil spills S^i close to the port through remote imagery, the USV needs to navigate the port effectively and attend to each spill event. This involves determining safe navigation paths among these oil spills. For this objective, we formulate the oil spills visibility graph $\mathbb{V}$. This graph connects spills that are within line of sight. The method for constructing this visibility graph is described in Algorithm 1.

Algorithm 1 Visibility Graph

Inputs: $\mathbb{V}^{\mathrm{E}} \leftarrow$ Initialize an empty edge set.
$\mathbb{V}^{V} \leftarrow$ Collection of oil spill nodes.
$\mathcal{O} \leftarrow$ Obstacles.
Output: $\mathbb{V} \leftarrow$ Visibility graph.
for each $\left\{\mathbb{V}^i \mid 0 \rightarrow n\right\}$ in $\mathbb{V}^V$ do:
 for each $\left\{\mathbb{V}^j \mid i+1 \rightarrow n\right\}$ in $\mathbb{V}^V$ do:
  if in *LineOf Sight* $\left(\mathcal{O}, \mathbb{V}^i, \mathbb{V}^j\right)$ then:
   Add edge $\mathbb{V}^E \leftarrow \left(\mathbb{V}^i, \mathbb{V}^j\right)$
return $\mathbb{V}$

The visibility graph provides all the direct safe routes that can be traveled without encountering obstacles. Now, we identify all the connected spill groups. These groups are defined as groups in that an obstacle-free path can connect any pair of spills. These groups are also called the interconnected subsets in the visibility graph.

As such, the challenge in this stage is to identify set $\mathcal{C}$, which contains all interconnected subsets derived from the visibility graph. To solve this challenge and detect these interconnected subsets, we execute the FindInterconnected-Subsets, as outlined in Algorithm 2. The process involves generating an empty subset called $\mathcal{C}_i$, and then randomly choosing a spill from the set, which comprises all the spills.

Algorithm 2 FindInterconnectedSubsets

Inputs: $\mathbb{V} \leftarrow$ visibility graph.
Output: $\mathcal{C} \leftarrow$ Collection of interconnected Subsets.
$\mathcal{C} \leftarrow \{\}$
$\mathbb{V}^{\mathrm{vis}} \leftarrow \emptyset$
for each $\left\{\mathbb{V}^i \mid 0 \rightarrow n\right\}$ in $\mathbb{V}$ do
 if $\mathbb{V}^i \notin \mathbb{V}^{vis}$ then
  $\mathcal{S} \leftarrow$ *FloodFill* $\left(\mathbb{V}^i, \mathbb{V}^{vis,\mathbb{V}}\right)$
  Add $\mathcal{S}$ to $\mathcal{C}$
Return $\mathcal{C}$

Next, we implement the *FloodFill* (Algorithm 3) until the interconnected subset is found. This iterative process is repeated for all the spills associated with interconnected

Algorithm 3 FloodFill

Inputs: $\mathbb{V} \leftarrow$ visibility graph.
$\mathbb{V}^s \leftarrow$ *Start* node.
$\mathbb{V}^{vis} \leftarrow$ Visited nodes.
Output: $\mathcal{S} \leftarrow$ Interconnected Subset.
Add $\mathbb{V}^s$ to $\mathbb{V}^{vis}$
$\mathcal{S} \leftarrow \{\mathbb{V}^s\}$
for each $\left\{\mathbb{V}^i \mid 0 \rightarrow n\right\}$ in $\mathbb{V}\left[\mathbb{V}^s\right]$ do
 if $\mathbb{V}^i \notin \mathbb{V}^{vis}$ then
  $\mathcal{S} \leftarrow \mathcal{S} \cup$ *FloodFill* $\left(\mathbb{V}^i, \mathbb{V}^{vis}, \mathbb{V}\right)$
Return $\mathcal{S}$

subsets. The result of this grouping procedure, which involves finding the interconnected subsets is depicted in Fig. 2.

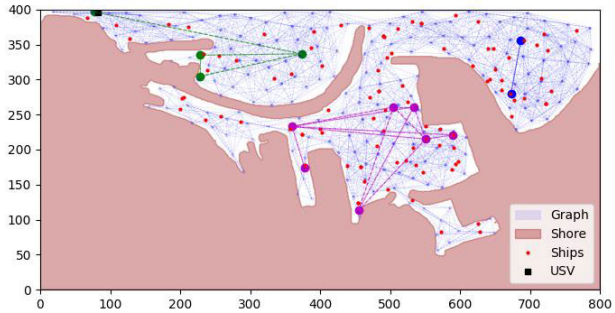


FIGURE 2. Illustrating the interconnected clusters derived from the visibility graph.

Subsequently, a connection between the generated subsets needs to be established, which requires identifying the two nearest accessible nodes within each interconnected subset.

The algorithm is designed to generate the shortest path that links two separate subsets. To achieve this, the algorithm must first identify the closest spills within the departing and arriving groups and connect them via the most minimal path.

Algorithm 4 MinimalPathFinding, commences the process by arbitrarily selecting oil spills from separate subsets.

Algorithm 4 MinimalPathFinding

Inputs: $\mathcal{S}^{\mathrm{D}} \leftarrow$ Depurture set.
$\mathcal{S}^{\mathrm{a}} \leftarrow$ Arrival set.
$\mathcal{O} \leftarrow$ Obstacles.
Output: $\mathcal{P} \leftarrow$ *path*
$\mathbb{V}^{\mathrm{d}} \leftarrow$ random node from $\mathcal{S}^{\mathrm{D}}$
$\mathbb{V}^{\mathrm{a}} \leftarrow$ random node from $\mathcal{S}^{\mathrm{a}}$
$\mathcal{TP} \leftarrow$ *Dijkstra* $\left(\mathbb{V}^{\mathrm{d}}, \mathbb{V}^{\mathrm{a}}\right)$
$\mathbb{V}_v^{\mathrm{d}} \leftarrow$ *FindVisibleNode* $\left(\mathcal{TP}, \mathcal{S}^{\mathrm{D}}, \mathcal{O}\right)$
$\mathbb{V}_v^{\mathrm{a}} \leftarrow$ *FindVisibleNode* (*Reverse*($\mathcal{TP}$), $\mathcal{S}^{\mathrm{a}}$, $\mathcal{O}$)
Cut $\mathcal{TP}$ from $\mathbb{V}_v^{\mathrm{d}}$ to $\mathbb{V}_v^{\mathrm{a}}$
$\mathbb{V}_{\mathrm{m}}^{\mathrm{d}} \leftarrow argmin_{d \in S^D} \left\| \mathrm{d} - \mathcal{TP}^0 \right\|$
$\mathbb{V}_{\mathrm{m}}^{\mathrm{d}} \leftarrow argmin_{a \in S^A} \| \mathrm{a} - \mathcal{TP}^n \|$
$\mathcal{P} \leftarrow \{\mathbb{V}_{\mathrm{m}}^{\mathrm{d}}, \mathcal{TP}, \mathbb{V}_{\mathrm{m}}^{\mathrm{a}}\}$
Return $\mathcal{P}$

Subsequently, these selected spills are connected by the shortest path, which is generated by the Dijkstra algorithm. This path is then utilized to search for the minimal connecting path between the subsets by finding the final visible nodes in each subset, as outlined in *FindVisibleNode* Algorithm 5.

Algorithm 5 FindVisibleNode

Inputs: $\mathcal{TP} \leftarrow$ Path
$\mathcal{S} \leftarrow$ Set
$\mathcal{O} \leftarrow$ Obstacles.
Output: $\mathbb{V} \leftarrow$ Visible Node.
for each $\left\{\mathcal{T}^i \mid 0 \rightarrow n\right\}$ in $\mathcal{TP}$ do:
 if $\forall \mathbb{V}^i \in \mathcal{S}, !\, LineOfSight\ \left(\mathcal{O}, \mathcal{TP}^i, \mathbb{V}^i\right)$ then
 return $\mathbb{V}^{i-1}$

The algorithm starts an iterative process in which it progresses along the path, searching for the last node that is visible in the departing subset and the first node in the arrival subset. These nodes are then used as initial and final locations in the connecting path between the subsets.

The next step involves generating the shortest path among the visited oil spills in each subset. As mentioned earlier, a graph was created that contains subsets and connecting paths between these sets, thereby ensuring the existence of an obstacle-free route between any two spills on the map.

Formulating the ATSP with designated starting and ending points allows the problem to be considered a point-to-point ATSP problem, thereby facilitating the solution. The approach to solving this problem, when provided with a predefined start-end couple, necessitates an auxiliary node.

The auxiliary node virtually links the initial and final nodes through a directed zero-cost edge, effectively converting our problem into a classic ATSP problem.

Consequently, a pathway from start to end is established without incurring additional costs. In order to solve our problem with minimal cost, it is necessary to transform our point-to-point ATSP into a classic ATSP problem formulation. This transformation is described in (4).

$$
\begin{aligned}
&min \sum_{i \in \mathcal{S}} \sum_{j \in \mathcal{S}, \boldsymbol{i<j}} c_{ij} x_{ij} \\
&s.t\ x_{ij} \in \{0,1\}, \quad i,j = 0,1,\ldots,n \\
&\quad u_i \in \mathbb{Z}, \quad i = 2,\ldots,n \\
&\quad \sum_{i=1, i \neq j} x_{ij} = 1, \quad j = 1,\ldots,n \\
&\quad \sum_{j=1, i \neq j} x_{ij} = 1, \quad i = 1,\ldots,n \\
&\quad u_i - u_j + (n+1) x_{ij} \leq n,\ \forall i \neq j,\ i \neq 1, j \neq 1 \\
&\quad 1 \leq u_i \leq n, \quad \forall i \neq 1 \\
&\quad x_{n+1,1} = 1 \\
&\quad x_{n,n+1} = 1
\end{aligned} \tag{4}
$$

In this equation, the relative distance between the nodes is denoted by c_{ij}, and the path is represented by a binary variable x_{ij}, which is equal to one if a specific path is taken and zero otherwise. To prevent subtours and ensure that each node is visited only once, the Miller-Tucker-Zemlin formulation is utilized. This formulation introduces the variable u_i, signifying the sequence of oil spill visits. Additionally, a dummy auxiliary node is defined as $n+1$ to ensure that node n is visited last. Accordingly, this connection is forced by setting the constraints $x_{n,n+1} = 1$, and $x_{n+1,1} = 1$, which are used to transform the point-to-point ATSP problem into a standard ATSP without incurring extra costs.

The last challenge in this process is to generate a global obstacle-free path that guides the USV through all the oil spills. However, for the ATSP solution to be viable, a fully connected, obstacle-free graph is required. Algorithm 2 guarantees only the existence of one route between any oil spills within the same subset, posing an additional constraint. This constraint, when combined with the requirements for a single visit to each spill and limited accessibility to certain spills, may lead to violations of obstacle limitations.

Therefore, in the final stage, we designed an iterative process that incrementally constructs an obstacle-free path among all the oil spills on the map and gradually generates the path to overcome this challenge.

The generation of the global path commences with Algorithm 1, which is tasked with constructing a visibility graph of the oil spills. This graph subsequently serves as the basis for identifying the available connected subsets among the oil spills. Upon completing this identification, the method calculates the sequence in which the USV visits these subsets, factoring in the relative distances of their centroids, and reorganizes the collection of subsets accordingly. This

Algorithm 6 GlobalPathSolver

Inputs: $\mathbb{V} \leftarrow$ visibility graph.
$\mathcal{O} \leftarrow$ Obstacles.
Output: $\mathcal{P}$ - *path*
set $\mathbb{V}^i$ to $\mathbb{V}$
While $\mathbb{V}^i \neq \emptyset$
 $\mathcal{C} \leftarrow$ FindInterconnectedSubsets $\left(\mathbb{V}^i\right)$
 $Cent \leftarrow IdentifyCentroids\ (\mathcal{C})$
 $dis \leftarrow ComputeCentroidsDistances\ (\ Cent\ )$
 $\mathcal{C} \leftarrow ReorderSubsets\ (\mathcal{C}, dis\ )$
 for each i in $\mathcal{C}$ do:
 $\mathcal{P}^{i,,ont} \leftarrow$ MinimalPathFinding $\left(\mathcal{C}^i, \mathcal{C}^{l+1}, \mathcal{O}\right)$
 $\mathcal{P}^{i,int} \leftarrow SolveATSP\ \left(\mathcal{C}^i, \mathcal{P}^0, \mathcal{P}^N\right)$
 $\mathcal{P}^c \leftarrow ConcatenatePaths\ \left(\mathcal{P}^{i,int}, \mathcal{P}^{i,con}\right)$
 for each $\left\{\mathcal{P}^j, \mathcal{P}^{j+1}\right\}$ in $\mathcal{P}$ do
 if $\mathcal{P}^j \models \mathbb{V}^j \cap !LineOfSight\ \left(\mathcal{O}, \mathcal{P}^j, \mathcal{P}^{j+1}\right)$ then
 $\mathbb{V}^i \leftarrow RemoveVertex\ \left(\mathbb{V}^i, \mathbb{V}^j\right)$
 else
 index $\leftarrow j$
 Break
 $\mathcal{P}^c_{connected} \leftarrow ConstructPath\ (\mathcal{P}^c, 0, index)$
return $\mathcal{P}^c_{connected}$

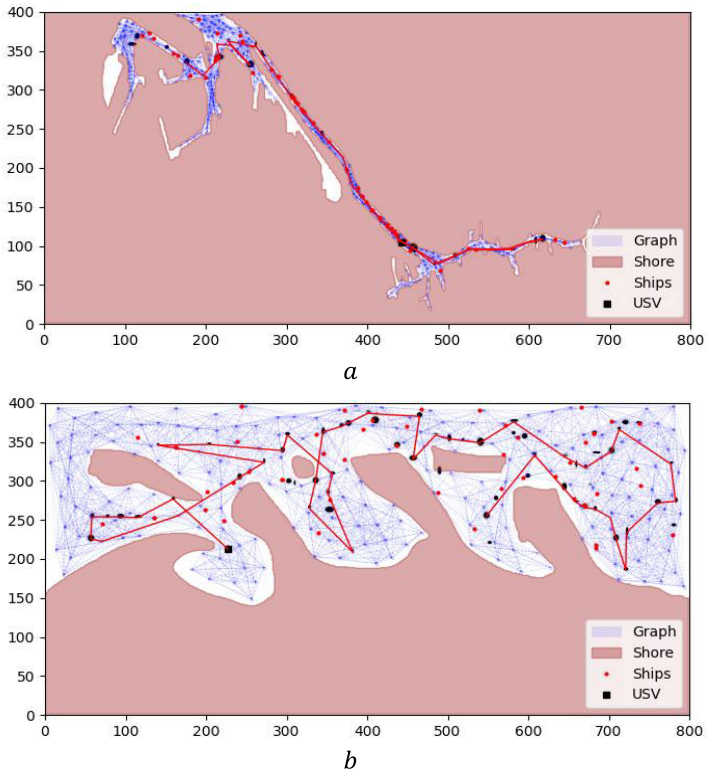


FIGURE 3. Presents a visualization of the global path planning solution. a) demonstrates a section of the Rotterdam port with narrow passages. b) illustration of a multi-docking port area.

reordered collection serves as the input for Algorithm 4. The algorithm identifies both the connecting path between the subsets, denoted as $\mathcal{P}^{i,con}$, and the interior path within each subset, denoted as $\mathcal{P}^{i,int}$. The interior subset is obtained by solving the ATSP problem as specified in (4). The entire process generates a candidate solution, denoted as $\mathcal{P}^c$, which undergoes an evaluation for connectivity. Only the connected sub-path of the candidate $\mathcal{P}^c_{connected} \subset \mathcal{P}^c$, is retained, and the method iteratively continues for the remaining unvisited oil spills. The process reaches completion when all oil pollutions have been addressed, as detailed in GlobalPathSolver Algorithm 6. Fig. 3 Presents the comprehensive solution for the global planning of USV oil cleaning.

D. RL-CPP

The RL-CPP is an approach founded on RL techniques explicitly intended to manage the dynamic characteristics of oil spills that demonstrate temporal growth. This methodology is outlined in the scheme represented in Fig. 4 and is designed to tackle the complexities of managing oil spill incidents in a dynamic and localized context.

The task commences with an initial state s^i, and the agent's mission is to effectively contain oil spills within a designated local area, denoted as M^{nxn}. The state space of the task, denoted by S, is defined within a discrete $\mathbb{X}\ X\ \mathbb{Y}$ coordinate system, wherein the obstacle-free area is represented as $\mathbb{X}\ X\ \mathbb{Y} \in \mathcal{M}_{free}$. The local area $\mathcal{M}$, which is identified by cells where the oil density exceeds a specific threshold, is represented by ϵ, and expressed as

$$\mathcal{M}^i_p = \begin{cases} 1 & P^i \geq \epsilon \\ 0 & else \end{cases} \tag{5}$$

The reward shaping presents a significant challenge during the development of the RL method, as it necessitates formulating a reward function that encompasses all aspects of the cleaning scenario. We divided the reward function into three distinct components to address this challenge.

The first reward function $\mathcal{R}_1$, the net impact of the cleaning action performed by the agent and formulated as:

$$\mathcal{R}^i_1 = \sum_i \mathcal{M}^i_p - \sum_i \mathcal{M}'^i_p \tag{6}$$

Here, $\mathcal{M}^i_p$ refers to the number of cells where the oil density exceeds the threshold ϵ before the execution of the agent's action a. Subsequently, $\mathcal{M}'^i_p$ represents the remaining oil density in the area following the agent's action. The second reward function aims to emphasize the reduction of unnecessary movement and avoidance of repeated positioning.

$$\mathcal{R}^i_2 = \begin{cases} -0.1 & free\ location \\ -0.5 & occupied\ location \end{cases} \tag{7}$$

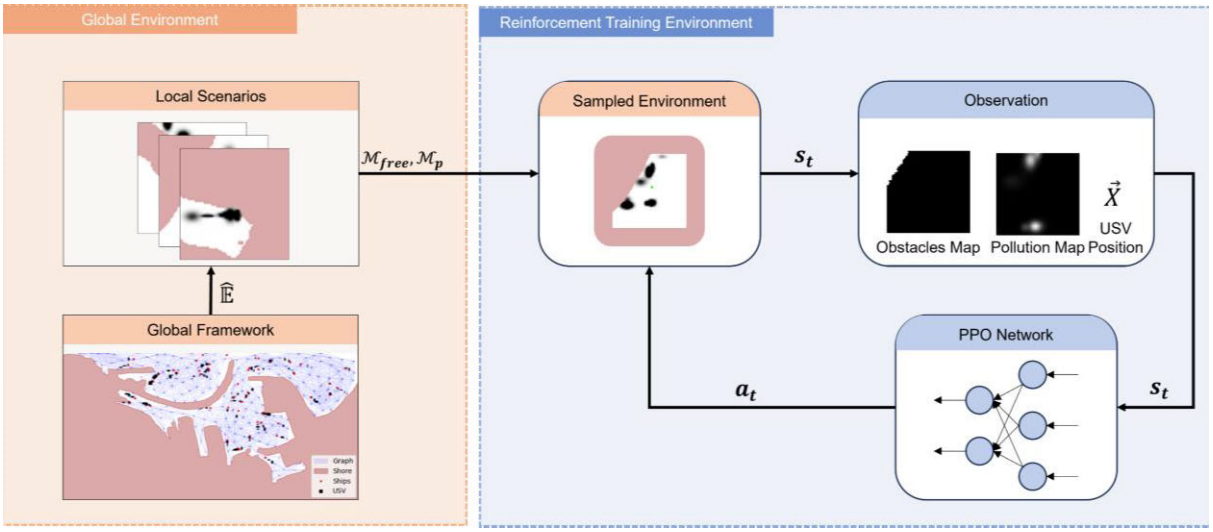


FIGURE 4. The system overview begins with creating a global framework for the training process of the RL agent. This framework extracts local scenarios, which are compiled into a dataset. This dataset is then utilized to train the agent's local CPP.

The third reward component is a penalty for colliding with obstacles.

$$\mathcal{R}_3^i = \begin{cases} 0 & obstacle\ free \\ -50 & obstacle \end{cases} \tag{8}$$

These components together assemble the reward function as

$$\mathcal{R} = \mathcal{R}_1^i + \mathcal{R}_2^i + \mathcal{R}_3^i \tag{9}$$

The agent's course of action is governed by the policy π, which dictates the agent's response based on the state it receives. Within the framework of the Markov decision process, the objective is to derive a policy π that maps states to actions $\pi : S \rightarrow A$ while maximizing the agent's reward for mitigating oil pollution. Specifically, given an initial state s, the aim is to find policy π^* that satisfies the following condition $\sum_i \mathcal{M}'^i_p = 0$.

E. TRAIN RL AGENT

The training scenarios for the RL agent are derived from the global framework, as illustrated in Fig. 4. Initially, local data from multiple oil pollution events were collected to establish a dataset, denoted as $\widehat{\mathbb{E}}$. The data collection process included the detection of oil spills, as well as the capture and storage of data for each local event. This data contained specific details such as the locations of obstacle-free areas, represented as $\mathcal{M}_{free}$, and the pollution data, denoted by $\mathcal{M}_p$. Once collected, this data was integrated into the RL framework to provide training data for the agent.

The training process commences with the RL framework randomly selecting a pair of obstacle and pollution maps from the dataset $\mathcal{M}_{free}, \mathcal{M}_P^{init} \sim \widehat{\mathbb{E}}$. A random initial state, s^{init}, is set for the agent. Based on this initial state, the first set of observations is recorded. These observations include limited-range visual data of obstacles $\mathcal{M}^i_{free,local}$, and oil slicks $\mathcal{M}^i_{p,local}$, in the vicinity of the USV.

The policy subsequently dictates the agent's movements at each time step. This continuous interaction between the agent's observed environment and its subsequent actions evolved throughout the RL training process, enabling the agent to learn how to navigate and clean oil spills effectively.

Fig. 5 presents the training area involving oil spills, where the agent is confined to a restricted area, and operates to clean the spills. The agent's cleaning capability is localized around its position, denoted by $C^i_{p,local}$, and it operates without prior knowledge about the local pollution shape.

F. PROXIMAL POLICY OPTIMIZATION

The PPO [26] algorithm achieves stability by balancing the trade-off between exploration and exploitation. This ensures the reliability of the acquired policy. Such stability is attained by an objective function that restricts substantial changes in the policy during each update, thereby providing both stability and robustness to the method.

The process initiates with the agent gathering data from the environment based on the existing policy, as illustrated in Fig. 4. These experiences are stored in a buffer and subsequently utilized to compute the gradients essential for updating the policy parameters. The gradient estimator is given as follows:

$$\hat{g} = \widehat{\mathbb{E}}_t \left[\nabla_\theta log \pi_\theta \left(a_t \mid s_t \right) \hat{A}_t \right] \tag{10}$$

Consequently, policy gradient estimator, is expressed as

$$L^{PG}(\theta) = \widehat{\mathbb{E}}_t \left[log \pi_\theta \left(a_t \mid s_t \right) \hat{A}_t \right] \tag{11}$$

Although straightforward Optimization on $L^{PG}(\theta)$ might fail, the PPO approach ensures training stability using a unique objective function.

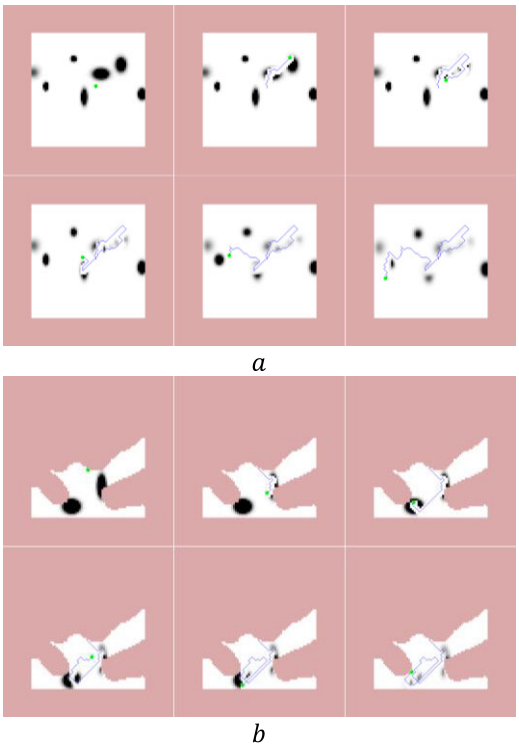


FIGURE 5. The RL agent is represented in green and located in a restricted area, surrounded by obstacles shown in brown. Guided by a policy, the agent works to clean up the local pollution, which is marked in black, following a path indicated in blue. Sub-figures *a* and *b* demonstrate the agent cleaning process in multi-destination and obstacle surrounded scenarios.

The method relies on a crucial component termed the probability ratio $r(\theta)$. The ratio measures the extent of deviation between the new policy's probability of taking specific action and the old policy's probability of taking the same action.

$$r_t(\theta) = \frac{\pi_\theta\left(a_t \mid s_t\right)}{\pi_\theta^{old}\left(a_t \mid s_t\right)} \tag{12}$$

The ratio serves as a component of PPO's policy objective function, known as a clipped surrogate objective. This function is specifically designed to ensure that the new policy does not substantially diverge from its previous policy. The clipped surrogate objective is defined as

$$J^{CLIP}(\theta) = \mathbb{E}\Big[min\left(r_t(\theta)\hat{A}_t \right. \\ \left. min\left(clip\left(r_t, 1-\epsilon, 1+\epsilon\right)\hat{A}_t\right)\right) \tag{13}$$

In this equation, $\hat{A}_t$ is the Generalized Advantage Estimation (GAE), which can be determined using the following equation:

$$A_t^k = \sum_{l=0}^{k-1} (\gamma\lambda)^l \delta_{t+l}^V \tag{14}$$

Here, δ_{t+l}^V represents the temporal difference residual, which calculated according to the equation:

$$\delta_t = r_t + \gamma V\left(s_{t+1}\right) - V\left(s_t\right) \tag{15}$$

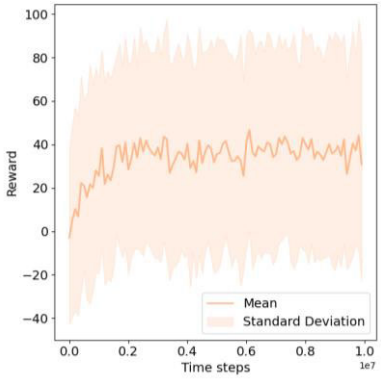


FIGURE 6. Displays the performance metrics of the RL agent throughout its training phase within a randomly configured local environment.

In addition to policy optimization, PPO uses a value function $V(s)$ to estimate the expected future return at each step. This function is referred to as value function loss and is computed as the mean squared error between the predicted value and the actual return.

$$L^{VF}(\phi) = \mathbb{E}\left[\left(V(s;\phi) - \mathcal{R}\right)^2\right] \tag{16}$$

The algorithm initiated by gathering transitions into a set of buffers denoted as $\mathcal{D}$. At each timestep, the policy dictates an action a_t, that the agent subsequently executes on its environment. This action produces a transition data consisting of the action a_t, the present state s_t, the subsequent state s_{t+1}, and the associated reward r_t reward. Following this, computations are performed to determine the cumulative reward $\mathcal{R}^j$, the temporal difference residual δ, and the GAE $\hat{A}^j$. Utilizing these computed values, the algorithm calculates the probability ratio $r\left(\theta\right)$, and the PPO objective PPO_{obj} calculated. Once all transitions in a trajectory are processed, a batch is randomly selected, and the policy parameters θ are subsequently updated based on this selected batch.

IV. EXPERIMENTS

To address the simulative experiments of this study, we examined separately and combined the global and local path planning.

A. LOCAL RL-CPP EXPERIMENTS

This section introduces an analysis focused on the implementation of the RL method in a local pollution cleaning context. The primary objective of the RL agent is to maneuver through various obstacles to cleanse multiple instances of oil pollution across differing local scenarios. The agent initiates the task in position, denoted as s^i, is randomly selected within $\mathcal{M}^{free}$. The agent's visual capabilities, represented by $\mathcal{M}^i_{free,local}$ and $\mathcal{M}^i_{p,local}$ are defined as a $10x10$ mask. Additionally, the cleaning capacity of the agent $C^i_{p,local}$ is $3x3$ mask.

The scenario was evaluated utilizing the PPO algorithm, configured with two hidden layers, each containing 512 units. The cleaning mask of the RL agent was specifically designed to remove all adjacent spills surrounding the agent's location.

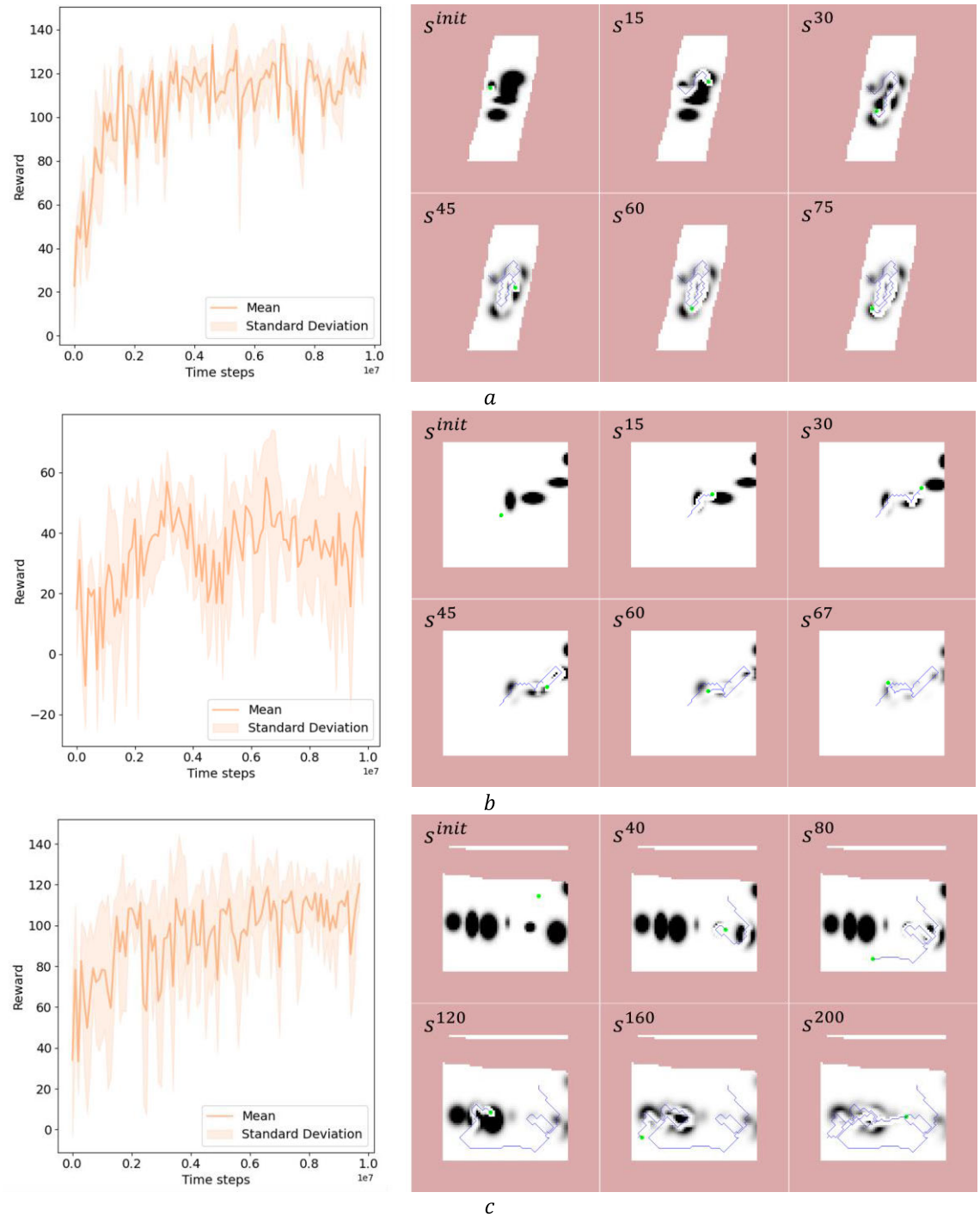


FIGURE 7. A composite representation of three distinct scenarios of RL policy navigating local cleaning scenarios. The figure consists of six individual images illustrating the spatial progressions of the RL agent in cleaning the localized pollution areas at different stages. Accompanied by a graph, the figure also details the corresponding reward metrics over the course of the training.

In addition, the pollution threshold value, denoted by ϵ^{prob} was set at 0.1.

In the training process of the RL agent, we employ a specific set of hyperparameters to optimize performance. We set the clipping range $\epsilon = 0.2$ to moderate the policy update and ensure stability. The learning rate is configured to $\alpha = 3 \cdot 10^{-4}$ to regulate the rate of policy optimization, and entropy regularization is incorporated with $\beta = 3 \cdot 10^{-4}$ to balance exploration and exploitation trade-offs. Furthermore, we use a discount factor $\gamma = 0.99$ to adequately weigh future rewards, and the trace decay is set to $\lambda = 0.95$ to manage the bias-variance trade off in the advantage estimation.

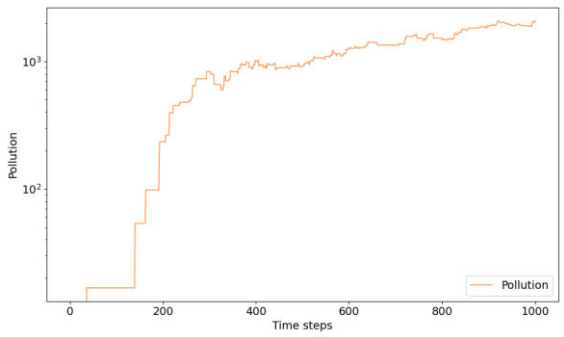


FIGURE 8. During the experimental phase, the oil pollution levels were presented on a logarithmic scale.

During the training process for our reinforcement learning model, a total of 10^7 time steps were executed across 32 parallel environments, a batch size of 64, and 400 distinct local environments. This comprehensive training process necessitated 8 hours of computational time on a single NVIDIA GeForce RTX 4080 GPU.

The evaluation metrics included both the mean value and the standard deviation of the agent's reward, computed based on 1000 epochs. The results of this experiment are depicted in Fig. 6. These results present a general improvement in the performance of the RL during the training process.

Fig. 7a demonstrates a local spill located within a narrow passage. The depiction of the cleaning process, which spans from $s^{init} \rightarrow s^{75}$ is performed by the USV. The USV performs a spiral motion with overlapping movement, as presented in s^{45}. In the presented scenario, a significant increase in rewards is observed throughout the training process, with rewards transitioning from 20 to 120.

Fig. 7b, a local scenario is presented, featuring multiple discrete spill locations. As illustrated, the agent initiates the process with an initial cleaning of the spills, subsequently returning for a secondary cleaning to remove the residuals. This event is characterized by a sparse reward structure, owing to the presence of several minor, distinct spills within the scenario. The reward pattern throughout the training

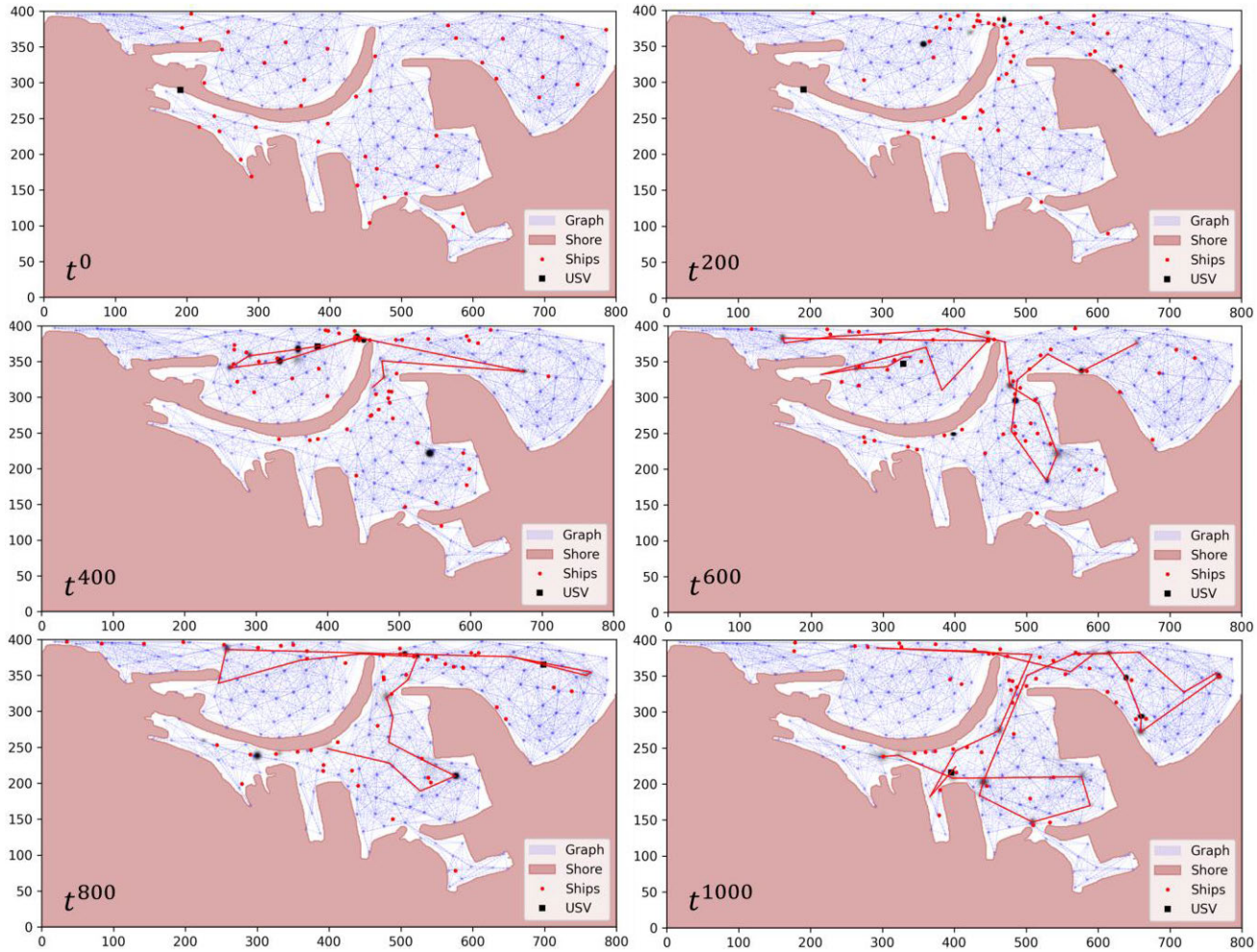


FIGURE 9. Sequential snapshots of the global path planning of the USV at six distinct time intervals. These images depict the evolution of the USV global navigation.

exhibits fluctuations, starting with an approximate average value of 20 and reaching 60 by the end of the training process.

Fig. 7c, a local spill scenario is depicted, featuring oil slicks positioned to both the left and the right of the local scenario. The agent commences its action within the visible sequence from $s^{init} \rightarrow s^{80}$, and subsequently starts to explore until it reaches s^{120}. From $s^{120} \rightarrow s^{200}$, the USV proceeds to clean the second location. Throughout the training process, rewards distinctly increase, with the average value rising from 60 to 120.

B. PLANNING EXPERIMENTS

The experiments involved a USV employing global planning combined with the RL policy designed for local oil pollution scenarios. Within this context, the maximal linear velocities for the USV and the vessels were set to $u_v^i = 10$, and $u_v^i = 1$ respectively. Both the USV and the vessels had maximal angular velocities set at $u_\delta^i = 1$. Additionally, the spillage rate for the vessels was determined in accordance with (2), where the increase factor for spillage probability was defined as $\epsilon = 10^{-5}$.

In Fig 8, the levels of oil pollution are represented on a logarithmic scale, showing an initial rise while the USV remains in an idle state during the time interval $t^0 \rightarrow t^{200}$. Following this phase, the USV initiates its cleaning process, resulting in a significant decline in the rate of increase of the pollution level. This reduction in the rise rate is prominent and is aligned with the USV's cleaning process.

Furthermore, Fig. 9 depicts the global planning process, specifically illustrating the evaluation of oil spills from vessels and the corresponding cleaning process executed by the USV during the experiment. The global path planning begins at t^{200} and is updated every 200 time steps or when the global path is completed. This update executes the global path planning process to create a new global route. The experiment was conducted in a simulated area of the Haifa port environment, with 50 simulated vessels randomly navigating within the port area.

During each pollution incident, the USV initiates the local planner to address the situation, and then continues the global route once the local cleaning process is completed.

In the given context, one global time step is equivalent to ten state transitions, expressed as $t^0 \rightarrow t^1 = s^0 \rightarrow s^{10}$. Fig. 10 illustrates various situations related to spill cleaning, showcasing fragments that include both single and multiple instances. These instances exhibit differing dispersion attributes, whether the spills are spread out or centralized. The cleaning operation is constrained to a total of 50 state transitions during the experiment.

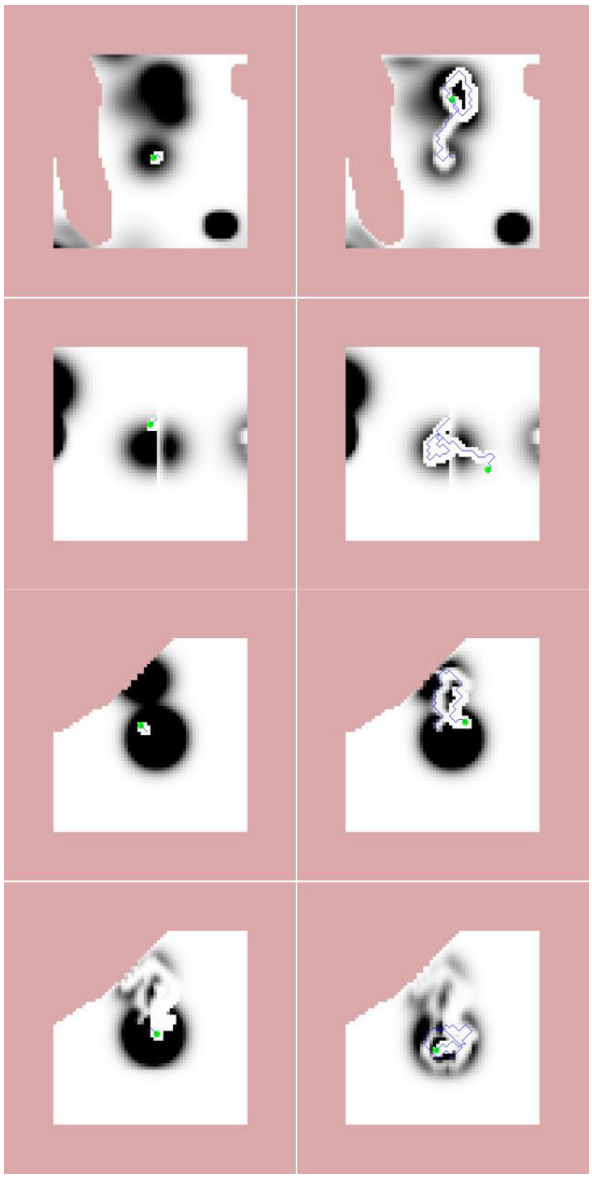


FIGURE 10. Demonstrate four distinct cleaning scenarios along the Experiment, where the left-side image presents the initial state, and the right-side shows the final stage of the cleaning process.

V. CONCLUSION AND FUTURE WORK

in this work, several key findings emerge that underscore the efficacy of the applied methodologies. As presented in Sec. IV, the hierarchical method employed herein combines global planning for long-term missions with a RL policy for short-term cleaning tasks. This approach effectively addresses challenges identified in the literature review, specifically those associated with cleaning dynamically changing multi-destination oil spills.

These challenges lead to development complexities such as efficiently clustering the oil spills into groups, identifying the departing and arriving oil spills in each group, and solving the ATSP problem with specific arrival and departure nodes in the absence of fully connected graph. Furthermore, incorporating an RL method with a classic approach requires delicate tuning of parameters to ensure seamless integration.

Fig. 6 demonstrates the RL agent's learning process, combined with Fig. 7, underscores the RL policy's capacity to yield efficient cleaning paths in an area cluttered with obstacles and multiple oil spills.

Furthermore, Fig. 8 illustrates the impact of the USV on reducing pollution levels within the port area, substantiating the effectiveness of the USV cleaning process. The data depicts a reduction in pollution levels, led by the USV's cleaning process.

As this is ongoing research, future work will focus on optimizing the single USV scenario and developing new methods for multi-agent scenarios.

This will include delivering improved versions of this method with adaptive state transition numbers during local cleaning and optimizing the periods for updating global path planning. An adaptive switching policy between global path planning and the RL will also be developed, aiming to generate optimal solutions that take into account RL policy's path.

Extending this method to multi-agent solutions will not only have a more significant impact on reducing pollution levels but will also open substantial research options. This extension encompasses various aspects: the allocation of responsibilities for pollution cleaning among team members and the establishment of policies for information exchange, such as a policy that could be based on event-triggered target allocation. Another significant challenge associated with this method is the acquisition of remote sensing data under conditions that impair visibility, such as cloudy weather. This particular challenge, however, presents opportunities for advancements in multiple areas: the development of new methods for efficient port surveillance, the establishment of protocols for data sharing tailored specifically to oil spill detection, and the formulation of robust control solutions employing both centralized and distributed intelligent systems to meet defined performance criteria. Lastly, research opportunities in RL include the acceleration of training through distributed strategies and the development of collaborative policies via distributed learning.

REFERENCES

[1] V. Jorge, R. Granada, R. Maidana, D. Jurak, G. Heck, A. Negreiros, D. dos Santos, L. Gonçalves, and A. Amory, "A survey on unmanned surface vehicles for disaster robotics: Main challenges and directions," *Sensors*, vol. 19, no. 3, p. 702, Feb. 2019, doi: 10.3390/S19030702.

[2] A. Dhaka and P. Chattopadhyay, "A review on physical remediation techniques for treatment of marine oil spills," *J. Environ. Manage.*, vol. 288, Jun. 2021, Art. no. 112428, doi: 10.1016/J.JENVMAN.2021.112428.

[3] N. Brazil, S. Nakhla, and S. Kenny, "Deployable oil dispersant system for fixed wing aircraft," in *Proc. St. John's, OCEANS*, Jan. 2015, pp. 1–9, doi: 10.1109/OCEANS.2014.7003122.

[4] G. De Cubber, D. Doroftei, D. Serrano, K. Chintamani, R. Sabino, and S. Ourevitch, "The EU-ICARUS project: Developing assistive robotic tools for search and rescue operations," in *Proc. IEEE Int. Symp. Saf., Secur., Rescue Robot. (SSRR)*, Oct. 2013, pp. 1–4, doi: 10.1109/SSRR.2013.6719323.

[5] G. Antonelli, "Advancing the EU marine robotics research infrastructure network: The EU marine robots project," in *Proc. OCEANS San Diego Porto*, Sep. 2021, pp. 1–10, doi: 10.23919/OCEANS44145.2021.9705791.

[6] S. Jung, H. Cho, D. Kim, K. Kim, J.-I. Han, and H. Myung, "Development of algal Bloom removal system using unmanned aerial vehicle and surface vehicle," *IEEE Access*, vol. 5, pp. 22166–22176, 2017, doi: 10.1109/ACCESS.2017.2764328.

[7] W. Jo, Y. Hoashi, L. L. Paredes Aguilar, M. Postigo-Malaga, J. M. Garcia-Bravo, and B.-C. Min, "A low-cost and small USV platform for water quality monitoring," *HardwareX*, vol. 6, Oct. 2019, Art. no. e00076, doi: 10.1016/j.ohx.2019.e00076.

[8] O. L. Osen, A. Havnegjerde, V. Kamsvag, S. Liavaag, and R. T. Bye, "A low cost USV for Aqua farm inspection," in *Proc. Techno-Ocean Return Oceans*, 2017, pp. 291–298, doi: 10.1109/Techno-Ocean.2016.7890664.

[9] W. A. Maawali, A. Al Naabi, M. Al Yaruubi, A. Saleem, and A. A. Maashri, "Design and implementation of an unmanned surface vehicle for oil spill handling," in *Proc. 1st Int. Conf. Unmanned Vehicle Systems-Oman (UVS)*, Feb. 2019, pp. 1–6, doi: 10.1109/UVS.2019.8658267.

[10] J. M. Giron-Sierra and J. F. Jimenez, "Using an USV for automatic deployment of a boom around a ship: Simulation and scale experiment," in *Proc. 2018 MTS/IEEE Charleston OCEAN*, Jan. 2019, pp. 1–9, doi: 10.1109/OCEANS.2018.8604707.

[11] J. M. Giron-Sierra, A. T. Gheorghita, G. Angulo, and J. F. Jimenez, "Towing a boom with two USVs for oil spill recovery: Scaled experimental development," in *Proc. 13th Int. Conf. Control Automat. Robot. Vis. (ICARCV)*, 2014, pp. 1729–1734, doi: 10.1109/ICARCV.2014.7064577.

[12] S. Luo, "Image processing and model-based spill coverage path planning for unmanned surface vehicles," in *Proc. MTS/IEEE Seattle*, Oct. 2019, pp. 1–9, doi: 10.23919/OCEANS40490.2019.8962662.

[13] J. Song, S. Gupta, J. Hare, and S. Zhou, "Adaptive cleaning of oil spills by autonomous vehicles under partial information," in *Proc. IEEE Conf. OCEANS*, Sep. 2023, pp. 1–5. [Online]. Available: https://ieeexplore-ieee-org.ezlibrary.technion.ac.il/document/6741246

[14] X. Geng, Z. Chen, W. Yang, D. Shi, and K. Zhao, "Solving the traveling salesman problem based on an adaptive simulated annealing algorithm with greedy search," *Appl. Soft Comput.*, vol. 11, no. 4, pp. 3680–3689, Jun. 2011, doi: 10.1016/J.ASOC.2011.01.039.

[15] J.-Y. Potvin, "Genetic algorithms for the traveling salesman problem," *Ann. Oper. Res.*, vol. 63, pp. 339–370, Jun. 1996, doi: 10.1007/BF02125403.

[16] J. Yang, X. Shi, M. Marchese, and Y. Liang, "An ant colony optimization method for generalized TSP problem," *Prog. Natural Sci.*, vol. 18, no. 11, pp. 1417–1422, Nov. 2008, doi: 10.1016/J.PNSC.2008.03.028.

[17] S. Hougardy, F. Zaiser, and X. Zhong, "The approximation ratio of the 2-Opt heuristic for the metric traveling salesman problem," *Oper. Res. Lett.*, vol. 48, no. 4, pp. 401–404, Jul. 2020, doi: 10.1016/j.orl.2020.05.007.

[18] R. Zenklusen, "A 1.5-approximation for path TSP," in *Proc. Annu. ACM-SIAM Symp. Discrete Algorithms*, May 2018, pp. 1539–1549, doi: 10.1137/1.9781611975482.93.

[19] S. M. LaValle, *Planning Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 2006, doi: 10.1017/CBO9780511546877.

[20] E. Galceran and M. Carreras, "A survey on coverage path planning for robotics," *Robot. Auto. Syst.*, vol. 61, no. 12, pp. 1258–1276, Dec. 2013, doi: 10.1016/J.ROBOT.2013.09.004.

[21] Z. Han, Y. Hao, and Y. Zhang, "A full coverage path planning method for unmanned surface vessels," in *Proc. 3rd Int. Conf. Geol., Mapping Remote Sens. (ICGMRS)*, Apr. 2022, pp. 428–433, doi: 10.1109/ICGMRS55602.2022.9849262.

[22] Z. Sheny, P. Agrawal, J. P. Wilson, R. Harvey, and S. Gupta, "CPPNet: A coverage path planning network," in *Proc. Oceans Conf. Record*, 2021, pp. 1–5, doi: 10.23919/OCEANS44145.2021.9705671.

[23] P. T. Kyaw, A. Paing, T. T. Thu, R. E. Mohan, A. Vu Le, and P. Veerajagadheswar, "Coverage path planning for decomposition reconfigurable grid-maps using deep reinforcement learning based travelling salesman problem," *IEEE Access*, vol. 8, pp. 225945–225956, 2020, doi: 10.1109/ACCESS.2020.3045027.

[24] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015, doi: 10.1038/NATURE14236.

[25] T. P. Lillicrap, "Continuous control with deep reinforcement learning," in *Proc. 4th Int. Conf. Learn. Represent. (ICLR)*, Sep. 2016, pp. 1–14.

[26] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.

[27] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proc. 35th Int. Conf. Mach. Learn. (ICML)*, vol. 5, Jan. 2018, Accessed: Jul. 16, 2023, pp. 2976–2989.

[28] X. Zhou, P. Wu, H. Zhang, W. Guo, and Y. Liu, "Learn to navigate: Cooperative path planning for unmanned surface vehicles using deep reinforcement learning," *IEEE Access*, vol. 7, pp. 165262–165278, 2019, doi: 10.1109/ACCESS.2019.2953326.

[29] J. Amendola, L. S. Miura, A. H. R. Costa, F. G. Cozman, and E. A. Tannuri, "Navigation in restricted channels under environmental conditions: Fast-time simulation by asynchronous deep reinforcement learning," *IEEE Access*, vol. 8, pp. 149199–149213, 2020, doi: 10.1109/ACCESS.2020.3015661.

[30] S. Y. Luis, D. G. Reina, and S. L. T. Marín, "A deep reinforcement learning approach for the patrolling problem of water resources through autonomous surface vehicles: The Ypacarai lake case," *IEEE Access*, vol. 8, pp. 204076–204093, 2020, doi: 10.1109/ACCESS.2020.3036938.

[31] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Process. Mag.*, vol. 34, no. 6, pp. 26–38, Nov. 2017, doi: 10.1109/MSP.2017.2743240.

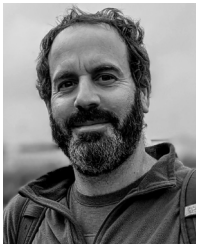
[32] Y. Qiao, J. Yin, W. Wang, F. Duarte, J. Yang, and C. Ratti, "Survey of deep learning for autonomous surface vehicles in marine environments," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 4, pp. 3678–3701, Apr. 2023, doi: 10.1109/TITS.2023.3235911.

[33] A. Faust, "PRM-RL: Long-range robotic navigation tasks by combining reinforcement learning and sampling-based planning," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, May 2018, pp. 5113–5120, doi: 10.1109/ICRA.2018.8461096.

[34] X. Zhang, M. Wu, H. Ma, T. Hu, and J. Yuan, "Multi-task long-range urban driving based on hierarchical planning and reinforcement learning," in *Proc. IEEE Conf. Intell. Transp. Syst. (ITSC)*, Sep. 2021, pp. 726–733, doi: 10.1109/ITSC48978.2021.9564705.

OREN ELMAKIS received the B.Sc. degree in mechanical engineering from the Technion—Israel Institute of Technology, Haifa, Israel, in 2019.

Currently, he is a direct-track Ph.D. student at the Civil, Environmental, and Agricultural Robotics Lab (CEAR) affiliated with the Technion Autonomous Systems and Robotics Program. His research interests include robotic multi-agents decision making, motion control, and collaborative state estimation in civil and environmental applications.



AMIR DEGANI (Member, IEEE) received the B.Sc. degree (summa cum laude) in mechanical engineering from the Technion—Israel Institute of Technology, Haifa, Israel, in 2002, and the M.S. and Ph.D. degrees in robotics from Carnegie Mellon University, Pittsburgh, PA, USA, in 2006 and 2010, respectively. From 2011 to 2019, he was an Assistant Professor in civil and environmental engineering with the Technion—Israel Institute of Technology and the Director of the Civil, Environmental and Agricultural Robotics (CEAR) Laboratory researching robotic legged locomotion and autonomous systems in civil and agriculture applications. In 2019, he was promoted to a tenured Associate Professor. He is currently an Associate Professor with the Technion—Israel Institute of Technology. His research interests include mechanism analysis, synthesis, control and motion planning, and design with emphasis on minimalistic concepts and the study of nonlinear dynamic hybrid systems. He has six patents in the robotics field and he has received the Best Paper Award at the IEEE BioRob Conference, in 2006, the Best Video Award at the IEEE ICRA Conference, in 2010, and the JTCF Novel Technology Paper Award in IEEE IROS 2015. He is also an Associate Editor of IEEE Robotics and Automation Letters and he has been an Associate Editor of IEEE Transactions on Robotics and the IEEE ICRA Conference and IROS Conference.