2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

RUMP: Robust Underwater Motion Planning in Dynamic Environments of Fast-moving Obstacles

Herman Biørn Amundsen[1], Torben Falleth Olsen[2], Marios Xanthidis[3], Martin Føre[2], and Eleni Kelasidi[3]

***Abstract*— Robust underwater motion planning of autonomous underwater vehicles (AUVs) in dynamic cluttered environments is a problem that has yet to be addressed in depth. Due to advances in technology and computational capacity, AUVs are expected to operate safely and autonomously in increasingly challenging environments, necessitating methods that are able to safely navigate robots in real-time. Though, most solutions remain overly cautious and conservative. This paper proposes RUMP, a novel locally-optimal motion planning framework for robust real-time autonomous underwater navigation in 3D cluttered environments consisting of observed static and dynamic obstacles. The problem is modeled using path optimization and can be solved in real-time with a common nonlinear solver. The constructed objective function allows deciding the local goal during optimization to both maximize safety within a planning horizon and minimize the expected distance to the target position. Furthermore, path safety is considered for the entire transition between consecutive states, utilizing a novel approach for continuous spatiotemporal collision checks. The proposed formulation provides safe performance even in environments with obstacles that may move orders of magnitude faster than the AUV itself. Simulation experiments, in different challenging scenarios of obstacles moving up to 100 times faster than the robot, showcase robustness and efficient real-time performance of more than 15 Hz.**

I. Introduction

Autonomous underwater vehicles (AUVs) are expected to operate in challenging underwater settings, executing tasks related to environmental monitoring, infrastructure inspection and maintenance in sectors such as oil-and-gas, aquaculture, and maritime security [1]–[4]. Such environments place a major challenge for state-of-the-art techniques both for state estimation [5] due to degraded sensor capabilities underwater, and for motion planning due to complex dynamics of the robots. Moreover, dynamic obstacles in the workspace, or even other AUVs [6]–[9], may often move faster than the target platform itself. This is especially true for autonomous aquaculture operations, a yet largely unexplored domain in robotics and a motivation for this work.

Modern aquaculture facilities consist of multiple cages, in which robots are expected to operate safely, while sharing workspace with deformable structures, such as the cages' nets, floating and bottom collars, and mooring lines [10], moving monitoring devices or other AUVs, and up to 200,000 fish, that generally move faster than the robot [11]. An example of the challenging conditions robots face in fish cages is shown in Figure 1. Enabling safe autonomous aquaculture operations necessitates robust real-time motion planning methods that can move a robot safely while surrounded by multiple dynamic obstacles.

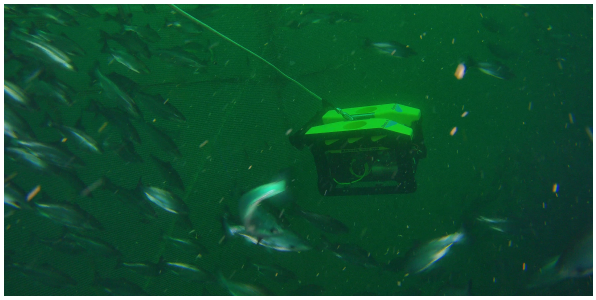

Fig. 1. The Argus Mini remotely operated vehicle in a fish farm close to a net and living fish; one of the target applications of this paper.

This paper proposes RUMP (Robust Underwater Motion Planning), a motion planning approach to enable safe operation in challenging environments, including aquaculture. This new framework addresses the challenging problem of motion planning in dynamically changing environments. For the development of RUMP, we got inspiration from robust methods utilized by the aerial robotics community, such as model predictive control (MPC) [12], and path optimization-based methods with strong collision-free guarantees in static [13] or dynamic environments [14].

RUMP, assuming knowledge of the detected dynamic obstacles, their estimated trajectories, localization, and a target destination, produces safe locally-optimal trajectories in the presence of both static and moving obstacles and agents. We introduce novel spatiotemporal collision constraints that provide in real-time continuous safety guarantees and efficient non-conservative solutions. This is achieved by computing clearance continuously for the entire transition towards each state, while the local goal is computed during optimization to maximize safety and maintain the specified horizon.

RUMP is able to avoid not only obstacles that move slower than the AUV, but also multiple obstacles that may move orders of magnitude faster, and is to the best of our knowledge the first real-time framework with such capabilities. Safety is maintained under the assumption that the robot is given enough time to detect and react to obstacles, and that these obstacles do not have malicious intents to collide with the vehicle; both of which are reasonable in several maritime

[1] Herman Biørn Amundsen and Martin Føre are with the Department of Engineering Cybernetics, NTNU [herman.b.amundsen, martin.fore]@ntnu.no

[2] Torben Falleth Olsen are with the Department of Engineering Cybernetics, NTNU and Kongsberg Discovery [torben.falleth.olsen@kd.kongsberg.com]

[3] Marios Xanthidis and Eleni Kelasidi are with the Aquaculture Robotics and Automation Group, SINTEF Ocean [marios.xanthidis, eleni.kelasidi]@sintef.no

This work was supported by the Research Council of Norway (ResiFarm: NO-327292, CHANGE: N313737) and Kongsberg Discovery.

domains and specifically during aquaculture operations.

To summarize, we present RUMP, a novel real-time locally-optimal motion planner that:

1) incorporates spatiotemporal collision avoidance, and produces efficient non-conservative safe paths in dynamic environments
2) enables robots to avoid obstacles moving orders of magnitude faster, assuming enough time for the robot to react and no malice by the interjecting objects.
3) computes in real-time a safe local goal that minimizes the expected path length towards the global goal
4) is rigorously tested in simulated dynamic environments of unique challenges featuring dynamic obstacles moving up to 100 times faster than the AUV

The related work is presented in Section II, the problem statement in Section III, and the proposed method in Section IV. The paper concludes with evaluation through extensive simulations in Section V, and conclusions in Section VI.

II. Related Work

The problem of transitioning a robotic system from an initial to a goal state while avoiding obstacles, also referred as motion planning [15], is among the most computationally hard problems, with P-SPACE hard complexity [16]. Recent reviews [17]–[19] indicate that decoupling motion planning with a global low-resolution and a local high-resolution planner fits especially well in the underwater domain, due to the limited range of sensor information, the partial observability of the environment, and the necessity for real-time performance; a prerequisite that becomes even harder to satisfy when the AUVs operate in a non-static environment.

Early work on safe navigation and obstacle avoidance in dynamic environments produced reactive and conservative behavior employing geometric methods in 2D [20], [21], along with popular implementations in real conditions with on-board sensors [22]. In the maritime domain, Zhang *et al.* presented a pipeline for an AUV avoiding dynamic obstacles in 2D [23] utilizing acoustic sensors, while Benjamin *et al.* utilized cameras to navigate an autonomous surface vehicle (ASV) through a dynamic environment by maintaining a safe distance [24]. Recent work by Lin *et al.* utilized deep learning and trained in static environments but showed capabilities of safe navigation also with moving obstacles [25].

The problem of safe navigation with dynamic obstacles has been explored to an extent by the aerial robotics community, with studies addressing the necessity for addressing this problem for unmanned aerial vehicles (UAVs) by comparing different mainstream approaches [26]. Jenie *et al.* introduced a geometric method for avoiding dynamic obstacles reactively by utilizing velocity vectors [27]. Lindqvist *et al.* provided a nonlinear MPC (NMPC) method for UAVs navigating in 3D and avoiding dynamic obstacles by keeping an increased distance for safety [28]. In a similar fashion, recent work utilized an NMPC and added chance constraints to increase safety and enable UAVs to deal with moving obstacles with uncertainty [29], while other research in the domain chose the direction of increasing computational efficiency of the NMPC to avoid effectively moving obstacles in 2D with Reinforcement Learning [30].

Several methods have been presented for safe underwater navigation in static environments based on sampling-based approaches [31]–[35], reactive geometric methods [36], deep learning techniques [37], [38], path optimization [39], [40], and model predictive control [41]–[43], but there is also a fair amount of research for handling dynamic obstacles. A 3D sampling-based motion planner was proposed for handling moving obstacles underwater, by projecting their motion and producing paths that avoided the area occupied by the obstacles during the entire operation [44]. Other researchers utilized particle swarm optimization approaches to avoid dynamic obstacles in 3D by utilizing velocity vectors during the optimization [45]. Lin *et al.* [25] provided a framework based on deep-learning that could reactively avoid dynamic obstacles in 2D, although it was trained only on static environments. The strategy of inflating moving obstacles has been employed in solutions that combine sampling-based techniques and efficient graph search [46], or combined with semantic awareness and computing a repulsive force from the moving obstacles [47]. Finally, in our previous works, efficient motion planners have been developed to allow AUVs to handle dynamic obstacles based on elastic bands [48], and with strong safety guarantees in the presence of uncertainty, currents, disturbances, and moving obstacles [49].

To the best of our knowledge, all the works above produce either conservative paths by avoiding areas that may be collision-free spatially or temporally, safety is resolution-dependent and challenges real-time performance, or assumes obstacles with limited velocities. RUMP addresses this research gap, by combining concepts from two powerful techniques: Firstly, we borrowed the formulation of Trajopt [13] which performs collision checks between the swept-out volumes of transitions of the robot with the volumes of the static obstacles. Secondly, we expanded on concepts from ITOMP [14], which handles dynamic obstacles by applying spatiotemporal collision constraints between the swept-out volumes of the moving obstacles with individual states of the robot for short time intervals. In this context, RUMP introduces novel spatiotemporal collision checking between the swept-out volumes of the robot and the corresponding swept-out volumes of the moving obstacles at each time step, resulting in increased safety and the capacity to produce safe trajectories even for obstacles moving arbitrarily fast, without sacrificing efficiency.

III. Problem Statement

This paper aims to compute safe trajectories for autonomous operations of AUVs in dynamic environments with static and moving obstacles. Let the configuration space be $\mathcal{C} = X \times Y \times Z$ for a robot capable of moving freely in the three translation axes. For the formulation of the proposed method, we assume holonomic kinematics, and that the vehicle and each obstacle could be represented by a single sphere of various sizes each. Please note that the key ideas

introduced in this work are not explicitly bounded by the above assumptions.

More formally, let $s_{t_1} = [x_1, y_1, z_1]$ be the current position of the robot at time t_1, G a goal region centered around a goal position $s_g = [x_g, y_g, z_g]$, and ρ_H a Euclidean distance defining the sphere in $\mathbb{R}^3$ of the planning horizon within a robot decides a trajectory. The planning horizon is indicated as $H_{s_{t_1}} = \mathcal{P}(s_{t_1}, \rho_H)$, with $\mathcal{P}(.)$ defining a sphere. Also let $S_{t_1}^{t_n} = \{s_{t_1}, s_{t_2}, \ldots, s_{t_n}\}$, where $S_{t_1}^{t_n} \in H_{s_{t_1}}$, be a trajectory starting from the current robot position that can be represented at arbitrary resolution for different values of $n \in \mathcal{N}$, and a set of times $T = \{t_1, t_2, \ldots, t_n\}$ when the robot reaches the corresponding waypoints of $S_{t_1}^{t_n}$. Additionally, either $s_{t_n} \in G$ or $s_{t_n} \in H_{s_{t_1}}$ so that the final state of each trajectory will either transition the robot to the goal, or it will move the robot greedily towards the goal. Finally, let P_s^r be the sphere representing the volume occupied by the robot at state $s \in \mathcal{C}$, and P_t^o be the volume occupied by a spherical obstacle $o \in O$ at time t, if O is the set of all obstacles. Then, the objective of our work can be described simply as:

$$f(S_{t_1}^{t_n}) = \min_{S_{t_1}^{t_n}} \sum_{i=1}^{n-1} \left\| s_{t_{i+1}} - s_{t_i} \right\| + \left\| s_g - s_{t_n} \right\|, \quad (1)$$

$$\text{s.t.} \quad \bigcup_{\substack{t_i \in T \\ s_{t_i} \in S_{t_1}^{t_n}}} (P_{s_{t_i}}^r \cap \bigcup_{o \in O} P_{t_i}^o) = \varnothing \quad (2)$$

In more simple terms, the goal is to provide locally optimal paths within $H_{s_{t_1}}$, that drive the robot towards G, while no collisions take place with any static or dynamic obstacle.

IV. Method Description

In this section, the proposed method will be described and details will be given regarding our contributions. RUMP aims to provide computationally efficient and minimal paths with robust collision avoidance capabilities. As mentioned earlier, we propose a very computationally efficient formulation, that enables avoidance of obstacles moving at arbitrary velocities, under reasonable assumptions.

For this purpose, we exploit path optimization, we introduce a novel spatiotemporal collision constraint, we propose a new term in the objective function and a new constraint to determine on-the-run local goals that minimize the local and the expected global path length, and finally, we solve the problem by utilizing numerical optimization.

The pipeline of RUMP for each iteration is the following:

1) The initial path is produced by interpolation from the current s_{t_1} position towards the global goal s_g until the limits of $H_{s_{t_1}}$ for the first iteration, or with warm starting utilizing the previous solutions for the rest, and prepared for the optimization solver. Additionally, nearby obstacles with their estimated velocities are processed.
2) The optimization problem is solved, utilizing the proposed objective function and constraints.
3) A safe locally-optimal path is produced.
4) The path informs a low-level path tracker to enable the robot to follow the produced trajectory.

A. Assumptions

Firstly, RUMP utilizes path optimization, thus it is bounded by the capacity of the optimization solver to handle local minima that might be encountered. For the entire duration of this section, we assume that the solver can handle the constructed problems. In practice and to our experience, the fast replanning rate achieved in this work, was capable of resolving rare instances of solver failure. For more persistent cases, fast sampling-based techniques could be utilized for warm restarting as proposed in previous work [40].

Secondly, we assume no malicious intent by the moving obstacles, especially the ones moving faster than the AUV. In other words, no fast-moving objects adjust their trajectories to maximize the potential for a collision.

The final assumption is that the AUV is capable of detecting and tracking moving nearby objects early. To quantify this, let ρ^r be the corresponding radius of the robot sphere P_s^r, ρ^o the corresponding radius of P_t^o of an obstacle $o \in O$, d_{detect}^o the minimum necessary distance to track the moving obstacle measured from the collision point, v_r the velocity of robot, v_o the velocity of the dynamic obstacle o, and t_{plan} the replanning period. If the obstacle o is on collision course with the AUV, then the assumption that o will not alter but maintain its present trajectory would represent the most extreme situation. The robot would then need to cover a distance of $\rho^r + \rho^o$ for time t_{avoid} to reach safety, while the obstacle covers the distance d_{detect}^o for t_{obs}. Thus, to guarantee safety, enforcing $t_{avoid} + t_{plan} < t_{obs}$, implies:

$$d_{detect}^o > \frac{\rho^r + \rho^o}{||v_r||} ||v_o|| + t_{plan} ||v_o|| \quad (3)$$

for each moving obstacle o in potential collision with the AUV. The satisfaction of Equation (3) guarantees safe obstacle avoidance by our proposed system for obstacles moving with arbitrary velocities. No further major assumptions are introduced in the paragraphs below, and although beyond the primary scope of this paper, applications of the proposed principles could exceed applications in the marine domain.

B. Proposed Objective Function

To form the optimization problem, the initial path that lies exclusively within $H_{s_{t_1}}$ is represented with n consecutive states, in each replanning step. To encourage uniformity, the states are equally distributed along the path, while the number of states n is computed with the goal of keeping a desired distance between each consecutive state. By expecting optimistically the minimum possible path, if k is the desired distance between each pair of consecutive states and ρ_H the radius of $H_{s_{t_1}}$, the initial number of states is $n = \left\lfloor \frac{\min\{\rho_H, ||s_g - s_{t_1}||\}}{k} \right\rfloor + 1$, where $\lfloor \cdot \rfloor$ is the rounding-down operator. For every next replanning query, the numerator of the previous formula is replaced with the total path length of the previously optimized path, similarly to previous work [49], under the assumption that the expected path length will not significantly change, or that it will adjust quickly, given the fast replanning frequency RUMP achieves.

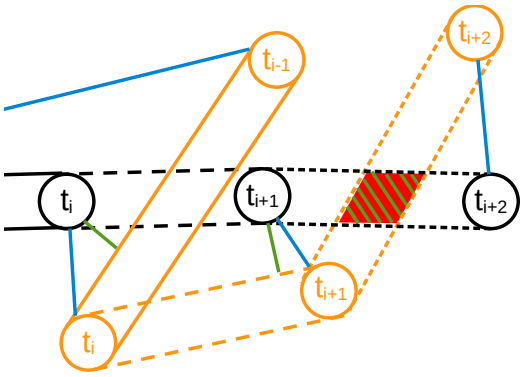

Fig. 2. A 2D instance of a robot (shown with black) and a dynamic obstacle (shown with orange) moving in a common workspace at different moments. The swept-out volumes between different moments are indicated with solid lines for $[t_{i-1},t_i]$, long-dashed lines for $[t_i,t_{i+1}]$, and short-dashed lines for $[t_{i+1},t_{i+2}]$ respectively. If only the distance between corresponding states is taken into account (blue), then due to low resolutions collision may still occur (red-green area). By considering the distance between the swept-out volumes (green) resolution-free safety is guaranteed and potential collisions are detected (red-green area).

Please note that although such assumptions may potentially momentarily affect the desired path quality, it has no effect on path safety.

We use a similar objective function as the one described in Equation (1), using a weight $w > 0$ for the first term:

$$f(S_{t_1}^{t_n}) = w \sum_{i=1}^{n-1} \left\| s_{t_{i+1}} - s_{t_i} \right\|^2 + \left\| s_g - s_{t_n} \right\|^2 . \quad (4)$$

The first term ensures that the optimized path will remain minimal in terms of path length, while the second term assists in producing globally minimal paths, which is discussed more thoroughly in paragraph IV-D. Finally, the variables optimized are $S_{t_2}^{t_n}$, given that the goal is to produce the future corresponding waypoints of the robot, so the initial state t_1 is not considered.

C. Spatiotemporal Collision Constraints

As shown in Section II, past optimization-based approaches dealing with static or dynamic obstacles depended on collision constraints between trajectory states and current or future positions of obstacles, or, at best, either between swept-out volumes of the obstacles and the specific states of the trajectory or swept-out volumes of the trajectory and inflated static obstacles. One of the main contributions of this work is a novel collision constraint formulation between the swept-out volumes of the trajectory with the corresponding swept-out volumes of the moving obstacles. A depiction of the main idea can be shown in Figure 2. In contrast to our contribution, techniques that utilize distances between specific states with inflation would either suffer from collisions [46], [48], not being able to represent to a desired resolution the problem, as shown in Figure 2, or, at best, they will produce unrealistically large clearances that will result to overly conservative solutions and low computational performance [49].

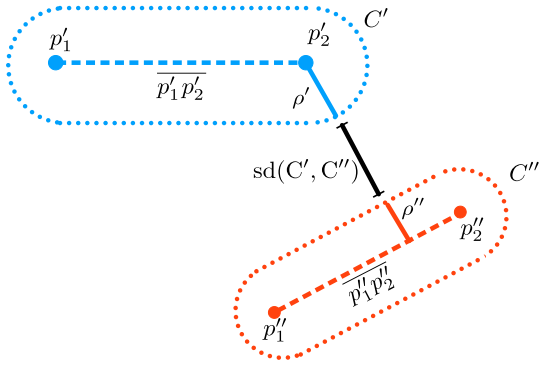

Fig. 3. A 2D projection of an instance of two spherocylinders, along with the computation of their distance, utilizing their corresponding fundamental line segments.

1) General Formulation: Borrowing terminology from the formulation proposed by Schulman *et al.* [13], if A_t is the arbitrary volume occupied by the robot at time $t \in T$ and B_t^o the arbitrary volume at time t for each obstacle $o \in O$, then the proposed general collision constraint to guarantee safety takes the following form, for each $i, j \in \mathcal{N}$ where $0 < i < |T|$ and $0 < j < |O|$:

$$sd(convhull(A_{t_i}, A_{t_{i+1}}), convhull(B_{t_i}^{o_j}, B_{t_{i+1}}^{o_j})) > 0 , \quad (5)$$

where $sd(.)$ represents the signed distance for separating 3D convex objects ([50], [51]), and $convhull$ is the common convex hull of two 3D volumes. The above formulation guarantees that for the corresponding path segments the vehicle will avoid moving obstacles, without employing a conservative policy such as inflating the obstacles, and more importantly, by naturally handling obstacles no matter if they are static or they move many times faster than the robot itself.

2) Simplified Formulation: Although the formulation in Equation (5) introduces a new way for guaranteeing safe, efficient, and non-conservative solutions in dynamic environments with obstacles of arbitrary shapes and velocities, it depends on the resolution of the 3D shapes involved, thus real-time performance could be compromised. Without invalidating the potential of utilizing the general formulation in real-time applications — something left for future work — we chose to limit some of this generality for the sake of simplicity. For this purpose, as mentioned in Section III, both the AUV and the obstacles are assumed to be spheres, a common assumption for motion planners targeting real-time performance in papers presented in Section II.

With this assumption the problem is simplified significantly, since A_{t_i} and $B_{t_i}^{o_j}$ are replaced by $P_{s_{t_i}}^r$ and $P_t^{o_j}$ respectively. Then, the corresponding convex hulls are the spherocylinders $convhull(P_{s_{t_i}}^r, P_{s_{t_{i+1}}}^r)$ and $convhull(P_{t_i}^{o_j}, P_{t_{i+1}}^{o_j})$ and only defining efficiently the signed distance function $sd()$ remains. The volume of each such spherocylinder of radius ρ' between two 3D points p'_1 and p'_2 is equivalent to a cylinder of height $\|p'_1 - p'_2\|$ with fitted hemispheres of radius ρ' on the sides. This implies that the spherocylinder C' could be defined as the collection of all points p' where $p' \in C' \iff \mathrm{dist}(\mathrm{p}', \overline{\mathrm{p}'_1\mathrm{p}'_2}) \leq \rho'$, if $dist(.)$ is the Euclidean distance and $\overline{p'_1p'_2}$ represents the line

segment between the points p'_1 and p'_2. Let C'' be another spherocylinder with ρ'', p''_1 and p''_2, then the signed distance function $sd(.)$ between the two spherocylinders is simply $sd(C', C'') = dist(\overline{p'_1 p'_2}, \overline{p''_1 p''_2}) - \rho' - \rho''$, where for positive values the volumes are separated, for negative values they are in collision, and for 0 they are touching. An example of this analysis is shown in Figure 3. Finally, if p^o_t is the position of an obstacle $o \in O$ at time $t \in T$, the spatiotemporal collision constraint of Equation (5) is simplified:

$$dist(\overline{s_{t_i} s_{t_{i+1}}}, \overline{p^{o_j}_{t_i} p^{o_j}_{t_{i+1}}}) - \rho^r - \rho^{o_j} - \epsilon > 0 \,, \qquad (6)$$

where $\epsilon \geq 0$ is an optional positive parameter representing a minimum clearance to be kept at all times from nearby obstacles.

D. Online Local Goal Computation

Simply pivoting the local goal towards the global goal in the presence of obstacles, could either produce unsafe paths, drive the AUV on a collision trajectory, or necessitate increasing the planning horizon, thus computational expenses, in hopes that a safe local goal could be assigned nearby. In an effort to reduce the complexity of the potential global planner, maintain a desired planning horizon, and at the same time produce paths that are safe in their entirety, we propose determining the local goal online during optimization.

The local goal should be computed in a way that will not introduce discontinuities in the path, drive the robot towards a safe location, minimizes the expected path length to the global goal for efficiency, and maintains the entire trajectory within the planning horizon. Equations (4) and (6) resolve all challenges but the last one. Thus, to ensure that the local goal remains at the boundaries of the planning horizon $H_{s_{t_1}}$, we propose the following additional equality constraint:

$$||s_{t_n} - s_{t_1}|| - \rho_H = 0 \,. \qquad (7)$$

E. Estimating Obstacle Trajectories

Until this point, we have assumed known trajectories for all the obstacles and a method that provides the exact location of the obstacles for the corresponding states of the robot. This subsection describes how to obtain such information, which is needed for applying the constraints shown in Equation (6).

In the absence of more information regarding the specific attributes of the detected obstacles, linear motion is assumed, while any changes in their trajectories could either be accommodated by the high replanning rate of RUMP, or covered by the assumptions of paragraph IV-A. With the above assumption, for two consecutive measurements of the position of an obstacle $o \in O$ at times t' and t'', where $0 < t' < t'' \leq t_1$, corresponding to $p^o_{t'}$ and $p^o_{t''}$ respectively, the estimated velocity could be computed as:

$$v_o = \frac{p^o_{t''} - p^o_{t'}}{t'' - t'} \,. \qquad (8)$$

For each time t with $t > t_1$, the position of the obstacle can be predicted simply:

$$p^o_t = p^o_{t''} + v_o(t - t'') \,. \qquad (9)$$

To compute the necessary times that the obstacles need to be projected given the current state of the optimization, we assume similarly for the robot that it maintains a constant velocity v_r. The corresponding times $t_i \in T$, $0 > i > n$, can be computed solely based on the positions of the waypoints for a path $S = \{s_1, s_2, \ldots, s_n\}$, with $s_j, s_{j+1} \in S$:

$$t_i = t_1 + \frac{\sum_{j=1}^{i-1} \|s_{j+1} - s_j\|}{||v_r||} \,. \qquad (10)$$

V. Experimental Results

In this section, simulation experiments that demonstrate the performance of RUMP are presented. The simulations were performed in FhSim [52], [53], a C++-framework dedicated to simulating marine systems. To solve the numerical optimization problem, we utilized the CasADi library [54] with the nonlinear program solver IPOPT [55]. The robot platform is a remotely operated vehicle (ROV), similar to the Argus Mini ROV (Fig. 1) [56]. It is holonomic and actuated in four degrees of freedom (surge, sway, heave, and yaw). The commanded speed of the robot was 0.5 m/s.

To control the robot, we utilize a 3D line-of-sight (LOS) guidance law [57] and PID control. The guidance law, given as inputs the current configuration of the robot and the next waypoint S_{t_2} of the planned trajectory, calculates the necessary velocity and heading reference signals. The PID controller then calculates the applied forces that enable the robot track these reference signals.

We present results from four case studies:

- A cluttered environment with 12 static obstacles.
- A cluttered environment with 12 dynamic, slow-moving obstacles. The total speed of each obstacle is bounded by the maximum speed of the robot.
- A cluttered environment with 12 dynamic, fast-moving obstacles. Each obstacle can move at a speed three times the maximum speed of the robot.
- An environment with 8 very fast-moving obstacles, each moving with a speed up to 100 times faster than the robot; a number selected with minimal deliberation given that the proposed method is constructed to avoid obstacles moving arbitrary fast. Each obstacle is initialized by moving at a speed and a direction that guarantees collision given the computed robot trajectory when spawned. Thus, deciding a new path fast is essential for the safety of the robot.

We compare RUMP, the new proposed technique utilizing the continuous spatiotemporal constraints from Section IV-C, with a baseline that evaluates collisions only between the states of the trajectory and the corresponding positions of the obstacles, ignoring the in between transitions. We set a desired resolution of $k = 1$ m between consecutive trajectory states, and the planning horizon at $\rho_H = 10$ m for test cases 1-3 and $\rho_H = 50$ m for test case 4.

The simulation results are presented in Figure 4. As expected, it can be seen that the performance of RUMP was identical to the baseline for test case 1, as the effect of

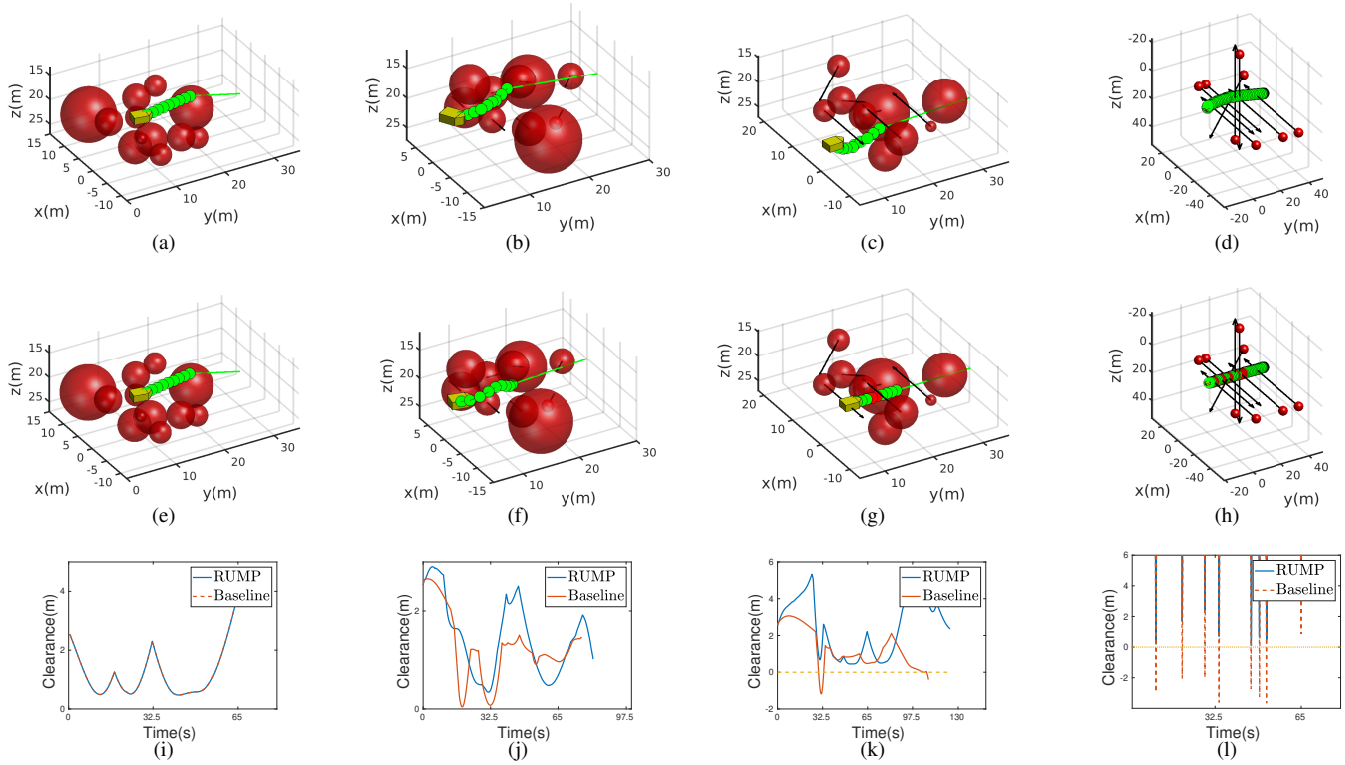

Fig. 4. Results from the simulation studies. Figures 4(a)-4(d) and 4(e)-4(h) display snapshots of the planned trajectory by RUMP, the proposed method, and the baseline respectively. Each columnn corresponds to each test case in numerical order. Green indicates safe states and red states in collision. Figures 4(i)-4(l) display the clearance between the vehicle and the nearest obstacles from instances between different case. Any value bellow 0 indicates collision.

the proposed spatiotemporal collision constraints is naturally degraded to only spatial for static obstacles. Regarding test case 2, the baseline's performance in terms of mission time is slightly better but risky, while RUMP maintains the desired safety distance in all cases. For test cases 3 and 4, RUMP greatly outperforms the baseline planner in terms of safety. As the obstacles move at a higher speed than the vehicle, the baseline planner does not plan trajectories that avoid future collisions, due to dependence on path resolution, as shown in the sketch of Figure 2). RUMP, however, successfully adjusts its path and avoids fast moving obstacles by evaluating spatiotemporal intersections between the planned trajectory and the obstacle trajectories.

Our simulations also showcase the real-time capacity of RUMP. Using a personal laptop (Dell Latitude 7420 with an eight-core, 2.60GHz Intel i5 processor), the average planning frequencies, for the case studies, were in the range of 16-75 Hz. Table I displays the average total mission time and replanning time for a set of ten repetitions of all test cases. The results show that despite the extra computational expense of RUMP, its performance is still comparable to the baseline method and well within real-time requirements. While the mission times were slightly longer for RUMP, this can be explained by RUMPs ability to find efficient collision-free trajectories, in contrast to the unsafe baseline. We aspire that with more implementation improvements, the replanning frequency can be reduced further, allowing for challenging time-sensitive applications exceeding the underwater domain.

		Avg. mission time	Avg. replanning time
Case 1	RUMP	66.91 s ($\pm$ 0.01 s)	0.016 s ($\pm 6.7 * 10^{-4}$ s)
	Baseline	66.90 s ($\pm$ 0.00 s)	0.017 s ($\pm 7.5 * 10^{-4}$ s)
Case 2	RUMP	82.05 s ($\pm$ 0.92 s)	0.047 s ($\pm 9.8 * 10^{-4}$ s)
	Baseline	76.07 s ($\pm$ 0.01 s)	0.017 s ($\pm 6.8 * 10^{-4}$ s)
Case 3	RUMP	119.03 s ($\pm$ 0.83 s)	0.031 s ($\pm 7.0 * 10^{-4}$ s)
	Baseline	108.67 s($\pm$ 0.03 s)	0.012 s ($\pm 4.4 * 10^{-4}$ s)
Case 4	RUMP	88.72 s ($\pm$ 0.10 s)	0.018 s ($\pm 6.0 * 10^{-4}$ s)
	Baseline	82.76 s($\pm$ 0.01 s)	0.013 s ($\pm 5.6 * 10^{-4}$ s)

TABLE I

AVERAGE MISSION TIME AND REPLANNING TIME FOR TEN SIMULATIONS OF EACH TEST CASE. RED INDICATES RUNS WITH COLLISIONS.

VI. CONCLUSIONS

In conclusion, a general motion planning framework for underwater vehicles was developed to solve safe autonomous navigation in challenging dynamic environments. The resulting method, utilizing novel concepts for spatiotemporal collision formulation, can compute in real-time safe and non-conservative paths in workspaces with multiple obstacles moving arbitrarily fast. Simulation experiments demonstrated the scalability of the technique, and its robustness to handle obstacles that may move up to 100 times faster than the AUV itself. At the same time, high performance is achieved with replanning frequency up to 75 Hz, and is invariant to the velocity of the surrounding obstacles. Future work will focus on field testing in industrial-scale fish farms featuring several static and moving objects, investigating extensions towards adapting it for different robotic systems, and handling inspection tasks.

REFERENCES

[1] Y. R. Petillot, G. Antonelli, G. Casalino, and F. Ferreira, "Underwater robots: From remotely operated vehicles to intervention-autonomous underwater vehicles," *IEEE Robotics & Automation Magazine*, vol. 26, no. 2, pp. 94–101, 2019.

[2] A. Shukla and H. Karki, "Application of robotics in offshore oil and gas industry— a review part II," *Robotics and Autonomous Systems*, vol. 75, pp. 508–524, 2016.

[3] L. Yu, E. Yang, P. Ren, C. Luo, G. Dobie, D. Gu, and X. Yan, "Inspection robots in oil and gas industry: a review of current solutions and future trends," in *Proc. 25th International Conference on Automation and Computing (ICAC)*, 2019, pp. 1–6.

[4] M. Føre, K. Frank, T. Norton, E. Svendsen, J. A. Alfredsen, T. Dempster, H. Eguiraun, W. Watson, A. Stahl, L. M. Sunde, C. Schellewald, K. R. Skøien, M. O. Alver, and D. Berckmans, "Precision fish farming: A new framework to improve production in aquaculture," *Biosystems Engineering*, vol. 173, pp. 176 – 193, 2018.

[5] B. Joshi, S. Rahman, M. Kalaitzakis, B. Cain, J. Johnson, M. Xanthidis, N. Karapetyan, A. Hernandez, A. Quattrini Li, N. Vitzilaios, and I. Rekleitis, "Experimental Comparison of Open Source Visual-Inertial-Based State Estimation Algorithms in the Underwater Domain," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Macau, 2019, pp. 7221–7227.

[6] B. Allotta, R. Costanzi, E. Meli, L. Pugi, A. Ridolfi, and G. Vettori, "Cooperative localization of a team of auvs by a tetrahedral configuration," *Robotics and Autonomous Systems*, vol. 62, no. 8, pp. 1228–1237, 2014.

[7] S. Petillo and H. Schmidt, "Exploiting adaptive and collaborative AUV autonomy for detection and characterization of internal waves," *IEEE Journal of Oceanic Engineering*, vol. 39, no. 1, pp. 150–164, 2013.

[8] N. Tsiogkas, G. Frost, N. Monni, and D. Lane, "Facilitating multi-auv collaboration for marine archaeology," in *OCEANS 2015-Genova*. IEEE, 2015, pp. 1–4.

[9] M. Xanthidis, B. Joshi, M. Roznere, W. Wang, N. Burgdorfer, A. Q. Li, P. Mordohai, S. Nelakuditi, and I. Rekleitis, "Towards mapping of underwater structures by a team of autonomous underwater vehicles," in *The International Symposium of Robotics Research*. Springer, 2022, pp. 170–185.

[10] P. Klebert, Øystein Patursson, P. C. Endresen, P. Rundtop, J. Birkevold, and H. W. Rasmussen, "Three-dimensional deformation of a large circular flexible sea cage in high currents: Field experiment and modeling," *Ocean Engineering*, vol. 104, pp. 511 – 520, 2015.

[11] E. Kelasidi and E. Svendsen, "Robotics for sea-based fish farming," in *Encyclopedia of Smart Agriculture Technologies*. Springer, 2023, pp. 1–20.

[12] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.

[13] J. Schulman, Y. Duan, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel, "Motion planning with sequential convex optimization and convex collision checking," *The International Journal of Robotics Research*, vol. 33, no. 9, pp. 1251–1270, 2014.

[14] C. Park, J. Pan, and D. Manocha, "ITOMP: Incremental trajectory optimization for real-time replanning in dynamic environments," in *Twenty-Second International Conference on Automated Planning and Scheduling*, 2012.

[15] S. M. LaValle, *Planning Algorithms*. Cambridge Univ. press, 2006.

[16] J. H. Reif, "Complexity of the mover's problem and generalizations," in *20th Annual Symposium on Foundations of Computer Science*. IEEE, 1979, pp. 421–427.

[17] Y. Guo, H. Liu, X. Fan, and W. Lyu, "Research progress of path planning methods for autonomous underwater vehicle," *Mathematical Problems in Engineering*, vol. 2021, 2021.

[18] M. Panda, B. Das, B. Subudhi, and B. B. Pati, "A comprehensive review of path planning algorithms for autonomous underwater vehicles," *International Journal of Automation and Computing*, vol. 17, no. 3, pp. 321–352, 2020.

[19] R. Kot, "Review of collision avoidance and path planning algorithms used in autonomous underwater vehicles," *Electronics*, vol. 11, no. 15, p. 2301, 2022.

[20] L. Tychonievich, D. Zaret, J. Mantegna, R. Evans, E. Muehle, and S. Martin, "A maneuvering-board approach to path planning with moving obstacles," in *11th international joint conference on Artificial intelligence*, vol. 2, 1989, pp. 1017–1021.

[21] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *The international journal of robotics research*, vol. 17, no. 7, pp. 760–772, 1998.

[22] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23–33, 1997.

[23] W. Zhang, S. Wei, Y. Teng, J. Zhang, X. Wang, and Z. Yan, "Dynamic obstacle avoidance for unmanned underwater vehicles based on an improved velocity obstacle method," *Sensors*, vol. 17, no. 12, p. 2742, 2017.

[24] M. R. Benjamin, M. DeFilippo, P. Robinette, and M. Novitzky, "Obstacle avoidance using multiobjective optimization and a dynamic obstacle manager," *IEEE Journal of Oceanic Engineering*, vol. 44, no. 2, pp. 331–342, 2019.

[25] C. Lin, H. Wang, M. Fu, J. Yuan, and D. Yu, "A convolution neural network based obstacle avoiding method for unmanned underwater vehicle," *ICIC Express Letters*, vol. 13, no. 11, pp. 1079–1086, 2019.

[26] C. Zammit and E. J. Van Kampen, "Real-time 3D UAV path planning in dynamic environments with uncertainty," *Unmanned Systems*, pp. 1–17, 2022.

[27] Y. I. Jenie, E.-J. van Kampen, C. C. de Visser, J. Ellerbroek, and J. M. Hoekstra, "Three-dimensional velocity obstacle method for uncoordinated avoidance maneuvers of unmanned aerial vehicles," *Journal of Guidance, Control, and Dynamics*, vol. 39, no. 10, pp. 2312–2323, 2016.

[28] B. Lindqvist, S. S. Mansouri, A.-a. Agha-mohammadi, and G. Nikolakopoulos, "Nonlinear MPC for collision avoidance and control of UAVs with dynamic obstacles," *IEEE robotics and automation letters*, vol. 5, no. 4, pp. 6001–6008, 2020.

[29] T. Wakabayashi, Y. Suzuki, and S. Suzuki, "Dynamic obstacle avoidance for multi-rotor UAV using chance-constraints based on obstacle velocity," *Robotics and Autonomous Systems*, vol. 160, p. 104320, 2023.

[30] M. Ramezani, H. Habibi, J. L. Sanchez-Lopez, and H. Voos, "UAV path planning employing MPC-reinforcement learning method considering collision avoidance," in *2023 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2023, pp. 507–514.

[31] F. S. Hover, R. M. Eustice, A. Kim, B. Englot, H. Johannsson, M. Kaess, and J. J. Leonard, "Advanced perception, navigation and planning for autonomous in-water ship hull inspection," *The International Journal of Robotics Research*, vol. 31, no. 12, pp. 1445–1464, 2012.

[32] J. McMahon and E. Plaku, "Mission and motion planning for autonomous underwater vehicles operating in spatially and temporally complex environments," *IEEE Journal of Oceanic Engineering*, vol. 41, no. 4, pp. 893–912, 2016.

[33] E. Vidal, M. Moll, N. Palomeras, J. D. Hernández, M. Carreras, and L. E. Kavraki, "Online multilayered motion planning with dynamic constraints for autonomous underwater vehicles," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8936–8942.

[34] J. D. Hernández, E. Vidal, M. Moll, N. Palomeras, M. Carreras, and L. E. Kavraki, "Online motion planning for unexplored underwater environments using autonomous underwater vehicles," *Journal of Field Robotics*, vol. 36, no. 2, pp. 370–396, 2019.

[35] È. Pairet, J. D. Hernández, M. Carreras, Y. Petillot, and M. Lahijanian, "Online mapping and motion planning under uncertainty for safe navigation in unknown environments," *Transactions on Automation Science and Engineering, IEEE*, 2021.

[36] M. S. Wiig, K. Y. Pettersen, and T. R. Krogstad, "A 3D reactive collision avoidance algorithm for underactuated underwater vehicles," *Journal of Field Robotics*, vol. 37, no. 6, pp. 1094–1122, 2020.

[37] T. Manderson, J. C. G. Higuera, S. Wapnick, J.-F. Tremblay, F. Shkurti, D. Meger, and G. Dudek, "Vision-based goal-conditioned policies for underwater navigation in the presence of obstacles," *Robotics: Science and Systems XVI*, Jul 2020.

[38] P. Yang, H. Liu, M. Roznere, and A. Q. Li, "Monocular camera and single-beam sonar-based underwater collision-free navigation with domain randomization," in *The International Symposium of Robotics Research*. Springer, 2022, pp. 85–101.

[39] H. Yu, W. Lu, and D. Liu, "A unified closed-loop motion planning approach for an I-AUV in cluttered environment with localization uncertainty," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 4646–4652.

[40] M. Xanthidis, N. Karapetyan, H. Damron, S. Rahman, J. Johnson, A. O'Connell, J. M. O'Kane, and I. Rekleitis, "Navigation in the presence of obstacles for an agile autonomous underwater vehicle," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 892–899.

[41] B. Sun, W. Zhang, A. Song, X. Zhu, and D. Zhu, "Trajectory tracking and obstacle avoidance control of unmanned underwater vehicles based on mpc," in *2018 IEEE 8th International Conference on Underwater System Technology: Theory and Applications (USYS)*. IEEE, 2018, pp. 1–6.

[42] S. Heshmati-Alamdari, G. C. Karras, P. Marantos, and K. J. Kyriakopoulos, "A robust predictive control approach for underwater robotic vehicles," *IEEE Transactions on Control Systems Technology*, vol. 28, no. 6, pp. 2352–2363, 2019.

[43] X. Yao, X. Wang, F. Wang, and L. Zhang, "Path following based on waypoints and real-time obstacle avoidance control of an autonomous underwater vehicle," *Sensors*, vol. 20, no. 3, p. 795, 2020.

[44] C. S. Tan, R. Sutton, and J. Chudley, "An incremental stochastic motion planning technique for autonomous underwater vehicles," *IFAC Proceedings Volumes*, vol. 37, no. 10, pp. 483–488, 2004.

[45] Y. Zhuang, S. Sharma, B. Subudhi, H. Huang, and J. Wan, "Efficient collision-free path planning for autonomous underwater vehicles in dynamic environments with a hybrid optimization algorithm," *Ocean Engineering*, vol. 127, pp. 190–199, 2016.

[46] A. Cardaillac and M. Ludvigsen, "Ruled path planning framework for safe and dynamic navigation," in *OCEANS 2021: San Diego–Porto*. IEEE, 2021.

[47] J. Hong, K. de Langis, C. Wyethv, C. Walaszek, and J. Sattar, "Semantically-aware strategies for stereo-visual robotic obstacle avoidance," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 2450–2456.

[48] M. Føre, S. Fjæra, S. J. Ohrem, E. Kelasidi, N. Bloecher, and H. B. Amundsen, "Adaptive motion planning and path following for permanent resident biofouling prevention robot operating in fish farms," in *OCEANS 2021: San Diego–Porto*. IEEE, 2021.

[49] M. Xanthidis, E. Kelasidi, and K. Alexis, "Resiplan: Closing the planning-acting loop for safe underwater navigation," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 3138–3145.

[50] E. Gilbert, D. Johnson, and S. Keerthi, "A fast procedure for computing the distance between complex objects in three space," in *International Conference on Robotics and Automation (ICRA)*, vol. 4. IEEE, 1987, pp. 1883–1889.

[51] G. Van Den Bergen, "Proximity queries and penetration depth computation on 3D game objects," in *Game developers conference*, vol. 170, 2001.

[52] K.-J. Reite, M. Føre, K. G. Aarsæther, J. Jensen, P. Rundtop, L. T. Kyllingstad, P. C. Endresen, D. Kristiansen, V. Johansen, and A. Fredheim, "FhSim - time domain simulations of marine systems," in *Proc. ASME 33rd International Conference on Ocean, Offshore and Arctic Engineering*, 2014.

[53] B. Su, K.-J. Reite, M. Føre, K. G. Aarsæther, M. Alver, P. C. Endresen, D. Kristiansen, J. Haugen, W. Caharija, and A. Tsarau, "A multipurpose framework for modelling and simulation of marine aquaculture systems," in *Proc. ASME 38th International Conference on Ocean, Offshore and Arctic Engineering*, 2019.

[54] 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.

[55] A. Wachter and L. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Math. Program*, vol. 106, pp. 25–57, 2006.

[56] M. Skaldebø, H. B. Amundsen, B. Su, and E. Kelasidi, "Modeling of remotely operated vehicle (ROV) operations for aquaculture," in *Proc. IEEE International Conference on Mechatronics and Automation (ICMA)*, 2023, pp. 705–712.

[57] M. Breivik and T. Fossen, "Principles of guidance-based path following in 2D and 3D," in *Proc. 44th IEEE Conference on Decision and Control*, 2005, pp. 627–634.