2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

REACT: Real-time Entanglement-Aware Coverage Path Planning for Tethered Underwater Vehicles

Abdelhakim Amer, Mohit Mehndiratta, Yury Brodskiy, Bilal Wehbe, and Erdal Kayacan

***Abstract*— Inspection of underwater structures with tethered underwater vehicles is often hindered by the risk of tether entanglement. We propose REACT (real-time entanglement-aware coverage path planning for tethered underwater vehicles), a framework designed to overcome this limitation. REACT comprises a computationally efficient geometry-based tether model using the signed distance field (SDF) map for accurate, real-time simulation of taut tether configurations around arbitrary structures in 3D. This model enables an efficient online replanning strategy by enforcing a maximum tether length constraint, thereby actively preventing entanglement. By integrating REACT into a coverage path planning framework, we achieve safe and entanglement-free inspection paths, previously challenging due to tether constraints. The complete REACT framework's efficacy is validated in a pipe inspection scenario, demonstrating safe navigation and full-coverage inspection. Simulation results show that REACT achieves complete coverage while maintaining tether constraints and completing the total mission 20% faster than conventional planners, despite a longer inspection time due to proactive avoidance of entanglement that eliminates extensive post-mission disentanglement. Real-world experiments confirm these benefits, where REACT completes the full mission, while the baseline planner fails due to physical tether entanglement.**

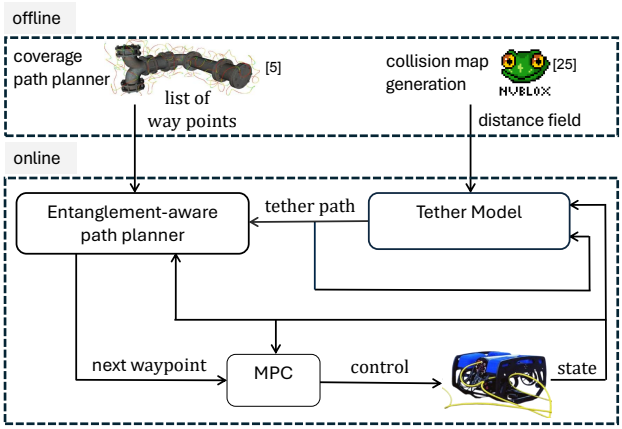


Fig. 1: Overview of real-time entanglement-aware coverage path planning for tethered underwater vehicles (REACT): Offline, a signed distance field (SDF) map is generated from a point cloud, and an off-the-shelf CPP [5] is used to compute a near distance-optimal waypoint sequence. Online, REACT provides an entanglement-free next waypoint to the controller.

I. Introduction

Operating in complex, hazardous, and otherwise inaccessible environments, remotely operated vehicles (ROVs) have become essential for modern exploration and intervention tasks. They enable a diverse range of demanding applications, including surveying, infrastructure inspection, and deep-sea exploration [1]–[3], thereby expanding operational possibilities. Most ROVs are tethered to a host platform to maintain reliable communication and ensure a continuous power supply during long-duration missions. However, this tethering infrastructure introduces operational challenges related to planning and control while posing the risk of being entangled with underwater objects such as flora, fauna, or underwater structures.

Numerous coverage path planner (CPP) algorithms are proposed in the literature for inspection-related tasks with untethered systems. In essence, they calculate approximate distance-optimal paths for a thorough inspection of 3D structures [4]–[6]. In addition, exploration path planners are employed to determine the next points of view to map unknown terrains [7]. However, these path-planning methods are restricted to untethered systems, as they do not account for possible entanglements with the surroundings. Hence, the literature still lacks tether-aware inspection planners.

During operation, entanglement can occur when the vehicle's movement is restricted due to interaction between the tether and objects in the environment. The tether can loop around obstacles, thereby limiting the vehicle's mobility and substantially reducing operational range in a worst-case scenario. Consequently, operators need to carry longer tethers to account for possible entanglements without a proper planner at hand. In this work, we propose REACT that fills this gap in underwater asset inspection [1]. In essence, when integrated into the coverage path planning framework, REACT renders entanglement-free paths, thus ensuring task completion without the need to carry additional tether lengths. Besides, the proposed planning framework reduces overall operational time by circumventing the post-completion detangling process, often required with traditional path-planning methods. The contributions of this work can be summarized as follows:

A. Amer is with the Artificial Intelligence in Robotics Laboratory (AiR Lab), Department of Electrical and Computer Engineering, Aarhus University, 8000 Aarhus C, Denmark (abdelhakim@ece.au.dk); M. Mehndiratta is with NestAI, Espoo, Finland (mohit.mehndiratta@nestai.com); Y. Brodskiy is with EIVA a/s, Skanderborg, Denmark (ybr@eiva.com); B. Wehbe is with Deutsches Forschungszentrum für Künstliche Intelligenz GmbH (DFKI), Bremen, Germany (bilal.wehbe@dfki.de); E. Kayacan is with the Automatic Control Group, Department of Electrical Engineering and Information Technology, Paderborn University, Germany (erdal.kayacan@uni-paderborn.de).

[1]Interactive demo available at https://abdelhakim96.github.io/REACT/

- A computationally efficient tether model that computes the entangled tether configuration.
- An efficient online replanning method that prevents entanglement by incorporating a tether-length constraint.
- A complete entanglement-aware inspection framework integrating CPP, SDF-based mapping, and model predictive control (MPC) to enable safe asset inspections.
- Demonstration of the framework in simulation and real-world tests, showcasing safe and full coverage inspection.

The remainder of this paper is organized as follows: Section II reviews related work. Section III presents the overall framework for tethered underwater inspection. Section IV introduces the taut-tether model, and Section V describes the proposed planner. Experimental results are shown in Sections VI and VII, followed by conclusions in Section VIII.

II. State of the art

The risk of entanglement with obstacles presents a serious challenge for robots with conventional path planners. To systematically address this, a taxonomy of entanglement definitions was presented in [8], where existing interpretations from the literature were cataloged and new definitions were introduced, paving the way for developing new path planning strategies that account for entanglement.

Initial efforts in entanglement-aware path planning focused on 2D environments and relied on offline computation. Notably, [9] introduced the non-entangling traveling salesman problem, formulating the ROV path planning problem as a mixed-integer program that simultaneously determines the waypoint sequence and the path's homotopy class, guaranteeing an entanglement-free trajectory. Similarly, [10] addressed 2D waypoint coverage under tether constraints. While effective for predefined scenarios, these offline approaches lack adaptability in dynamic environments. Similarly, [11] propose a spanning tree-based optimization for 2D coverage.

To address the need for real-time adaptability, subsequent research explored online path planning algorithms, still primarily in 2D. In [12], they introduce a homotopy-augmented topological approach combined with graph search techniques , allowing for dynamic adjustments to the path based on environmental perception. Similarly, in [13], a hybrid A* variant utilizing a modified tangent graph was developed. This method efficiently plans curvature-constrained paths for tethered robots subject to winding angle constraints, demonstrating guarantees and providing simulation results for online entanglement avoidance. Online cell-based decomposition strategies are also proposed to dynamically handle tether constraints and environmental changes in 2D [14].

The complexity of tether-aware path planning further increases when coordinating multiple robots. Early work in this area is presented in [15], followed by the method proposed in [16]. More recently, in [17], an efficient online path planner for multi-robot systems was presented. In this method, a homotopy-based high-level planner was integrated with trajectory optimization and smoothing techniques to generate entanglement-free paths. Similarly, [18] propose a solution to the tethered robot pair motion planning problem for tethered robots in 2D, employing a reduced visibility graph to account for tether interactions with polygonal obstacles. However, despite these advances, these approaches remain constrained to 2D environments. Moreover, while preventive paths to avoid entanglement can be planned, strategies for path planning once tether entanglement has already occurred are not provided.

Real-world applications frequently require navigation in 3D environments, such as with underwater robots. Consequently, in [19] and [20], topological aspects and optimization techniques for 3D tethered navigation were explored. In [21], a 3D exploration path planner incorporating explicit contact avoidance constraints for the tether was presented, facilitating safer navigation for single tethered robots in complex 3D spaces.

The increased complexity of 3D multi-robot scenarios was addressed in [22], where previous 2D work was extended to 3D. Further advances were introduced in [23] and [24], where path planning strategies explicitly considered the topological constraints imposed by multiple interacting tethers in 3D were proposed. While these methods advance the state of the art in multi-robot coordination, they are generally designed for offline computation and are not suited for online planning where real-time performance is needed.

In summary, existing path planners that account for tether constraints often face limitations for practical online CPP in complex 3D settings. Many are too computationally intensive for real-time use [10], [22]–[24], lack integrated tether-aware CPP frameworks, or rely on simplifying assumptions such as 2D environments or basic obstacle shapes [12], [13], [17], hindering generalization to real-world inspection tasks. To address these limitations, REACT is proposed, which enables real-time, entanglement-aware path planning in arbitrary 3D environments.

III. Overall REACT Framework

REACT framework, as depicted in Fig. 1, consists of two main components: an offline planning phase and an online execution phase. In the offline phase, the environment is provided as a point cloud, including the object to be inspected. Then, a SDF map is generated from this point cloud using the nvblox library [25]. Subsequently, the extracted point cloud of the structure under inspection is processed via FC-Planner [5] to compute an optimal waypoint sequence (referred to as nominal waypoints), generating a path that renders full inspection coverage while disregarding the tether constraints.

In the online phase, tether constraints are managed by an entanglement-aware replanner that ensures that the maximum tether length is not exceeded. The proposed REACT framework incorporates a modular online layer that can be seamlessly integrated with any existing offline coverage or path planner. By decoupling coverage planning from tether-constrained replanning, the framework transforms an otherwise highly complex and computationally intractable joint problem into two tractable subproblems: offline coverage planning and fast online tether management. This separation

not only enables the use of standard off-the-shelf coverage planners, but also greatly simplifies integration with other planning frameworks.

Operating online provides further advantages: it allows the system to handle operational disturbances and model uncertainties that cannot be fully anticipated during offline planning. Moreover, it naturally supports scenarios where the map expands during operation, such as in exploration or when new obstacles are detected, enabling adaptive planning in previously unknown environments. The online management-aware planner then provides entanglement-free reference trajectories that are tracked by a MPC controller, which translates them into optimal wrench commands for the ROV. This ensures that the vehicle follows the re-planned paths while respecting dynamics and actuation limits.

IV. Tether Modeling

In this section, we describe a computationally efficient, geometry-based tether model for tethered underwater vehicles. This model predicts the tether path, specifically the positions of each node along the tether's length, based on the ROV's trajectory, and assumes a geometry-based constraint where the tether remains taut and fully stretched at all times. The main idea of the proposed tether model is inspired by the shortcutting algorithm of the ropeRRT path planner [26], which simplifies sampled trajectories similar to a rope tightening around an obstacle.

A. Tether model description

To efficiently compute the tether configuration around obstacles, the continuous tether is discretized into a finite set of nodes, separated by a fixed resolution δ. Each node represents a point along the tether and collectively approximates the overall 3D shape of the tether. This discretization enables computationally efficient collision checking and shortcutting.

Let the tether path at time t be denoted by $\mathbf{P}_{\text{tether}}(t) = \{\mathbf{p}_i(t)\}_{i=1}^{n}$, where each node $\mathbf{p}_i(t) \in \mathbb{R}^3$ represents the position of the i-th node in 3D space at time t, and n is the total number of nodes in the tether path. The ROV position at time t, denoted by $\mathbf{p}_{\text{rov}}(t) \in \mathbb{R}^3$, is appended at the end $\mathbf{p}_{n+1}(t)$. The proposed tether model then iteratively computes the equivalent taut-tether path through sequential shortcutting operation. Fig. 2 illustrates an example of the shortcut operation. Starting at the last node ($\mathbf{p}_j(t) = \mathbf{p}_n(t)$), the algorithm attempts to shortcut the path segment between each pair of nodes $(\mathbf{p}_i(t), \mathbf{p}_j(t))$ where $i > j$. If the line of sight between $\mathbf{p}_i(t)$ and $\mathbf{p}_j(t)$ is collision-free, as determined via an SDF map ($\mathcal{M}_{sdf}$), the intermediate nodes are replaced with a straight segment sampled at known resolution δ. Conversely, if the line of sight encounters a collision, j shifts to the preceding node, and the collision-free line of sight is rechecked. This process is repeated until a collision-free line of sight is found, and the intermediate nodes are then replaced.

Finally, after the shortcutting step, a pulling operation is applied to each node, moving it incrementally toward the tether endpoint $\mathbf{p}_{n+1}(t)$. This ensures that nodes are not left stuck in the cavities of non-convex obstacles. The shortcutting and pulling operations are applied iteratively until convergence, resulting in a taut and collision-free tether path $\mathbf{P}_{\text{tether}}(t+1)$. The full procedure is described in Algorithm 1.

Algorithm 1: Taut-tether model

Input : $\mathbf{p}_{\text{rov}}(t)$, $\mathbf{P}_{\text{tether}}(t)$, $\mathcal{M}_{sdf}$, δ
Return: $\mathbf{P}_{\text{tether}}(t+1)$

```
appendPath(P_tether(t), p_rov(t));
for i ← len(P_tether(t)) − 1 to 0 do
    for j ← i − 1 to 0 do
        if checkShortcut(P_tether(t), i, j) then
            replaceNodes(P_tether(t), i, j, δ);
        else
            if not checkLineOfSight (M_sdf,
              P_tether(t)) then
                break;
            if isInCollision(M_sdf, P_tether(t)[j]) then
                pullNode(P_tether(t)[j],
                  P_tether(t).end(), δ);
return P_tether(t + 1);
```

Remark 1: Sparse environment reconstructions may create gaps in the SDF, causing obstacles to appear partially free. This can lead to unrealistic tether shortcutting through obstacles unless sufficient voxel resolution is applied.

Remark 2: The tether discretization resolution δ should be chosen consistently with the SDF voxel resolution to ensure reliable collision detection.

V. Entanglement-Aware Path Planner

Next, we present the entanglement-aware path planner, a local planner designed to address the real-time entanglement avoidance problem for tethered underwater vehicles. The planner continuously monitors the tether configuration and dynamically adjusts the vehicle's target to prevent the tether length from exceeding a specified maximum allowable length, $L_{\max}$, while ensuring safe navigation toward a series of predefined reference waypoints.

At each time step t, the planner receives the current estimated tether path, $\mathbf{P}_{\text{tether}}(t)$, from the tether model. It also maintains an ordered list of target waypoints, $\mathbf{W} = \{\mathbf{p}_{\text{waypoint}}(k)\}_{k=1}^{m}$, where each $\mathbf{p}_{\text{waypoint}}(k)$ specifies a 3D position the ROV must reach sequentially. The current position of the ROV, $\mathbf{p}_{\text{rov}}(t)$, and the current waypoint index, k, are also maintained as part of the planner's state. The current tether length $L_{\text{tether}}(t)$ is computed from the tether path and compared against the maximum allowable tether length $L_{\max}$ to determine whether replanning is needed.

The planner operates in two main modes: normal mode and recovery mode. In normal mode, when the tether length is within the allowable limit, the planner sets the ROV's target position directly to the current waypoint. It continuously

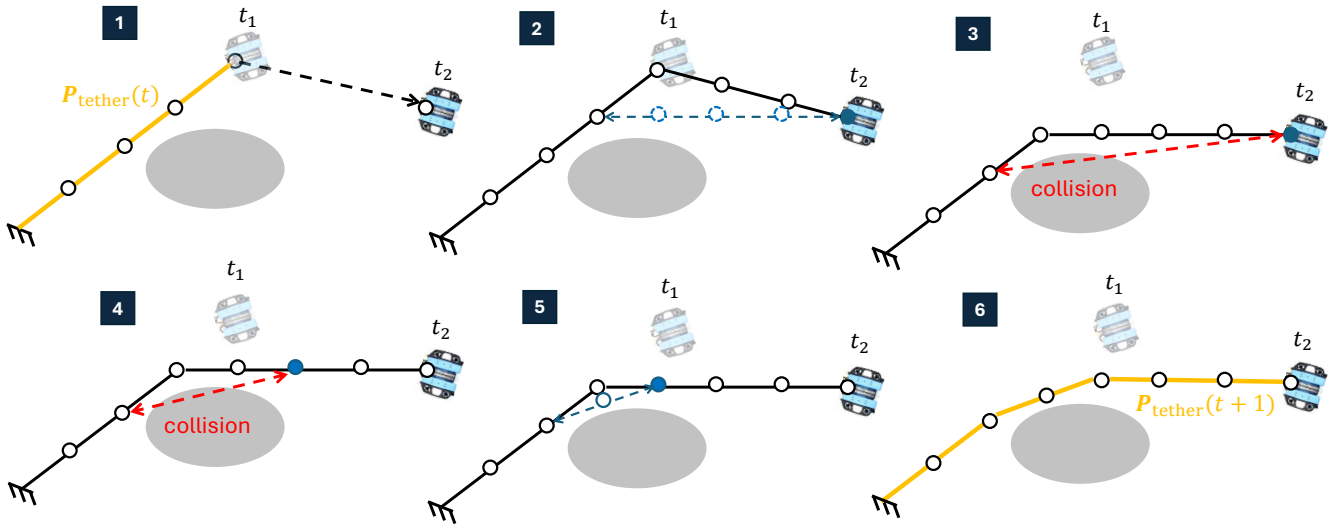


Fig. 2: Tether shortcutting during ROV motion from t_1 to t_2. (1) Initial tether with new ROV position $\mathbf{p}_{\text{rov}}$ appended; (2) Successful shortcut from the end node; (3) Collision encountered when attempting further shortcutting, skipping to the next node; (4) Another collision detected from the new node; (5) Successful shortcut from a subsequent node; (6) Final tether configuration (yellow) after applying all feasible shortcuts.

monitors the ROV's position and, upon detecting that the waypoint has been reached, advances the waypoint index to the next target in the sequence. This mode prioritizes nominal waypoint following behavior.

If the tether length exceeds the limit, the planner switches to recovery mode to avoid entanglement. In this mode, a recovery path is generated by searching for an alternative safe path toward the current waypoint that respects the tether length constraint. The ROV's target position, $\mathbf{p}_{\text{target}}$, is then updated to incrementally follow this recovery path. The planner remains in recovery mode until the end of the recovery path is reached, at which point it switches back to normal mode to continue waypoint tracking.

This dual-mode approach ensures that the vehicle maintains a safe tether configuration while effectively progressing through its mission waypoints. By continuously switching between nominal waypoint tracking and recovery behaviors in response to tether length measurements, the planner enables real-time, entanglement-aware path planning. The overall planning framework is summarized in Algorithm 2.

A. Disentanglement path search

To implement the disentanglement behavior described above, the recovery path search algorithm (Algorithm 3) is employed. This algorithm takes as input the current tether path $\mathbf{P}_{\text{tether}}(t)$, the current target waypoint $\mathbf{W}[k]$, and the maximum allowable tether length $L_{\max}$. Its goal is to generate a recovery path $\mathbf{P}_{\text{recovery}}$ that guides the ROV safely toward the waypoint while respecting the tether length constraint.

The algorithm performs a backward search along the tether path, starting from the ROV's current position at the tether's end and progressing toward the tether starting node ($\mathbf{p}_0$). At each iteration, a candidate pivot point $\mathbf{p}_{\text{pivot}}$ on the tether is selected as a potential point from which the ROV can attempt an alternative route toward the target waypoint.

For each pivot, two path segments are constructed. The first segment, $\mathbf{P}_{\text{r1}}$, corresponds to the reversed portion of the tether from the pivot $\mathbf{p}_{\text{pivot}}$ to the ROV's current position, representing the retraced section of the tether. The second segment, $\mathbf{P}_{\text{s}(n-i)}$, is a newly planned shortest path connecting the pivot to the target waypoint $\mathbf{W}[k]$, which can be computed using sampling-based methods such as RRT*. These segments are then concatenated with the tether path from $\mathbf{p}_0$ up to the pivot, forming an augmented candidate path $\mathbf{P}_{\text{aug}}$, which is used to simulate the tether configuration and estimate its length L_{tether} at the next time step. The total tether length L_{tether} is then compared against the maximum allowable length $L_{\max}$. If the length constraint is satisfied, the recovery path $\mathbf{P}_{\text{recovery}} = \mathbf{P}_{\text{r1}} \cup \mathbf{P}_{\text{s}(n-i)}$ is considered feasible.

If no feasible path is found after examining all possible pivot points, the algorithm defaults to a fallback strategy where the tether length constraint is treated as a soft limit. In this case, the recovery path consists solely of the shortest path from the ROV's current position directly to the target waypoint, temporarily disregarding tether length constraints. The resulting recovery path provides a safe and viable trajectory for the ROV to follow during disentanglement operations by combining the reversed tether segment with the newly planned path to the waypoint, as shown in Fig. 3.

B. Recovery path refinement

Once the recovery path $\mathbf{P}_{\text{recovery}}$ is generated through the disentanglement search, it undergoes a refinement process to generate a safe path $\mathbf{P}_{\text{safe}}$ with obstacle clearance δ_{obst} and improved smoothness prior to execution. This refinement is performed iteratively and consists of three consecutive

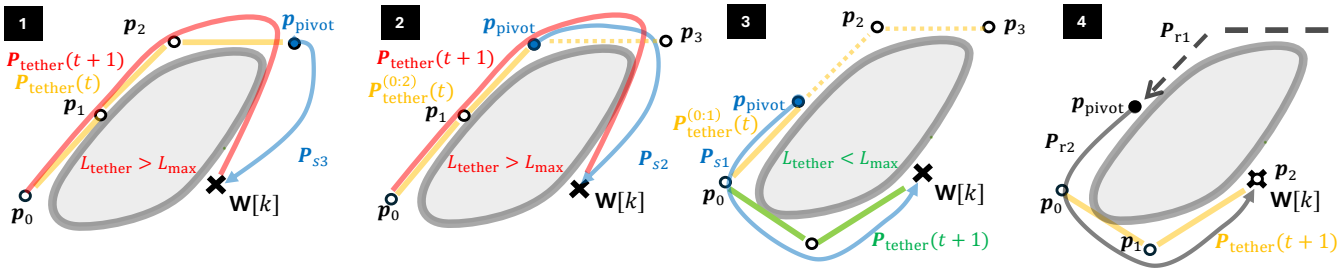


Fig. 3: Disentanglement path search. At time t, the tether configuration is $\mathbf{P}_{\text{tether}}(t) = \{\mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3\}$. (1) The shortest path $\mathbf{P}_{s3}$ from $\mathbf{p}_3$ to the waypoint $\mathbf{W}[k]$ is appended to the tether path, forming the augmented path $\mathbf{P}^{(0:3)}_{\text{tether}} \cup \mathbf{P}_{s3}$, which is passed to the tether model to compute $\mathbf{P}_{\text{tether}}(t+1)$. The resulting tether length exceeds $L_{\max}$, indicating a violation. (2) The pivot shifts to $\mathbf{p}_2$, and the augmented path $\mathbf{P}^{(0:2)}_{\text{tether}} \cup \mathbf{P}_{s2}$ is evaluated, but still violates the constraint. (3) With pivot at $\mathbf{p}_1$, the augmented path $\mathbf{P}^{(0:1)}_{\text{tether}} \cup \mathbf{P}_{s1}$ yields a feasible configuration $\mathbf{P}_{\text{tether}}(t+1)$ with $L_{\text{tether}} \leq L_{\max}$. (4) The recovery path $\mathbf{P}_{\text{recovery}}$ is thus executed, consisting of $\mathbf{P}_{\text{r1}}$, tracing back from $\mathbf{p}_3$ to $\mathbf{p}_1$, and $\mathbf{P}_{\text{r2}} = \mathbf{P}_{s1}$, leading to the waypoint $\mathbf{W}[k]$.

Algorithm 2: Entanglement-aware path planner

Input : $\mathbf{W}$, $L_{\max}$, $\mathbf{P}_{\text{tether}}(t)$, $\mathbf{p}_{\text{rov}}(t)$
Return: $\mathbf{p}_{\text{target}}$

$mode \leftarrow$ NORMAL;
$\mathbf{P}_{\text{recovery}} \leftarrow \emptyset$;
$k \leftarrow 0$;
$L_{\text{tether}}(t) \leftarrow \text{computeLength}\big(\mathbf{P}_{\text{tether}}(t)\big)$;
if $mode =$ NORMAL **then**
  if $L_{\text{tether}}(t) > L_{\max}$ **then**
    $mode \leftarrow$ RECOVERY;
    $\mathbf{P}_{\text{recovery}} \leftarrow$ disEntanglementSearch$\big(\mathbf{P}_{\text{tether}}(t), \mathbf{W}[k]\big)$;
    $\mathbf{p}_{\text{target}} \leftarrow \text{followPath}\big(\mathbf{P}_{\text{recovery}}, \mathbf{p}_{\text{rov}}(t)\big)$;
  else
    $\mathbf{p}_{\text{target}} \leftarrow \mathbf{W}[k]$;
    if reachedWaypoint$\big(\mathbf{p}_{rov}(t), \mathbf{W}[k]\big)$ **then**
      $k \leftarrow k+1$;
else if $mode =$ RECOVERY **then**
  $\mathbf{p}_{\text{target}} \leftarrow \text{followPath}\big(\mathbf{P}_{\text{recovery}}, \mathbf{p}_{\text{rov}}(t)\big)$;
  if reachedEndOfPath$\big(\mathbf{p}_{rov}(t), \mathbf{P}_{safe}\big)$ **then**
    $mode \leftarrow$ NORMAL;
return $\mathbf{p}_{\text{target}}$;

Algorithm 3: Disentanglement path search

Input : $\mathbf{P}_{\text{tether}}(t), \mathbf{W}[k], L_{\max}$
Return: Recovery path $\mathbf{P}_{\text{recovery}}$

$found\_feasible \leftarrow$ **False**;
$n \leftarrow \text{len}\big(\mathbf{P}_{\text{tether}}(t)\big)$;
for $i \leftarrow n-1$ downto 0 **do**
  $\mathbf{p}_{\text{pivot}} \leftarrow \mathbf{P}_{\text{tether}}(t)[i]$;
  $\mathbf{P}_{\text{r1}} \leftarrow \text{reverseSegment}\big(i, n-1, \mathbf{P}_{\text{tether}}(t)\big)$;
  $\mathbf{P}_{\text{s}(n-i)} \leftarrow \text{planShortestPath}\big(\mathbf{p}_{\text{pivot}}, \mathbf{W}[k]\big)$;
  $\mathbf{P}_{\text{aug}} \leftarrow \text{concatenate}\big(\mathbf{P}_{\text{tether}}(t)[0:i], \mathbf{P}_{\text{s}(n-i)}\big)$;
  $\mathbf{P}_{\text{tether}}(t+1) \leftarrow \text{tetherModel}\big(\mathbf{P}_{\text{aug}}\big)$;
  $L_{\text{tether}} \leftarrow \text{computeLength}\big(\mathbf{P}_{\text{tether}}(t+1)\big)$;
  if $L_{tether} \leq L_{\max}$ **then**
    $\mathbf{P}_{\text{r2}} \leftarrow \mathbf{P}_{\text{s}(n-i)}$;
    $found\_feasible \leftarrow$ **True**;
    break;
if *not* $found\_feasible$ **then**
  $\mathbf{p}_{\text{pivot}} \leftarrow \mathbf{P}_{\text{tether}}(t)[-1]$;
  $\mathbf{P}_{\text{r1}} \leftarrow \emptyset$;
  $\mathbf{P}_{\text{r2}} \leftarrow \text{planShortestPath}\big(\mathbf{p}_{\text{pivot}}, \mathbf{W}[k]\big)$;
$\mathbf{P}_{\text{recovery}} \leftarrow \mathbf{P}_{\text{r1}} \cup \mathbf{P}_{\text{r2}}$;
return $\mathbf{P}_{\text{recovery}}$;

operations: centroid offset, stochastic local perturbation, and polynomial smoothing. The resulting safe trajectory $\mathbf{P}_{\text{safe}}$ is validated to ensure collision-free execution.

The centroid offsetting step is designed to increase clearance from obstacles that may lie near the geometric center of the path. The geometric centroid of the path is calculated as the mean of all points, $\mathbf{c} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{p}_i$, where N denotes the number of points in $\mathbf{P}_{\text{recovery}}$ and $\mathbf{p}_i$ represents the i-th point. Each point is then displaced along the normalized vector that points from the centroid to the point, $\mathbf{d}_i = (\mathbf{p}_i - \mathbf{c})/\|\mathbf{p}_i - \mathbf{c}\|$, producing the updated position $\mathbf{p}'_i = \mathbf{p}_i + \delta_{obst}\mathbf{d}_i$. This operation effectively increases the clearance from the obstacles.

Following centroid offsetting, the path is further refined using stochastic local perturbations. Each intermediate point along the path (excluding the endpoints) is locally adjusted by sampling random directions uniformly on the surface of a sphere and displacing the point along the sampled direction by a small distance. The resulting candidate points are evaluated for collisions and only those that are collision-free are accepted. Multiple trials are allowed for each point to maximize the likelihood of finding a feasible local adjustment. This perturbation mechanism adjusts the path points to locally increase clearance from nearby obstacles, ensuring

the trajectory remains safely away from potential collisions while preserving its overall structure.

The final stage of refinement involves smoothing the path using a third-order polynomial fitting. The path is divided into overlapping segments and for each segment separate cubic polynomials are fitted to the x, y, and z coordinates of the points using a least squares approximation. Each point in the segment is then replaced by the corresponding evaluation of the polynomial functions, producing a smoother trajectory. The overlapped segments ensure continuity between the polynomial pieces and prevent abrupt changes in curvature, resulting in a dynamically feasible path suitable for the ROV. After completion of these operations, the refined path is subjected to collision verification. If any collisions are detected, the refinement process is repeated iteratively until a collision-free trajectory is obtained.

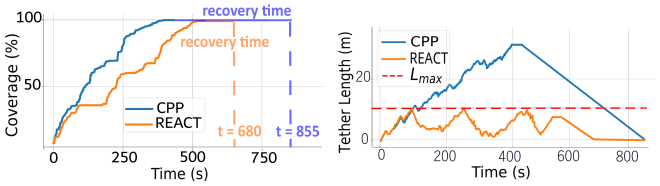
VI. Simulations Experiments

This section presents the simulation results for the proposed REACT method. The entire framework is implemented in C++ to ensure computational efficiency and is integrated with ROS to facilitate modularity and deployment on real-world robots. Furthermore, the open motion planning (OMPL) library [27] is utilized to compute the shortest paths.

A. Simulation experimental setup

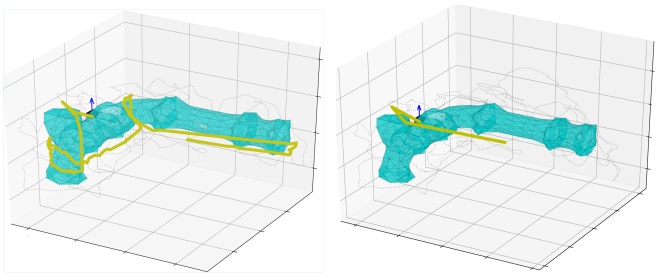
The path planner is implemented for a BlueROV2 underwater robot. An MPC approach accounts for model constraints and provides the optimal control input $\mathbf{u} \in \mathbb{R}^4$, where $\mathbf{u} = [F_x, F_y, F_z, M_z]^T$ represents the forces in the X, Y, and Z directions, and the moment about the Z-axis. The goal is to follow the desired reference trajectory $\mathbf{x}^{\text{ref}} \in \mathbb{R}^4$, where $\mathbf{x}^{\text{ref}} = [x^{\text{ref}}, y^{\text{ref}}, z^{\text{ref}}, \psi^{\text{ref}}]^T$ contains the reference position $(x^{\text{ref}}, y^{\text{ref}}, z^{\text{ref}})$ and the reference yaw angle ψ^{ref}. For further details about the controller and model, the reader is referred to [28]. The entanglement-aware path planner provides $\mathbf{x}^{ref}$ in real-time for the MPC.

We perform a comparative analysis of the proposed REACT method and a baseline conventional CPP (FC-Planner [5]), which does not explicitly handle entanglement. The simulation setup uses a 1/10-scale pipe structure based on the model described in [5]. The simulated onboard camera has a 70-degree field of view. The tether constraint is implemented by specifying a maximum allowable tether length of $L_{\text{max}} = 10$m. Note that the resulting planned trajectories are geometrically identical to those in the full-scale environment, simply scaled down proportionally. The primary motivation for using the scaled model is to accelerate simulation runs while preserving the validity of the results. Coverage is computed in a manner similar to [6]. At each time step, the position and orientation of the camera are used to determine which triangles in the environment mesh are visible. A triangle is marked as visible if its centroid is within the inspection range, its surface normal faces the camera, and its projection lies within the camera's field of view. Over time, the set of all uniquely observed triangles accumulates. The coverage at time t is defined as the ratio of the number of unique visible triangles up to time t to the total number of triangles in the environment.



(a) Coverage vs. time. (b) Tether Length vs. time.

Fig. 4: Comparison of coverage and tether length.



(a) CPP final tether path. (b) REACT final tether path.

Fig. 5: 3D views of final trajectories: (a) CPP results in entangled tether path, (b) REACT yields a non-entangled tether path, reflecting effective entanglement avoidance.

B. Simulation results

The performance metrics for both planners are summarized in Table I and Table II, while Figs. 4 and 5 visualize the performance of the planners in terms of environmental coverage, tether length behavior, and final trajectory configurations. The evaluation focuses on comparing mission efficiency, constraint compliance, and safety aspects between REACT and the conventional CPP.

The results present two phases: the inspection phase and the recovery phase, where the recovery time represents the estimated duration required to return to the starting position after complete inspection while disentangling the entire tether. The results highlight distinct trade-offs between the two planning strategies. As shown in Table I, the CPP method achieves a shorter inspection time (429s vs. 546s) due to its straightforward path execution without rerouting for disentanglement. However, focusing solely on inspection

TABLE I: Coverage performance comparison between REACT and CPP baseline showing inspection time, recovery time, total mission duration, and final coverage.

Planner	Inspection time (s)	Recovery time (s)	Total time (s)	Coverage (%)
REACT	546	134	**680**	**99.91**
CPP	429	426	855	99.82

TABLE II: Comparison of tether constraint compliance.

Planner	Maximum tether length (m)	Duration of length exceedance (s)
CPP	31.16	327.37
REACT	**10.52**	**10.36**

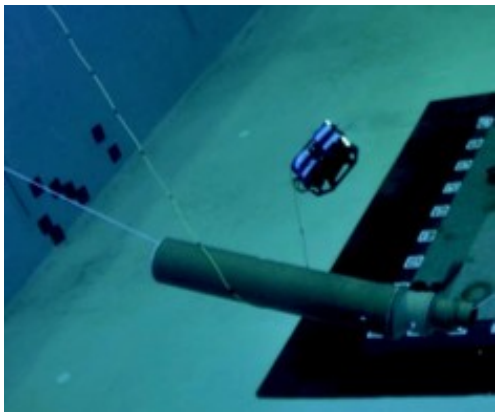


Fig. 6: During the experiment, the ROV became entangled with the pipe, illustrating a typical CPP entanglement.

speed overlooks the critical aspect of tether management in constrained environments. The CPP exhibits a significantly longer total mission time (855s vs. 680s) because extensive disentanglement is required after inspection completion.

REACT demonstrates multiple instances of entanglement detection and resolution, as evidenced by the peaks in the tether length curve in Fig. 4b and the corresponding flat regions in the coverage curve in Fig. 4a, where the ROV inspection progress stops to reroute and find paths without entanglement. This reactive replanning behavior directly results in a longer inspection time, as the system prioritizes tether safety over raw speed. In particular, the limit on the length of the tether is rarely exceeded by REACT, as shown in Table II and Fig. 4b. Conversely, the CPP method severely violates this constraint, reaching 31.16m for extended periods (327.37s), risking unrecoverable tether entanglement. The 3D trajectory visualizations in Fig. 5 further illustrate this difference, showing the entangled tether geometry resulting from CPP versus the non-entangled configuration achieved by REACT. Therefore, REACT demonstrates superior performance for safe tethered ROV inspection.

VII. Real-world Experiments

Real-world experiments were conducted at $23\text{m}\times19\text{m}\times8\text{m}$ test basin equipped with a 12-camera Qualisys motion capture system. The experimental platform is a BlueROV2, configured identically to the simulation setup. The test environment consists of a pipe structure with a diameter of 0.7m and a length of 2.5m, as shown in Fig. 6. Two planning modes, identical to those evaluated in simulation, are tested. The first is REACT, which represents the full system incorporating REACT. The second is the CPP baseline without REACT. In both modes, a helical trajectory is given as a reference.

A. Real-world experimental results

Real-world tests illustrate the need for REACT compared to the classic CPP approach. Figure 7 shows the mission trajectories for both methods. As such, the baseline CPP planner failed to complete the inspection task. At $t = 124$ s, the tether entangles the pipe structure, stopping the ROV from further progress. Hence, the overall coverage achieved is only 80.9% (Table III).

TABLE III: Coverage Performance in Real-world Trials. REACT takes a maximum of 0.4822 s (per re-planning step) during the inspection to compute the fully replanned path.

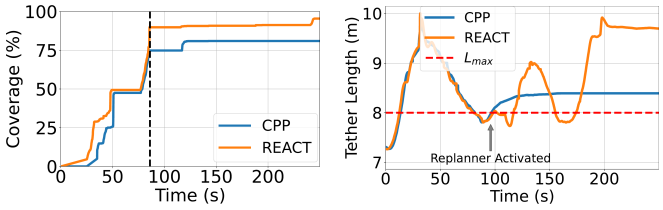
Planner	Inspection	Coverage (%)	Computation overhead (s)
REACT	Full	95.6	0.4822
CPP	Failed (Entangled)	80.9	N/A

The proposed REACT planner completed the full mission, achieving 95.6% coverage. The planner adapted to tether constraints throughout the mission. Unlike the simulation results, the soft tether length limit of the 8m was exceeded in real-world tests. As such, when no entanglement-free paths to the goal are available, REACT treats the length constraints as a soft limit rather than a hard limitation. However, when entanglement is detected and there is an entanglement-free path around the pipe, REACT finds a path that detangles the ROV from the structure. This capability is demonstrated at $t = 96$s when REACT detected an imminent tether entanglement. The planner computed a new path that safely untangled the ROV from the pipe structure, allowing inspection to continue. The system only exceeded the soft tether limit when necessary for mission completion while prioritizing entanglement avoidance whenever possible.

The baseline CPP planner, lacking entanglement-awareness, became permanently stuck (see Fig. 6). This demonstrates a key advantage of REACT that was not visible in the simulation study, where tether forces were not modeled. Although the REACT adds computation compared to the baseline, the maximum replanning time of 0.4822 s remained well within the onboard platform's capabilities, allowing real-time operation. This overhead, while higher than the conventional CPP approach, was low enough to enable timely adaptation to tether entanglement events. These results demonstrate that standard planners without tether constraints fail even during relatively simple underwater 3D inspection tasks. While the baseline initially followed the planned path, it became entangled and subsequently immobilized midway through the mission. On the other hand, REACT's ability to detect and avoid entanglement through replanning enabled a successful mission completion. These findings highlight the need for entanglement-aware planning in real underwater operations.

VIII. Conclusion

We present REACT, a real-time, entanglement-aware path planning framework for tethered underwater vehicles that supports underwater asset inspection tasks. Through comparative simulation and real-world experiments, we demonstrate that conventional planners face tether entanglement issues, rendering incomplete missions, as they cannot account for tether constraints. In simulation, REACT shows effective tether management while maintaining full coverage, achieving a shorter total mission time (680s vs. 855s) despite



(a) Coverage vs. time for both planners.

(b) Tether length vs. time for both planners.

Fig. 7: Real-world ROV trajectories. Without REACT, the mission fails due to tether entanglement. With REACT, the ROV completes the mission successfully. At $t = 96$s, the replanner is activated, allowing REACT to disentangle the ROV and continue the inspection. In contrast, at $t = 124$s, the ROV using the conventional CPP becomes stuck.

longer inspection time due to its proactive entanglement avoidance that eliminates the need for extensive post-mission disentanglement. The real-world experiments confirm these findings, where the baseline CPP planner achieves only 80.9% coverage before becoming physically stuck due to tether entanglement, while REACT completes the full mission with 95.6% coverage through adaptive replanning.

Acknowledgment

This work was supported by Innovation Fund Denmark (Grant No. 2040-00032B) and EIVA A/S. The authors would like to thank Nikolas Dahn, Tom Slawik, and Dr. Leif Christensen from DFKI, as well as Adis Hodzic from EIVA, for their experimental support. We also thank Dr. Andriy Sarabakha, Dr. Muqing Cao, Dr. Louis Petit, and Dr. Olaya Tuñón for their valuable and insightful discussions.

References

[1] A. Amer, O. Álvarez-Tuñón, H. İ. Uğurlu, J. L. F. Sejersen, Y. Brodskiy, and E. Kayacan, "Unav-sim: A visually realistic underwater robotics simulator and synthetic data-generation framework," in *2023 21st International Conference on Advanced Robotics (ICAR)*. IEEE, 2023, pp. 570–576.

[2] A. Amer, D. Felsager, Y. Brodskiy, and A. Sarabakha, "Modelling of Underwater Vehicles using Physics-Informed Neural Networks with Control," 2025. [Online]. Available: https://doi.org/10.48550/arXiv.2504.20019

[3] A. H. K. S. A. Amer, "Autonomous marine robots optimal control and path planning with data-driven models," Ph.D. dissertation, AARHUS UNIVERSITY, 2025.

[4] A. Bircher, K. Alexis, M. Burri, P. Oettershagen, S. Omari, T. Mantel, and R. Siegwart, "Structural inspection path planning via iterative viewpoint resampling with application to aerial robotics," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 6423–6430.

[5] C. Feng, H. Li, M. Zhang, X. Chen, B. Zhou, and S. Shen, "Fc-planner: A skeleton-guided planning framework for fast aerial coverage of complex 3d scenes," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 8686–8692.

[6] A. Amer, M. Mehndiratta, J. le Fevre Sejersen, H. X. Pham, and E. Kayacan, "Visual Tracking Nonlinear Model Predictive Control Method for Autonomous Wind Turbine Inspection," in *2023 21st International Conference on Advanced Robotics (ICAR)*. IEEE, 2023, pp. 431–438.

[7] T. Dang, M. Tranzatto, S. Khattak, F. Mascarich, K. Alexis, and M. Hutter, "Graph-based subterranean exploration path planning using aerial and legged robots," *Journal of Field Robotics*, vol. 37, no. 8, pp. 1363–1388, 2020.

[8] G. Battocletti, D. Boskos, D. Toliæ, I. Palunko, and B. De Schutter, "Entanglement definitions for tethered robots: Exploration and analysis," *IEEE Access*, 2024.

[9] S. McCammon and G. A. Hollinger, "Planning and executing optimal non-entangling paths for tethered underwater vehicles," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 3040–3046.

[10] L. Mechsy, M. Dias, W. Pragithmukar, and A. Kulasekera, "A novel offline coverage path planning algorithm for a tethered robot," in *2017 17th International Conference on Control, Automation and Systems (ICCAS)*. IEEE, 2017, pp. 218–223.

[11] X. Peng, F. Schwarzentruber, O. Simonin, and C. Solnon, "Spanning-tree based coverage for a tethered robot," *IEEE Robotics and Automation Letters*, 2025.

[12] S. Kim, S. Bhattacharya, and V. Kumar, "Path planning for a tethered mobile robot," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 1132–1139.

[13] V. S. Chipade, R. Kumar, and S. Z. Yong, "WiTHy A*: Winding-Constrained Motion Planning for Tethered Robot using Hybrid A*," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 8771–8777.

[14] R. H. Teshnizi and D. A. Shell, "Computing cell-based decompositions dynamically for planning motions of tethered robots," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 6130–6135.

[15] S. Hert and V. Lumelsky, "The ties that bind: Motion planning for multiple tethered robots," *Robotics and autonomous systems*, vol. 17, no. 3, pp. 187–215, 1996.

[16] X. Zhang and Q.-C. Pham, "Planning coordinated motions for tethered planar mobile robots," *Robotics and Autonomous Systems*, vol. 118, pp. 189–203, 2019.

[17] M. Cao, K. Cao, S. Yuan, T.-M. Nguyen, and L. Xie, "Neptune: Nonentangling trajectory planning for multiple tethered unmanned vehicles," *IEEE Transactions on Robotics*, vol. 39, no. 4, pp. 2786–2804, 2023.

[18] R. H. Teshnizi and D. A. Shell, "Motion planning for a pair of tethered robots," *Autonomous Robots*, vol. 45, no. 4, pp. 503–521, 2021. [Online]. Available: https://doi.org/10.1007/s10514-021-09972-x

[19] S. Bhattacharya, M. Likhachev, and V. Kumar, "Topological constraints in search-based robot path planning," *Autonomous Robots*, vol. 33, pp. 273–290, 2012.

[20] S. Martinez-Rozas, D. Alejo, F. Caballero, and L. Merino, "Optimization-based trajectory planning for tethered aerial robots," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 362–368.

[21] L. Petit and A. L. Desbiens, "TAPE: Tether-Aware Path Planning for Autonomous Exploration of Unknown 3D Cavities Using a Tangle-Compatible Tethered Aerial Robot," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 10 550–10 557, 2022.

[22] S. Hert and V. Lumelsky, "Motion planning in r/sup 3/for multiple tethered robots," *IEEE transactions on robotics and automation*, vol. 15, no. 4, pp. 623–639, 1999.

[23] A. Patil, M. Park, and J. Bae, "Coordinating tethered autonomous underwater vehicles towards entanglement-free navigation," *Robotics*, vol. 12, no. 3, p. 85, 2023.

[24] M. Cao, K. Cao, S. Yuan, K. Liu, Y. L. Wong, and L. Xie, "Path planning for multiple tethered robots using topological braids," *arXiv preprint arXiv:2305.00271*, 2023.

[25] A. Millane, H. Oleynikova, E. Wirbel, R. Steiner, V. Ramasamy, D. Tingdahl, and R. Siegwart, "nvblox: Gpu-accelerated incremental signed distance field mapping," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 2698–2705.

[26] L. Petit and A. L. Desbiens, "RRT-Rope: A deterministic shortening approach for fast near-optimal path planning in large-scale uncluttered 3d environments," in *2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2021, pp. 1111–1118.

[27] I. A. Sucan, M. Moll, and L. E. Kavraki, "The open motion planning library," *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, pp. 72–82, 2012.

[28] A. Amer, M. Mehndiratta, Y. Brodskiy, and E. Kayacan, "Empowering Autonomous Underwater Vehicles Using Learning-Based Model Predictive Control With Dynamic Forgetting Gaussian Processes," *IEEE Transactions on Control Systems Technology*, 2025.