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

A coverage motion planning approach for UVMS-based propeller cleaning in obstacle-occluded environments

Raksi Kopo, Fotis Panetsos*, Spyridon G. Tarantos*, Kostas J. Kyriakopoulos

Abstract— **This work addresses the problem of underwater propeller cleaning in environments containing obstacles using an Underwater Vehicle Manipulator System (UVMS). Prior propeller-cleaning approaches plan coverage tool paths without explicitly considering the connectivity of the associated Surface-Constrained Configuration Space (SCCS), leading to unnecessary lift-offs in obstacle-occluded settings. In contrast, we formulate the coverage problem in the disconnected SCCS as a Generalized Traveling Salesman Problem (GTSP) within a hierarchical framework, accounting for obstacles and attempting to minimize the number of tool lift-offs. We consider explicitly the tool lift-off paths in the GTSP cost formulation, utilizing the hierarchical framework to guide the search without exhaustively evaluating all possible paths. To achieve smoother tool paths with fewer turns, we introduce a cost that promotes alignment with desired coverage curves. Finally, we time-parameterize the coverage path into a whole-body UVMS trajectory by minimizing the duration of the cleaning task, while respecting the robot hardware limitations. The effectiveness of the proposed method is demonstrated in a realistic simulation scenario.**

I. Introduction

Maintaining the propeller of marine vessels clean from marine biofouling [1] is essential for increasing its lifetime [2] as well as the overall efficiency of the vessel [3], while reducing its ecological footprint. The constraints associated with a human-diver-based cleaning operation, i.e., high cost, limited diving times, and safety risks, promote the adoption of automated solutions.

Hull-crawling robots [4] are already used for replacing divers in cleaning the relatively flat surfaces of marine vessels [5], [6]. However, the highly curved geometry of the propeller, as well as its confined surroundings that usually consist of the propeller shaft, ship rudder, and ship hull, dictate increased dexterity. Robot manipulators equipped with cleaning tools such as polishing disks and brushes [7] can carry out such a dexterous task [8], [9]. However, their fixed base limits their effectiveness. To tackle this issue, [10] employs an Underwater Vehicle Manipulator System (UVMS) for cleaning the propeller of a deployed vessel, taking advantage of both the mobile base mobility and the robotic arm dexterity. As for the considered coverage method, [8], [9], [10] primarily address the coverage problem of a single propeller blade and plan tool paths directly in the task space without reasoning about the connectivity of the robot Surface-Constrained Configuration Space (SCCS).

The authors are with the Center for AI & Robotics (CAIR) and the Electrical Eng. Program, Engineering Division, New York University Abu Dhabi, Abu Dhabi, United Arab Emirates. E-mails: {rk4585,f.panetsos,spyridon.tarantos,kkyria}@nyu.edu.
* The authors contributed equally.

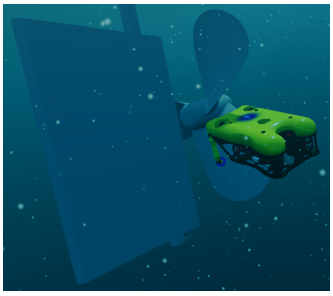

Fig. 1. The figure illustrates the scenario of a UVMS cleaning a marine propeller obstructed by an obstacle (rudder).

Consequently, when multiple surfaces (e.g., the three blades) or the presence of obstacles fragment the SCCS into disconnected components, the resulting paths may involve unnecessary tool lift-offs.

This work considers the problem of propeller cleaning using a UVMS equipped with a cleaning tool on its end-effector. Unlike the previous approaches [8], [9], [10], here we aim to program the cleaning of all the propeller blades, considering not only the individual blade-covering phase but also the robot transition between the individual blades. To make the scenario as realistic as possible, we consider the case in which the access to the propeller blades is interrupted by obstacles such as the vessel's rudder (see Figure 1). Both the presence of the obstacle, which may create discontinuities on the SCCS of the robot, and the fact that the area to be cleaned is not continuous, will force the robot to lift the cleaning tool and reattach it to the surface multiple times. Since establishing contact with the propeller may result to a poor surface quality or even damage the surface, minimizing the number of lift-offs is crucial and should be accounted for in the overall motion planning approach.

In the literature, there exist works that consider the problem of surface coverage when the robot SCCS is disconnected. In [11], [12], [13], [14], the singularity-free components of the disconnected SCCS are projected on the surface of interest. For each resulting area, a coverage planning algorithm is used for determining a tool path. In the case of overlapping projections, graph optimization approaches that minimize the tool lift-offs are employed, determining the portion of the disputed projected area that will correspond to each distinct SCCS component. These approaches rely on the assumption that for each surface point, there exist a finite number of Inverse Kinematic (IK) solutions to create the surface patches, and do not consider configuration space costs such as joint movement. Finally, these approaches do

not plan for the transition paths between surface parts and their entry and exit points.

An alternative approach is followed in [15], [16], [17], [18]. In its basic form [15], Joint Generalized Traveling Salesman Problem (JTSP), the surface coverage problem is treated as a Generalized Traveling Salesman Problem (GTSP) in the robot's SCCS, with nodes being the distinct robot configurations and groups of nodes formed by the configurations that correspond to the same surface point. The edges between the nodes of two different groups are weighted according to the transition cost from one configuration to the other. However, to mitigate the inherent combinatorial complexity of this method, one may resort to a sparser surface and SCCS sampling. To reduce the computational cost, [18] presents a hierarchically accelerated version of this method named hierarchically accelerated joint GTSP (H-JTSP). In particular, a *higher-level* GTSP (HL-GTSP) is solved in SCCS using only a reduced number of representative surface points and thus a reduced number of robot configurations in SCCS. Using the resulting connectivity to reduce the number of nodes in SCCS, a *lower-level* GTSP (LL-GTSP) is solved, considering all the surface points and the remaining nodes in the SCCS, leading eventually to the resulting coverage path. The above method can plan surface coverage paths as well as decide on the start and end configurations of the robot reconfiguration motion if required. However, this choice aims only to minimize the number of reconfigurations, disregarding the quality of the motion required for these reconfigurations. In the context of UVMS-based propeller cleaning, the starting and end configurations of the robot reconfiguration phase may significantly affect the length and thus the duration of the robot motion, mainly due to the robot's mobile base. As a result, this motion needs to be taken into account during the decision process. Furthermore, both JTSP and H-JTSP tend to generate tool paths with frequent sharp corners, which may lead to poor surface quality [19]. In the case of a marine propeller, the surface quality affects its performance [20]. Inspired by [21], where the isolines of various "slice" functions are used to generate desired coverage patterns, we design such a function to promote smoother patterns with few turns, similar to those suggested in [20] for marine propellers.

In this work, we extend the H-JTSP framework to UVMS-based propeller cleaning in obstacle-occluded environments. Compared to H-JTSP for fixed-base manipulators with no collision avoidance constraints, we: (i) add collision checking during IK solution sampling and GTSP edge cost design to generate a collision-free path. (ii) attempt piecewise-linear surface-constrained connections, instead of only relying on linear (in SCCS) segments for designing GTSP edge costs and paths; for disconnected surface components, we invoke a path planner for the reconfiguration motion and use the resulting path lengths in the GTSP cost design. In contrast, in H-JTSP, all reconfiguration GTSP edges are treated as equal with a constant large weight; (iii) reduce reconfiguration path planner computations in LL-GTSP by considering paths only between clusters chosen to be connected by a reconfiguration planner path in the HL-GTSP solution; (iv) promote coverage patterns with fewer sharp turns by sampling along equidistant geodesic isolines on blades and adding an isoline-alignment cost in LL-GTSP; and (v) convert the selected configuration-space path into an executable whole-body UVMS trajectory via spline smoothing and time minimization under velocity and acceleration constraints.

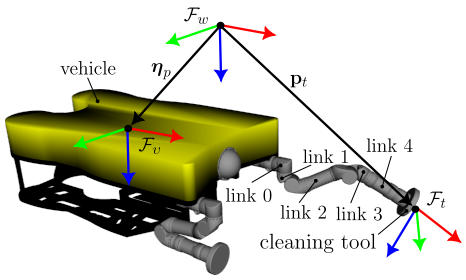

Fig. 2. The considered UVMS with a cleaning tool attached to its end-effector. Red, green, and blue arrows respectively denote the x, y, and z axes of the illustrated reference frames.

The rest of this paper is structured as follows: in Section II, we provide preliminary concepts and the formulation of our problem. In Section III, we describe our methodology. In Section IV, we demonstrate the effectiveness of our approach in realistic simulations. Finally, concluding remarks are offered in Section V.

II. Preliminaries

Figure 2 illustrates the considered UVMS consisting of a fully actuated mobile base and two robotic arms. For this work, only one of the robotic arms will be used, while the other will maintain its home configuration. We denote by $\mathcal{F}_w$ and $\mathcal{F}_v$ the inertial and vehicle-attached reference frames, respectively. We define the vehicle pose as $\boldsymbol{\eta} = \begin{bmatrix}\boldsymbol{\eta}_p^T & \boldsymbol{\eta}_o^T\end{bmatrix}^T \in \mathbb{R}^6$, where $\boldsymbol{\eta}_p \in \mathbb{R}^3$ is the position vector of the $\mathcal{F}_v$ origin with respect to $\mathcal{F}_w$ and $\boldsymbol{\eta}_o \in \mathbb{R}^3$ the vector representing the orientation of frame $\mathcal{F}_v$ with respect to $\mathcal{F}_w$. As for the configuration of the robotic arm, it is defined as $\boldsymbol{q} = \begin{bmatrix}\theta_1 & \dots & \theta_m\end{bmatrix}^T \in \mathbb{R}^m$, where θ_i is the angle of its i-th joint and m is the number of arm joints. By $\dot{\boldsymbol{q}} \in \mathbb{R}^m$ we denote its time derivative. Finally, we define the UVMS state by $\boldsymbol{\chi} = \begin{bmatrix}\boldsymbol{\xi}^T & \dot{\boldsymbol{\xi}}^T\end{bmatrix}^T$ where $\boldsymbol{\xi} = \begin{bmatrix}\boldsymbol{\eta}^T & \boldsymbol{q}^T\end{bmatrix}^T \in \mathbb{R}^n$ is the overall UVMS configuration with $n = 6 + m$.

The UVMS is equipped with a cleaning tool attached to its end-effector (see Figure 2). The considered cleaning tool has a rotating head with a circular *cleaning surface* (i.e., the part of the rotating head that comes into contact with the surface to be cleaned) of radius $r_t > 0$. We denote by $\mathcal{F}_t$ the tool frame whose origin is placed at the intersection of the cleaning surface and the axis of the tool rotation, while its x-axis is aligned with the axis of rotation pointing toward the surface to be cleaned. The position vector of the origin of $\mathcal{F}_t$ with respect to $\mathcal{F}_w$ is denoted by $\boldsymbol{p}_t \in \mathbb{R}^3$ while its x-axis unit vector by $\boldsymbol{n}_t \in \mathbb{R}^3$. Both vectors are related to the UVMS configuration by the forward kinematic maps $\boldsymbol{p}_t = \boldsymbol{k}_t(\boldsymbol{\xi})$ and $\boldsymbol{n}_t = \boldsymbol{\sigma}_t(\boldsymbol{\xi})$. Let also $\boldsymbol{n}_v = \boldsymbol{\sigma}_v(\boldsymbol{\xi}) \in \mathbb{R}^3$

be the vehicle's x-axis (surge) unit vector and the associated forward kinematic map.

Let us consider a point cloud $\mathcal{PC}_e \subset \mathbb{R}^3$ of the robot operational space acquired by an onboard underwater laser scanner. Let us also assume that the propeller does not move with respect to the world frame. In tuple $\mathcal{M}_O = \{\mathcal{M}_{O_1}, \mathcal{M}_{O_2}, ..., \mathcal{M}_{O_{N_o}}\}$ we collect the meshes of the N_o individual objects (including the propeller) extracted by processing the point cloud. By $\mathcal{M}_s = \{\mathcal{M}_1, \mathcal{M}_2, ..., \mathcal{M}_{N_s}\}$ we define a tuple containing the meshes of the N_s individual surfaces that the robot is called to clean.

A. Problem Statement

The problem consists in generating a whole-body UVMS motion that:

- R1: covers the surfaces collected in $\mathcal{M}_s$ with the tool, minimizing the number of tool lift-offs;
- R2: respects the UVMS kinematics as well as its hardware limitations (i.e., joint limits, velocity limits, and acceleration limits);
- R3: avoids collisions with the environment;
- R4: regulates the total time of the motion.

Note that requirements R2 and R3 are hard requirements since they are associated with robot safety, and thus they need to be enforced even at the expense of R1 and R4, which are related to the quality of the result and operation efficiency.

III. Methodology

Our approach is inspired by H-JTSP [18] in the sense that it follows a similar hierarchical structure illustrated in Figure 3:

1) Points sampled appropriately on the surfaces of $\mathcal{M}_s$ along with the associated surface normals are organized into $N_\mathcal{C}$ clusters based on their Euclidean distance and normal alignment.
2) At the representative point of each cluster (exemplar), a set of IK solutions is sampled. Each solution is then tested for its ability to *propagate* to each cluster point using an optimization-based IK solver. The clusters are subdivided until each exemplar has an adequate number of such IK solutions.
3) A HL-GTSP is formulated on the set of IK solutions at the exemplars $\Xi_\mathcal{C}$ (i.e., the considered groups of nodes correspond to the IK solutions of each exemplar) in order to determine an optimal sequence of IK solutions at the exemplars called *guide path*;
4) A LL-GTSP is formulated on the set Ξ_{LL} containing the IK solutions in the guide path and their corresponding propagated IK solutions at the rest of their cluster points. This determines a final coverage path.

Our approach builds upon the aforementioned structure to address the problem of mobile-base manipulator coverage in obstacle-occluded environments for the underwater propeller cleaning task in the following ways:

a) Coverage pattern promotion: We promote structured coverage paths by sampling surface points on the isolines of a distance field associated with a desired coverage pattern and shaping the LL-GTSP costs between two points based on their *isoline value difference*.

b) Collision-aware design: We check for collisions using the precomputed *signed distance field* (SDF) of the obstacles during the IK sampling and propagation step, as well as when designing the HL- and LL-GTSP weights.

c) GTSP weight design: We modify the function that checks for *linear* GTSP weights (i.e., for linear connections in $\Xi_\mathcal{C}$ and Ξ_{LL} for the HL- and LL-GTSP, respectively) by including collision checking and UVMS-related constraints. Furthermore, we have added a function that checks for *piecewise-linear* connections in $\Xi_\mathcal{C}$ and Ξ_{LL} by checking for linear connections of IK solutions on the linear interpolation of two surface points. Finally, GTSP edges spanning disconnected surfaces are costed with RRT-Connect. At HL-GTSP, we compute a single representative path for each pair of nearby exemplars from different surface components; at LL-GTSP, we compute paths for all point pairs between clusters chosen by the guide path as requiring reconfiguration.

d) Time parametrization: We concatenate the linear, piecewise-linear, and RRT-Connect paths chosen by the LL-GTSP into the final coverage path and parametrize it through cubic spline interpolation. Finally, we solve a Non-Linear Program (NLP) that minimizes the task duration to find a UVMS state trajectory satisfying velocity and acceleration limits.

In the rest of this section, we give the details of the aforementioned steps: subsection III-A presents the sampling and clustering step, subsection III-B describes the IK sampling and propagation procedure, subsections III-C and III-D detail the HL and LL-GTSP design, and finally, subsection III-E involves the generation of the final coverage path and UVMS state trajectory.

A. Sampling surface points and clustering

We sample points on each surface $\mathcal{M}_i$ in $\mathcal{M}_s$ with resolution set by the tool radius r_t.

a) Point sampling: For each blade surface $\mathcal{M}_i$, let $\phi_i(\boldsymbol{p})$ be the geodesic distance field from the blade root curve $\mathcal{P}_{\text{root},i} \subset \mathcal{M}_i$, computed via the heat method [22]. We sample along the isolines $\{\phi_i = jr_t\}$, $j = 0, 1, \ldots, \lfloor \phi_{i,\max}/r_t \rfloor$ (where $\phi_{i,\max} = \max_{\boldsymbol{p}\in\mathcal{M}_i} \phi_i(\boldsymbol{p})$), placing points at r_t spacing along each isoline.

Let the total set of N_c sampled points, and the set of corresponding surface normal vectors be

$$\mathcal{P}_c = \{\boldsymbol{p}_{c,1}, \ldots, \boldsymbol{p}_{c,N_c}\}, \qquad \mathcal{N}_c = \{\boldsymbol{n}_{c,1}, \ldots, \boldsymbol{n}_{c,N_c}\}$$

and define position vector-normal vector pairs $\boldsymbol{x}_{c,i} = (\boldsymbol{p}_{c,i}, \boldsymbol{n}_{c,i})$. Also define the isoline values at each point $\boldsymbol{p}_{c,i} \in \mathcal{M}_j$ as $\phi_{c,i}$ where $\phi_{c,i} = \phi_j(\boldsymbol{p}_{c,i})$.

b) Point connectivity: We reconstruct a surface from $\mathcal{P}_c$ using the Ball Pivoting Algorithm [23] and form the adjacency set $\mathcal{E}_c \subset \{1, ..., N_c\} \times \{1, \ldots, N_c\}$ by connecting indices whose points share a mesh edge.

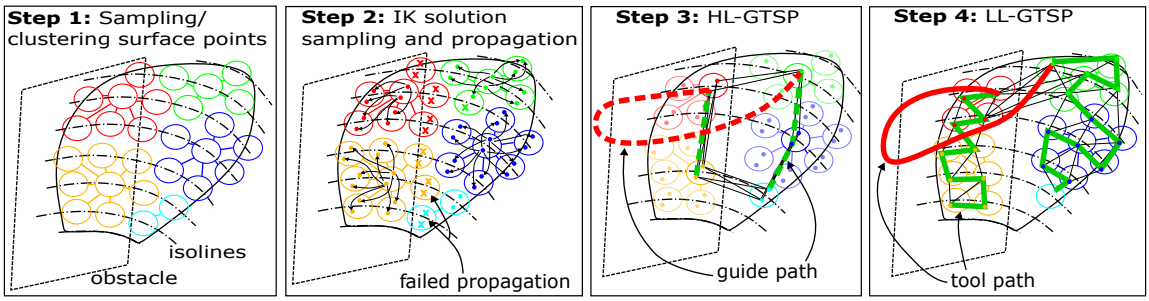

Fig. 3. Pipeline: (1) Sample and cluster surface points (circles) along isolines (dashed–dotted). Different colors illustrate clusters; (2) compute IK solutions (colored dots) at exemplars and propagate to non-exemplars (arrows), retain only fully propagated solutions and remove points with no IK solutions (marked by "x"); (3) solve the HL-GTSP to obtain a guide path (dashed line: the green part is surface-constrained and red part corresponds to tool lift-offs); (4) solve the LL-GTSP on the IKs retained by the guide path to produce the final coverage path.

c) Clustering into exemplars: We cluster the surface points into $N_{\mathcal{C}}$ sets $\mathcal{C} = \{\mathcal{C}_1, \ldots, \mathcal{C}_{N_{\mathcal{C}}}\}$ where $\mathcal{C}_i := \{j \in \{1, \ldots, N_c\} : \boldsymbol{p}_{c,j}$ belongs in the i-th cluster$\}$ using affinity propagation [24] with a similarity metric that combines Euclidean distance ($\|\cdot\|_2$) and normal alignment via a weighting parameter $w_s \geq 0$ which takes the form:

$$\mathrm{s}_{ij} = \begin{cases} -\left\|\boldsymbol{p}_{c,i} - \boldsymbol{p}_{c,j}\right\|_2 - w_s \arccos(\boldsymbol{n}_{c,i} \cdot \boldsymbol{n}_{c,j}), & i \neq j, \\ \text{percentile}_\tau(\{\mathrm{s}_{i,k} : k \neq i\}), & i = j, \end{cases} \tag{1}$$

where the diagonal term ($i = j$) is set via the τ-th percentile to control the number of exemplars (increasing τ yields more exemplars) [24]. For the i-th cluster, let $\boldsymbol{p}_{\mathcal{C}_i}$, $\boldsymbol{n}_{\mathcal{C}_i}$ and $\boldsymbol{x}_{\mathcal{C}_i}$ denote its exemplar point, normal, and pair, and let $N_{\mathcal{C}_i}$ be the cluster size. We define exemplar connectivity using $\mathcal{E}_c$ by defining the set

$$\mathcal{E}_{\mathcal{C}} := \{(i,j) : \exists\ k \in \mathcal{C}_i \text{ and } \ell \in \mathcal{C}_j \ \text{ with } (k,\ell) \in \mathcal{E}_c\}$$

Let N_{conn} be the number of connected components of the graph $G_c = \{\{1, \ldots, N_{\mathcal{C}}\}, \mathcal{E}_{\mathcal{C}}\}$ representing the surface connectivity and let $\kappa : \{1, \ldots, N_{\mathcal{C}}\} \to \{1, \ldots, N_{\text{conn}}\}$ map each exemplar index to its component label.

B. Sampling IK solutions at exemplars and propagation to non-exemplar points

Let $\boldsymbol{\Xi}_{\mathcal{C}_i}$ denote the set of IK solutions at exemplar $\boldsymbol{p}_{\mathcal{C}_i}$ and $\boldsymbol{\Xi}_{c,j}$ the set of IK solutions at surface point $\boldsymbol{p}_{c,j}$; let $M_{\mathcal{C}_i}$ be the size of $\boldsymbol{\Xi}_{\mathcal{C}_i}$. The k-th IK solution at exemplar i is $\boldsymbol{\xi}^k_{\mathcal{C}_i}$, and the ℓ-th IK solution at point j is $\boldsymbol{\xi}^\ell_{c,j}$.

a) IK solver via optimization and collision checking: Given a point-normal pair $\boldsymbol{x}_a = \{\boldsymbol{p}_a, \boldsymbol{n}_a\}$ and an objective function $J(\boldsymbol{\xi})$, we obtain an IK solution by solving:

$$\min_{\boldsymbol{\xi}} \ J(\boldsymbol{\xi}) \tag{2a}$$

$$\text{s.t.} \quad \boldsymbol{k}_t(\boldsymbol{\xi}) = \boldsymbol{p}_a, \tag{2b}$$

$$\boldsymbol{\sigma}_t(\boldsymbol{\xi}) \cdot \boldsymbol{n}_a = -1, \tag{2c}$$

$$\boldsymbol{\xi}_{\min} \leq \boldsymbol{\xi} \leq \boldsymbol{\xi}_{\max}, \tag{2d}$$

$$\boldsymbol{\sigma}_v(\boldsymbol{\xi}) \cdot (\boldsymbol{k}_t(\boldsymbol{\xi}) - \boldsymbol{\eta}_p) \geq d_{\text{safe}}, \tag{2e}$$

where (2b), (2c) are the tool position and orientation constraints respectively, $\boldsymbol{\xi}_{\min}, \boldsymbol{\xi}_{\max}$ are the configuration limits, and (2e) enforces *vehicle-arm clearance* for the UVMS by limiting the x-axis component of the tool position in $\mathcal{F}_v$ to a safe distance $d_{\text{safe}} > 0$[1]. Collision checking is performed as follows:

We precompute a signed distance field (SDF) $f_{\text{SDF}} : \mathbb{R}^3 \to \mathbb{R}$ for the obstacle set $\mathcal{O} = \bigcup_{i=1}^{N_o} \mathcal{M}_{O_i}$ in the considered workspace of the robot that contains $\mathcal{O}$. For a point with position vector $\boldsymbol{p}$ in this workspace, the signed distance value is defined as:

$$f_{\text{SDF}}(\boldsymbol{p}) = \begin{cases} \min_{\boldsymbol{y} \in \mathcal{O}} \|\boldsymbol{p} - \boldsymbol{y}\|_2, & \boldsymbol{p} \notin \text{int}(\mathcal{O}), \\ -\min_{\boldsymbol{y} \in \mathcal{O}} \|\boldsymbol{p} - \boldsymbol{y}\|_2, & \boldsymbol{p} \in \text{int}(\mathcal{O}), \end{cases} \tag{3}$$

where $\text{int}(\cdot)$ denotes interior in $\mathbb{R}^3$. We define $d_O(\boldsymbol{\xi}) = \min_{\boldsymbol{p} \in \mathcal{P}_R(\boldsymbol{\xi})} f_{\text{SDF}}(\boldsymbol{p})$ where $\mathcal{P}_R(\boldsymbol{\xi})$ is the set containing the representative points of the robot considered for collision checking. A configuration is collision-free if $d_O(\boldsymbol{\xi}) > 0$. Denote with $f_{\text{IK}}(J(\boldsymbol{\xi}), \boldsymbol{x}_a)$ the function generating a collision-free solution of (2) for a given function $J(\boldsymbol{\xi})$ and pair $\boldsymbol{x}_a$ by solving (2) and checking that the solution is collision-free.

b) Exemplar IK sampling: Sample M_{sam} arm configurations Q_{sam} within joint limits. For each exemplar i, populate $\boldsymbol{\Xi}_{\mathcal{C}_i}$ by solving $f_{\text{IK}}(\|\boldsymbol{q} - \boldsymbol{q}_{\text{sam}}\|_2, \boldsymbol{x}_{\mathcal{C}_i})$ for all $\boldsymbol{q}_{\text{sam}} \in Q_{\text{sam}}$. If no valid solution is found at the exemplar, we iterate over the cluster's pairs $\boldsymbol{x}_{c,j}$ and promote the first point with valid IK solutions to exemplar otherwise, we remove the cluster points and update $\mathcal{P}_c, \mathcal{N}_c, \mathcal{C}, \mathcal{E}_c, \mathcal{E}_{\mathcal{C}}$ accordingly.

c) Propagation to non-exemplar points: For each cluster $i = 1, \ldots, N_{\mathcal{C}}$ and for every non-exemplar point $j \in \mathcal{C}_i$, we try to *propagate* exemplar IK solutions by solving $f_{\text{IK}}(\|\boldsymbol{\xi} - \boldsymbol{\xi}^k_{\mathcal{C}_i}\|_A, \boldsymbol{x}_{c,j})$ for $k = 1, \ldots, M_{\mathcal{C}_i}$ and inserting the results into $\boldsymbol{\Xi}_{c,j}$. We denote $\|\boldsymbol{\xi}\|_A = \sqrt{\boldsymbol{\xi}^T A \boldsymbol{\xi}}$ and A constant positive-definite $n \times n$ diagonal weight matrix. To ensure propagation to every cluster point, we require that, at each exemplar, at least a fraction a of the exemplar IK solutions propagate to all points in the cluster. If this fails, we iteratively remove points from the cluster until the previous condition holds. If the points removed from the cluster admit valid IK solutions, we promote them to new exemplars (i.e., new clusters are added) and update $\mathcal{C}$; otherwise, we remove them completely and update $\mathcal{P}_c, \mathcal{N}_c, \mathcal{C}, \mathcal{E}_c, \mathcal{E}_{\mathcal{C}}$ accordingly.

[1]This constraint, in combination with the joint limits, enforces self-collision avoidance for our UVMS in Figure 2

Finally, at each cluster, we keep only those exemplar IK solutions (and their propagated counterparts) that propagated to all points and appropriately update the sets. Each IK solution at an exemplar corresponds to a single IK solution at each point in the same cluster. Thus, we can arrange the IK solutions at every cluster such that the i-th solution at the exemplar corresponds to the i-th solution in every cluster point.

C. HL-GTSP setup

We solve a high-level GTSP whose groups are the sets $\Xi_{\mathcal{C}_i}$. Since the GTSP output is a *tour* (i.e., a cyclic sequence of nodes), we add a dummy node connected with zero weights to all the other nodes and remove it from the final solution to get a *path*. To define edge costs, we test three forms of connections between two IK solutions $\boldsymbol{\xi}_a$, $\boldsymbol{\xi}_b$ associated with pairs $\boldsymbol{x}_a = (\boldsymbol{p}_a, \boldsymbol{n}_a)$ and $\boldsymbol{x}_b = (\boldsymbol{p}_b, \boldsymbol{n}_b)$: 1) surface-constrained linear connections, 2) surface-constrained piecewise linear connections, 3) surface-unconstrained connections:

a) Surface-constrained linear connections: Two IK solutions are *surface-constrained linearly connected* if N_{lin} sampled intermediate configurations on their linear interpolation satisfy the tool pose, clearance, and collision constraints:

$$f_{\text{lin}}(\boldsymbol{\xi}_a, \boldsymbol{\xi}_b) = \text{True iff } \left\|\boldsymbol{k}_t(\boldsymbol{\xi}^i_{\text{lerp}}) - \boldsymbol{p}^i_{\text{lerp}}\right\|_2 \le \tau_p, \tag{4a}$$

$$\left|\boldsymbol{\sigma}_t(\boldsymbol{\xi}^i_{\text{lerp}}) \cdot \boldsymbol{n}^i_{\text{lerp}} + 1\right| \le \tau_o, \tag{4b}$$

$$\boldsymbol{\sigma}_v(\boldsymbol{\xi}^i_{\text{lerp}}) \cdot \left(\boldsymbol{k}_t(\boldsymbol{\xi}^i_{\text{lerp}}) - \boldsymbol{\eta}^i_{\text{lerp},p}\right) \ge d_{\text{safe}}, \tag{4c}$$

$$d_O(\boldsymbol{\xi}^i_{\text{lerp}}) > 0. \tag{4d}$$

for $i = 1, \ldots, N_{\text{lin}}$, with $\boldsymbol{\xi}^i_{\text{lerp}} = \text{lerp}(\boldsymbol{\xi}_a, \boldsymbol{\xi}_b, i)$, $\boldsymbol{p}^i_{\text{lerp}} = \text{lerp}(\boldsymbol{p}_a, \boldsymbol{p}_b, i)$, and $\boldsymbol{n}^i_{\text{lerp}} = \text{lerp}(\boldsymbol{n}_a, \boldsymbol{n}_b, i)$ where $\text{lerp}(\cdot, \cdot, i)$ returns the i-th of N_{lin} equally spaced points on the linear interpolation between the inputs. For normal vectors and orientations, we use spherical linear interpolation (SLERP).

b) Surface-constrained piecewise-linear connections: We define a piecewise variant of the previous function by solving IK at N_{lin} interpolated point-normal pairs $\boldsymbol{x}^i_{\text{lerp}} = (\boldsymbol{p}^i_{\text{lerp}}, \boldsymbol{n}^i_{\text{lerp}})$ between $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$ and checking that they are linearly connected. We consider two IK solutions as *surface-constrained piecewise-linearly connected* if:

$$f_{\text{S}}(\boldsymbol{\xi}_a, \boldsymbol{\xi}_b) = \text{True iff } f_{\text{lin}}(\boldsymbol{\xi}_a, \boldsymbol{\xi}^*_1) \wedge \cdots \wedge f_{\text{lin}}(\boldsymbol{\xi}^*_{N_{\text{lin}}}, \boldsymbol{\xi}_b), \tag{5}$$
$$\boldsymbol{\xi}^*_i = f_{\text{IK}}\big(\|\boldsymbol{\xi} - \boldsymbol{\xi}^i_{\text{lerp}}\|_A, \boldsymbol{x}^i_{\text{lerp}}\big),$$

The length of the associated piecewise-linear path is denoted as $\ell_S(\boldsymbol{\xi}_a, \boldsymbol{\xi}_b)$.

c) Surface-unconstrained connections: For this type of connection, we do not require the tool to satisfy the position and orientation constraints of (2b)–(2c), (4a)–(4b). Instead, we consider *tool lift-off reconfigurations*, for which we employ the RRT-Connect planner [25], including $f_{\text{SDF}}(\cdot)$ for collision checking, vehicle-arm clearance constraint (2e), and configuration limits. RRT-Connect is effective and efficient for path planning in high-dimensional configuration spaces such as that of the UVMS. The length of a path found by RRT-Connect is denoted by $\ell_{\text{RRT}}(\boldsymbol{\xi}_a, \boldsymbol{\xi}_b)$.

d) HL edge costs: Considering two IK solutions $\boldsymbol{\xi}^k_{\mathcal{C}_i}, \boldsymbol{\xi}^\ell_{\mathcal{C}_j}$ at the clusters i, j $(i \neq j)$ respectively, the corresponding HL-GTSP edge cost is computed as follows:

$$W_{\text{HL}}\big(\boldsymbol{\xi}^k_{\mathcal{C}_i}, \boldsymbol{\xi}^\ell_{\mathcal{C}_j}\big) = \begin{cases} \big\|\boldsymbol{\xi}^k_{\mathcal{C}_i} - \boldsymbol{\xi}^\ell_{\mathcal{C}_j}\big\|_A, & \text{(A)} \\ B_{\text{HL},1} + \ell_S\big(\boldsymbol{\xi}^k_{\mathcal{C}_i}, \boldsymbol{\xi}^\ell_{\mathcal{C}_j}\big), & \text{(B)} \\ B_{\text{HL},2} + \ell_{\text{RRT}}\big(\bar{\boldsymbol{\xi}}_{\mathcal{C}_i}, \bar{\boldsymbol{\xi}}_{\mathcal{C}_j}\big), & \text{(C)} \\ B_{\text{HL},3} & (D). \end{cases} \tag{6}$$

Cases:

(A) $f_{\text{lin}}\big(\boldsymbol{\xi}^k_{\mathcal{C}_i}, \boldsymbol{\xi}^\ell_{\mathcal{C}_j}\big) \wedge (i,j) \in \mathcal{E}_{\mathcal{C}}$,

(B) $\neg f_{\text{lin}}\big(\boldsymbol{\xi}^k_{\mathcal{C}_i}, \boldsymbol{\xi}^\ell_{\mathcal{C}_j}\big) \wedge f_S\big(\boldsymbol{\xi}^k_{\mathcal{C}_i}, \boldsymbol{\xi}^\ell_{\mathcal{C}_j}\big) \wedge (i,j) \in \mathcal{E}_{\mathcal{C}}$,

(C) $\big(\neg f_{\text{lin}}\big(\boldsymbol{\xi}^k_{\mathcal{C}_i}, \boldsymbol{\xi}^\ell_{\mathcal{C}_j}\big) \wedge \neg f_S\big(\boldsymbol{\xi}^k_{\mathcal{C}_i}, \boldsymbol{\xi}^\ell_{\mathcal{C}_j}\big) \wedge (i,j) \in \mathcal{E}_{\mathcal{C}}\big) \vee$
$(\kappa(i) \neq \kappa(j) \wedge \big\|\boldsymbol{p}_{\mathcal{C}_i} - \boldsymbol{p}_{\mathcal{C}_j}\big\|_2 \le \tau_{\mathcal{C}})$,

(D) otherwise.

Case (A) corresponds to *linearly connected* edges between neighboring points. In case (B), we check for a *piecewise-linear connection* between neighbors using $f_S(\cdot, \cdot)$. Case (C) corresponds to *tool lift-off reconfigurations*: either no linear or piecewise-linear surface-constrained connection exists between neighboring IK solutions, or a transition between different surface components is required ($\kappa(i) \neq \kappa(j)$) when the exemplar distance is below a threshold $\tau_{\mathcal{C}}$. To assign a cost to such edges without planning over all $M_{\mathcal{C}_i} \times M_{\mathcal{C}_j}$ pairs, we approximate the cost by a single RRT-Connect path between *representative* IK solutions $\bar{\boldsymbol{\xi}}_{\mathcal{C}_i}$ and $\bar{\boldsymbol{\xi}}_{\mathcal{C}_j}$, where

$$\bar{\boldsymbol{\xi}}_{\mathcal{C}_i} = \arg\min_{\boldsymbol{\xi} \in \Xi_{\mathcal{C}_i}} \Big\|\boldsymbol{\xi} - \tfrac{1}{M_{\mathcal{C}_i}} \sum_{k=1}^{M_{\mathcal{C}_i}} \boldsymbol{\xi}^k_{\mathcal{C}_i}\Big\|_A.$$

The resulting path length $\ell_{\text{RRT}}\big(\bar{\boldsymbol{\xi}}_{\mathcal{C}_i}, \bar{\boldsymbol{\xi}}_{\mathcal{C}_j}\big)$ is used as a representative for all IK solution pairs between exemplars i, j under case (C). This approximation is acceptable because the HL-GTSP solution serves only to *guide* which inter-cluster connections are considered later in the LL-GTSP, where tool lift-off paths are refined using exact RRT-Connect paths. Case (D) collects the remaining undesirable tool lift-off reconfigurations: (i) pairs of non-neighboring exemplars within the same surface component ($\kappa(i) = \kappa(j)$), and (ii) pairs from different components whose exemplars are farther apart than $\tau_{\mathcal{C}}$. The former are connected through intermediate neighbors in the HL-GTSP solution, while the latter entail excessive path length and computation time, making them undesirable reconfigurations. To prioritize the choice of $N_{\mathcal{C}}$ optimal solution edges in the following order: case (A) - case (B) - case (C) - case (D), we set $B_{\text{HL},1} = N_{\mathcal{C}} \max(W^{(A)}_{\text{HL}}) + \varepsilon$, $B_{\text{HL},2} = N_{\mathcal{C}} \max(W^{(B)}_{\text{HL}}) + \varepsilon$, $B_{\text{HL},3} = N_{\mathcal{C}} \max(W^{(C)}_{\text{HL}}) + \varepsilon$, where $W^{(i)}_{HL}, i = \{A, B, C\}$ the weights of each case and $\varepsilon > 0$ an arbitrarily chosen small positive constant. This prioritizes choosing surface-constrained transitions over tool lift-offs.

Solving the HL-GTSP yields a guide path

$$\mathcal{G} = \{g_1, \ldots, g_{N_{\mathcal{C}}}\}, \qquad \mathcal{J} = \{j_1, \ldots, j_{N_{\mathcal{C}}}\},$$

where g_k is the cluster index and j_k the selected IK solution at the k-th exemplar in the guide-path. The IK solutions that pass to the LL-GTSP are collected in the set:

$$\Xi_{\text{LL}} = \{\boldsymbol{\xi}_{c,i}^{j_k} : i \in \mathcal{C}_{g_k},\ k = 1, \dots, N_{\mathcal{C}}\}.$$

We define guide-path tool lift-off reconfiguration edges $(\boldsymbol{\xi}_{\mathcal{C}_{g_k}}^{j_k}, \boldsymbol{\xi}_{\mathcal{C}_{g_{k+1}}}^{j_{k+1}})$ by defining the index sets

$$\mathcal{K}_{\text{RRT}} = \Big\{k :\ W_{\text{HL}}\big(\boldsymbol{\xi}_{\mathcal{C}_{g_k}}^{j_k}, \boldsymbol{\xi}_{\mathcal{C}_{g_{k+1}}}^{j_{k+1}}\big) \geq B_{\text{HL},2}\Big\}.$$

The guide-path induced LL tool lift-off reconfiguration candidate pairs are

$$\Xi_{\text{RRT}} = \bigcup_{k \in \mathcal{K}_{\text{RRT}}} \Big(\{\boldsymbol{\xi}_{c,i}^{j_k} : i \in \mathcal{C}_{g_k}\} \ \times\ \{\boldsymbol{\xi}_{c,i'}^{j_{k+1}} : i' \in \mathcal{C}_{g_{k+1}}\}\Big).$$

Thus, only points whose exemplar had a tool lift-off reconfiguration in the guide path will be examined for such a reconfiguration.

D. LL-GTSP setup

At the LL-GTSP, we connect all surface points using only the IK solutions in Ξ_{LL}. For $\boldsymbol{\xi}_{c,i}^{k}, \boldsymbol{\xi}_{c,j}^{\ell} \in \Xi_{\text{LL}}$ with $i \neq j$, we define

$$W_{\text{LL}}\big(\boldsymbol{\xi}_{c,i}^{k}, \boldsymbol{\xi}_{c,j}^{\ell}\big) = \begin{cases} \big\|\boldsymbol{\xi}_{c,i}^{k} - \boldsymbol{\xi}_{c,j}^{\ell}\big\|_A + \Delta\phi_{i,j}, & \text{(E)} \\ B_{\text{LL},1} + \ell_S\big(\boldsymbol{\xi}_{c,i}^{k}, \boldsymbol{\xi}_{c,j}^{\ell}\big) + \Delta\phi_{i,j}, & \text{(F)} \\ B_{\text{LL},2} + \ell_{\text{RRT}}\big(\boldsymbol{\xi}_{c,i}^{k}, \boldsymbol{\xi}_{c,j}^{\ell}\big), & \text{(G)} \\ B_{\text{LL},3} & \text{(H)}. \end{cases} \tag{7}$$

Cases:

(E) $f_{\text{lin}}\big(\boldsymbol{\xi}_{c,i}^{k}, \boldsymbol{\xi}_{c,j}^{\ell}\big) \wedge (i,j) \in \mathcal{E}_c$,
(F) $\neg f_{\text{lin}}\big(\boldsymbol{\xi}_{c,i}^{k}, \boldsymbol{\xi}_{c,j}^{\ell}\big) \wedge f_S\big(\boldsymbol{\xi}_{c,i}^{k}, \boldsymbol{\xi}_{c,j}^{\ell}\big) \wedge (i,j) \in \mathcal{E}_c$,
(G) $\neg$(E) $\wedge\ \neg$(F) $\wedge\ \big(\boldsymbol{\xi}_{c,i}^{k}, \boldsymbol{\xi}_{c,j}^{\ell}\big) \in \Xi_{\text{RRT}}$,
(H) otherwise.

where $\Delta\phi_{i,j} = \beta|\phi_{c,i} - \phi_{c,j}|$ is a weighted cost with weight $\beta > 0$ that promotes alignment with the isolines of the geodesic distance field calculated at each blade in III-A. Cases (E), (F) define the linearly connected and piecewise-linearly connected surface-constrained neighbor edges in the LL-GTSP. Case (G) corresponds to edges whose clusters were connected with a tool lift-off reconfiguration in HL-GTSP. This focuses planner calls exactly where HL indicates a lift-off is necessary, avoids exploding the number of RRT-Connect calls with all tool lift-off combinations at LL, and preserves the HL reconfiguration structure while allowing LL to refine the exact cluster entry/exit points. Case (H) contains the remaining tool lift-off reconfigurations. As in HL-GTSP, to prioritize edges in the following order: case (E) - case (F) - case (G) - case (H), we set $B_{\text{LL},1} = N_c \max(W_{\text{LL}}^{(E)}) + \varepsilon$, $B_{\text{LL},2} = N_c \max(W_{\text{LL}}^{(F)}) + \varepsilon$, $B_{\text{LL},3} = N_c \max(W_{\text{LL}}^{(G)}) + \varepsilon$, where $W_{LL}^{(i)}, i = \{E, F, G\}$ the weights of each case. This prioritizes surface-constrained transitions over tool lift-offs.

When calculating $W_{\text{LL}}(\cdot,\cdot)$, we store the paths in Case (F) and Case (G) so we can include them in the final coverage path if the GTSP solver selects their corresponding edges. If an edge in case (H) is selected, we use our RRT-Connect planner to calculate a path. Concatenating the selected linear, piecewise-linear, and RRT-Connect paths from the output LL-GTSP solution yields the C-space path with N configurations, $\Xi = \{\boldsymbol{\xi}_1, \dots, \boldsymbol{\xi}_N\}$.

E. Time parametrization

Given the path, our goal is to turn it into a trajectory that satisfies joint velocity and acceleration limits dictated by our robot hardware limitations, $\dot{\boldsymbol{\xi}}_{\min} \leq \dot{\boldsymbol{\xi}} \leq \dot{\boldsymbol{\xi}}_{\max}$, $\ddot{\boldsymbol{\xi}}_{\min} \leq \ddot{\boldsymbol{\xi}} \leq \ddot{\boldsymbol{\xi}}_{\max}$, while minimizing the total time required for the task. Using arclength l along Ξ, we fit a cubic spline to $\{(l_i, \boldsymbol{\xi}_i)\}_{i=1}^{N}$ and evaluate its first and second derivatives, $\frac{d\boldsymbol{\xi}}{dl}$ and $\frac{d^2\boldsymbol{\xi}}{dl^2}$ respectively, at l_i. To find the velocity state $\dot{\Xi} = \{\dot{\boldsymbol{\xi}}_i\}_{i=1}^{N}$, we solve the following NLP with decision variables the time duration between consecutive points, the first and second derivatives of l_i w.r.t. time $\Delta\mathcal{T} = \{\Delta t_i\}_{i=1}^{N-1}$, $\dot{\mathcal{L}} = \{\dot{l}_i\}_{i=1}^{N}$, $\ddot{\mathcal{L}} = \{\ddot{l}_i\}_{i=1}^{N}$:

$$\min_{\Delta\mathcal{T}, \dot{\mathcal{L}}, \ddot{\mathcal{L}}} \quad \sum_{i=1}^{N-1} \Delta t_i \tag{8a}$$

$$\text{s.t.} \quad \dot{\boldsymbol{\xi}}_{\min} \ \leq\ \frac{d\boldsymbol{\xi}}{dl}\bigg|_{l_i} \dot{l}_i \ \leq\ \dot{\boldsymbol{\xi}}_{\max}, \quad i = 1, \dots, N, \tag{8b}$$

$$\ddot{\boldsymbol{\xi}}_{\min} \ \leq\ \frac{d\boldsymbol{\xi}}{dl}\bigg|_{l_i} \ddot{l}_i + \frac{d^2\boldsymbol{\xi}}{dl^2}\bigg|_{l_i} \dot{l}_i^2 \ \leq\ \ddot{\boldsymbol{\xi}}_{\max}, \tag{8c}$$

$$l_{i+1} = l_i + \tfrac{1}{2}(\dot{l}_i + \dot{l}_{i+1})\Delta t_i, \quad i = 1, \dots, N-1, \tag{8d}$$

$$\dot{l}_{i+1} = \dot{l}_i + \ddot{l}_i \Delta t_i, \quad i = 1, \dots, N-1, \tag{8e}$$

$$\Delta t_i > 0, \quad i = 1, \dots, N-1,\ \dot{l}_1 = \dot{l}_N = 0. \tag{8f}$$

Set $t_1 = 0$ and $t_i = \sum_{j=1}^{i-1} \Delta t_j$; velocities at every time instance are $\dot{\boldsymbol{\xi}}_i = \frac{d\boldsymbol{\xi}}{dl}\big|_{l_i} \dot{l}_i$, $i = 1, \dots, N$. The resulting UVMS whole-body trajectory is $\{t_i, \boldsymbol{\xi}_i, \dot{\boldsymbol{\xi}}_i\}_{i=1}^{N}$.

IV. Simulations

In this section, we demonstrate the effectiveness of the proposed approach. We consider the underwater vehicle VideoRay Defender equipped with two Reach Alpha 5 robotic manipulators (only one of the manipulators is used). The dimensions of the vehicle are $71 \times 39 \times 23$ cm, while links 0 to 4 (see Figure 2) have lengths: $l_0 = 12$ cm, $l_1 = 2$ cm, $l_2 = 14$ cm, $l_3 = 2$ cm and $l_4 = 20$ cm (including the tool) respectively. The cleaning tool diameter is $2r_t = 10$ cm. The UVMS is assigned the task to clean the blades of a typical propeller (B-series) with a total diameter of 0.8 m. The access to the propeller is interrupted by the vessel's rudder as illustrated in Figure 6 with dimensions 1.3×0.82 m and 0.2 m minimum distance from the propeller. Although the rest of the vessel does not appear in the simulation environment, we assume that the robot cannot access the area behind the propeller and over a certain height above the edge of the rudder, constraints that are considered in our motion planning method. The rudder can rotate around its axis from $\pm 30°$. We set $M_{\text{sam}} = 100$, $d_{\text{safe}} = 0.5$ m, $A = I_{n \times n}$, $N_{\text{lin}} = 2$, $\tau_p = 0.5$ cm, $\tau_o = 0.2$ and $\tau_{\mathcal{C}} = 1$. For solving

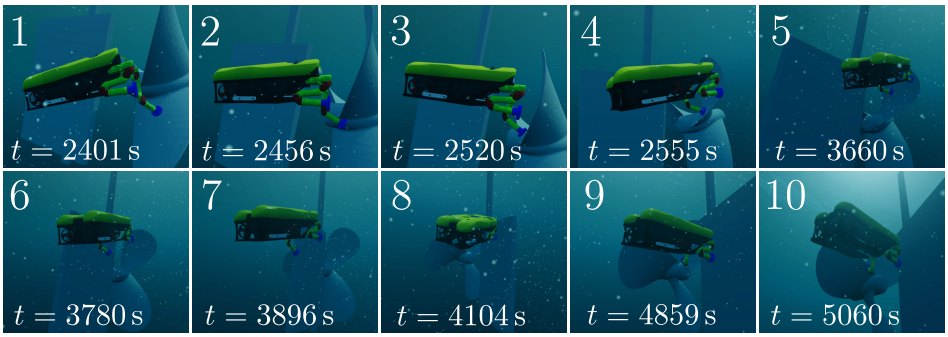

Fig. 4. Snapshots of the robot executing the generated trajectory. Snapshots (2),(3) correspond to the robot transition from Blade 1 to Blade 2, snapshots (6), (7) and (8) correspond to the transition between the two disconnected surfaces of Blade 2, while (9) to the transition between Blade 2 to Blade 3.

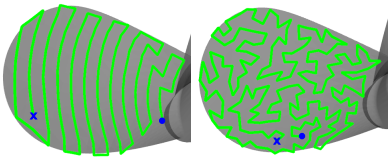

Fig. 5. Tool path with isoline alignment costs (left) and without (right). The blue dot and blue "x" represent the start and the end point of each path.

TABLE I

PERFORMANCE METRICS

Metric	Test 1a	Test 1b	Test 2
Tool path length [m]	7.6 ± 1.1	8.5 ± 2.5	27.1 ± 2.7
Min. dist. to obstacles [cm]	0.5 ± 0.1	0.5 ± 0.1	0.4 ± 0.1
# of sharp turns ($> 30°$)	31 ± 3	167 ± 4	312 ± 7
Trajectory time [s]	867 ± 11	1710 ± 17	6601 ± 25
# of covered/total points	$129 \pm 7/148$	$196 \pm 15/227$	$364 \pm 21/442$
Computation time [s]	602 ± 15	996 ± 16	4311 ± 40

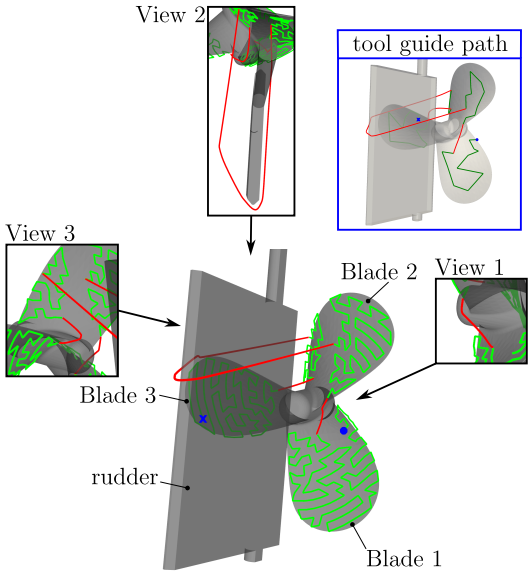

Fig. 6. The resulting tool coverage path from multiple views. With green color, we represent the tool path when it touches the propeller, while with red, the path when the tool is detached from the surface in order for the robot to perform a tool lift-off reconfiguration. The tool guide path connecting the exemplars, as obtained from the HL-GTSP solution, is also provided.

the HL and LL-GTSP, we used the GLKH solver [26]. The function $f_{\text{IK}}(\cdot, \cdot)$ and the NLP were implemented using CasADi [27]. We use the RRT-Connect algorithm provided by OMPL [28]. The simulations were conducted using the ROS-based simulator, Stonefish [29], and an Intel Core i9 CPU at 3.2 GHz with 64 GB of RAM. A PID controller was used for tracking the generated whole-body UVMS motion for demonstration purposes.

We evaluated our approach in two tests and summarized the results in Table I, averaged for 10 equally spaced rudder angles within $\pm 30°$. **Test 1** contrasts (a) isoline-based sampling with isoline-alignment costs ($\beta = 10$) against (b) uniform sampling of comparable density with no alignment cost ($\beta = 0$) strategy followed in [15], [18]. As illustrated in Figure 5, isoline guidance yields visibly smoother paths with far fewer sharp turns (i.e., tool path turns $> 30°$) (31 vs. 167) while keeping total path length comparable (7.6 m vs. 8.5 m). The reduction in high-curvature segments allows for faster motion, thus cutting execution time nearly in half (867 s vs. 1710 s). In **Test 2**, the robot is tasked to cover all three blades and to plan tool lift–offs between disconnected surface components. The resulting tool path is 27.1 m long, with minimum obstacle clearance 4 mm and 312 sharp turns. The time-parameterized trajectory produced by the NLP lasts 6601 s. The total computation time of 4311 s is acceptable for off-line planning. Due to collisions near the rudder, one blade splits into two components after removing points with no valid IK solutions (Section III-B), yielding four surface components with 364 out of 442 accessible points overall, which requires at least three tool lift-offs. The HL guide path (see Figure 6) identifies the minimum number of inter–component tool lift–off reconfigurations (three) and the entry/exit clusters where they should occur, while the LL stage refines the exact entry/exit points and computes the corresponding RRT–connect segments (red in Figure 6). The overall tool lift-off paths are short. The robot motion during the propeller cleaning is provided in the accompanying video, while snapshots of this motion are offered in Figure 4. The robot starts the cleaning operation from the lower blade as illustrated in both figures 4 and 6. After covering the blade, the robot lifts the cleaning tool to attach it to the second blade, performing the collision-free motion planned by the

RRT-connect algorithm (see snapshots 2-4). The robot covers the part of the second blade that can be covered from the right side of the rudder. At snapshots 6-8, we observe the reconfiguration motion that the robot performs to approach the rest of the second blade, avoiding collision with the rudder. Finally, the robot performs a last reconfiguration to bring the tool from the second blade to the third, covering eventually the accessible surfaces of the propeller blades.

V. Conclusion

In this work, we developed a coverage motion planning method for the task of UVMS-based propeller cleaning in the presence of obstacles. The proposed approach, consisting of a two-level generalized traveling salesman problem, is able to deal with scenarios in which multiple targeted surfaces and/or the presence of obstacles fragment the surface-constrained configuration space of the robot. The resulting whole-body collision-free path not only minimizes the reconfiguration motions dictated by the fragmented SCCS but also explicitly includes the associated reconfiguration paths. Moreover, by incorporating an isoline-alignment term into the GTSP weights, we bias the solution toward pattern-aligned tool paths, with few sharp turns, which may improve surface quality. Finally, the adopted time-parametrization approach converts the path into a whole-body trajectory that respects the robot hardware limitations while minimizing the total execution time. The proposed approach was validated in a realistic simulation scenario, showing its effectiveness.

Future work aims at considering the full robot dynamics and tool-surface dynamics for the motion planning process, as well as the experimental validation of our method.

Acknowledgment

This work is supported in part by the NYUAD Center for Artificial Intelligence and Robotics, funded by Tamkeen under the Research Institute Award CG010.

References

[1] P. Vuong, A. McKinley, and P. Kaur, "Understanding biofouling and contaminant accretion on submerged marine structures," *npj Materials Degradation*, vol. 7, no. 1, p. 50, Jun 2023.

[2] P. Liu, H. Zhang, Y. Fan, and D. Xu, "Microbially influenced corrosion of steel in marine environments: A review from mechanisms to prevention," *Microorganisms*, vol. 11, no. 9, 2023.

[3] M. L. Hakim, B. Nugroho, M. N. Nurrohman, I. K. Suastika, and I. K. A. P. Utama, "Investigation of fuel consumption on an operating ship due to biofouling growth and quality of anti-fouling coating," *IOP Conference Series: Earth and Environmental Science*, vol. 339, no. 1, p. 12037, 10 2019.

[4] H. Albitar, K. Dandan, A. Ananiev, and I. Kalaykov, "Underwater robotics: Surface cleaning technics, adhesion and locomotion systems," *International Journal of Advanced Robotic Systems*, vol. 13, no. 1, p. 7, 2016.

[5] T. Guo, X. Liu, T. He, and D. Song, "Synchro-drive-based underwater climbing adsorption robot," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 6250–6257, 2022.

[6] D. Souto, A. Faiña, F. López-Peña, and R. J. Duro, "Lappa: A new type of robot for underwater non-magnetic and complex hull cleaning," in *2013 IEEE International Conference on Robotics and Automation*, 2013, pp. 3409–3414.

[7] C. Song and W. Cui, "Review of Underwater Ship Hull Cleaning Technologies," *Journal of Marine Science and Application*, vol. 19, pp. 415 – 429, 2020.

[8] J. Long, Y. Tian, W. Chen, J. Leng, and Y. Wang, "Locating, trajectory planning and control of an underwater propeller cleaning manipulator," *Ocean Engineering*, vol. 243, p. 110262, 2022.

[9] Y. Liang, Z. Feng, J. Leng, and J. Long, "Propeller underwater cleaning trajectory planning," in *OCEANS 2019 MTS/IEEE SEATTLE*, 2019, pp. 1–5.

[10] R. Kopo, S. G. Tarantos, F. Panetsos, and K. J. Kyriakopoulos, "A whole-body uvms motion planning approach for underwater propeller cleaning," in *2025 Symposium on Maritime Informatics and Robotics*, 2025, pp. 1–8.

[11] T. Yang, J. Valls Miró, Y. Wang, and R. Xiong, "Non-revisiting coverage task with minimal discontinuities for non-redundant manipulators," in *Proceedings of Robotics: Science and Systems (RSS)*, Corvallis, Oregon, USA, July 2020.

[12] T. Yang, J. Valls Miro, Q. Lai, Y. Wang, and R. Xiong, "Cellular decomposition for nonrepetitive coverage task with minimum discontinuities," *IEEE/ASME Transactions on Mechatronics*, vol. PP, pp. 1–1, 08 2020.

[13] T. Yang, J. Valls Miro, M. Nguyen, Y. Wang, and R. Xiong, "Template-free nonrevisiting uniform coverage path planning on curved surfaces," *IEEE/ASME Transactions on Mechatronics*, vol. PP, pp. 1–9, 08 2023.

[14] T. Yang, J. Valls Miro, Y. Wang, and R. Xiong, "An improved maximal continuity graph solver for non-redundant manipulator non-revisiting coverage," *IEEE Transactions on Automation Science and Engineering*, vol. PP, pp. 1–13, 01 2024.

[15] J. Hess, G. D. Tipaldi, and W. Burgard, "Null space optimization for effective coverage of 3d surfaces using redundant manipulators," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 1923–1928.

[16] D. Kaljaca, B. Vroegindeweij, and E. van Henten, "Coverage trajectory planning for a bush trimming robot arm," *Journal of Field Robotics*, vol. 37, no. 2, pp. 283–308, 2020.

[17] C. Zhang, H. Qin, S. Sun, Y. Pan, K. Liu, T. Li, and X. Zhao, "Jpmdp: Joint base placement and multi-configuration path planning for 3d surface disinfection with a uv-c robotic system," *Robotics and Autonomous Systems*, vol. 174, p. 104644, 2024.

[18] Y. Wang and M. Gleicher, "Hierarchically accelerated coverage path planning for redundant manipulators," in *2025 IEEE International Conference on Robotics and Automation*, 2025, pp. 12 098–12 104.

[19] G. Yan, D. Zhang, J. Xu, and Y. Sun, "Corner smoothing for cnc machining of linear tool path: A review," *Journal of Advanced Manufacturing Science and Technology*, vol. 3, no. 2, p. 2023001, 2023. [Online]. Available: https://www.sciopen.com/article/10.51393/j.jamst.2023001

[20] T. Breteau, T. Damay, E. Duc, and J.-Y. Hascoët, "Design for manufacturing with tool paths adapted to marine propeller," *International Journal on Interactive Design and Manufacturing*, vol. 5, no. 4, pp. 271–275, 2011.

[21] E. U. Acar, H. Choset, A. A. Rizzi, P. N. Atkar, and D. Hull, "Morse decompositions for coverage tasks," *The International Journal of Robotics Research*, vol. 21, no. 4, pp. 331–344, 2002.

[22] K. Crane, C. Weischedel, and M. Wardetzky, "The heat method for distance computation," *Communications of the ACM*, vol. 60, no. 11, pp. 90–99, 11 2017.

[23] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, and G. Taubin, "The ball-pivoting algorithm for surface reconstruction," *IEEE Transactions on Visualization and Computer Graphics*, vol. 5, no. 4, pp. 349–359, 1999.

[24] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 5814, pp. 972–976, 2007.

[25] J. Kuffner and S. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *2000 IEEE International Conference on Robotics and Automation*, vol. 2, 2000, pp. 995–1001.

[26] K. Helsgaun, "Solving the equality generalized traveling salesman problem using the lin–kernighan–helsgaun algorithm," *Mathematical Programming Computation*, vol. 7, no. 3, pp. 269–287, 2015.

[27] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi – A software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019.

[28] I. A. Şucan, M. Moll, and L. E. Kavraki, "The Open Motion Planning Library," *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, pp. 72–82, December 2012.

[29] P. Cieślak, "Stonefish: An Advanced Open-Source Simulation Tool Designed for Marine Robotics, With a ROS Interface," in *OCEANS 2019 - Marseille*, Jun. 2019.