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

Experimental Comparison of Kinematic Task-Priority Control Methods for an Articulated Intervention-AUV

Bjørn Kåre Sæbø[1†], Markus H. Iversflaten[1†], Jan Tommy Gravdahl[1], Kristin Y. Pettersen[1]

***Abstract*— This work revisits two classical closed-loop inverse kinematics (CLIK) formulations for hierarchical control and investigates their differences in the context of articulated intervention-autonomous underwater vehicles (AIAUVs). The class of AIAUVs consists of free-floating, slender, multi-link vehicles with distributed thrusters and no distinct base, allowing the entire vehicle to be modeled and controlled as a manipulator. The concept of body-velocity sharing, a phenomenon where different tasks depend on overlapping body-frame motions, is introduced and formalized through the notion of body-sensitivity subspaces. Changing the location of the system's body-frame is shown to directly affect both controllers' closed-loop performance, and it shown that due to body-velocity sharing, tasks for AIAUVs most often fall into an intermediate regime between orthogonal and strictly incompatible tasks, causing the two task-priority formulations to differ. The theory is validated through open-water field trials with the Eelume-M, a 6-meter-long AIAUV, comparing the two control laws. The experiments confirm the theoretical predictions: the projected-residual law improves secondary-task tracking but is more sensitive to algorithmic singularities, whereas the post-projection law remains robust to such singularities at the cost of reduced secondary-task performance. These results provide practical guidelines for selecting kinematic task-priority control laws and body-frame placement for AIAUVs.**

I. Introduction

Over recent decades, subsea operations have steadily shifted from divers to teleoperated work-class remotely operated vehicles (ROVs), with relatively rudimentary systems initially taking over hazardous inspection and tooling tasks. Today, the field is moving toward far more capable platforms, underwater vehicle-manipulator systems (UVMSs) and articulated intervention-autonomous underwater vehicles (AIAUVs), that, in principle, can address most inspection, maintenance, and repair (IMR) needs. However, this rise in capabilities demands increasingly sophisticated and specialized control methods. Many foundational methods from terrestrial robotics, such as task-priority inverse kinematics, were transferred to the underwater domain in the 1990s and have matured since then [1], [2]. Yet as new vehicle classes with distinct characteristics emerge, it remains necessary to reevaluate these methods under new conditions.

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme, through the ERC Advanced Grant 101017697-CRÈME. The work is also supported by VISTA - a basic research program in collaboration between the Norwegian Academy of Science and Letters, and Equinor.

[1]Department of Engineering Cybernetics, Norwegian University of Science and Technology, NTNU, NO-7491 Trondheim, Norway {bjorn.k.sabo, markus.h.iversflaten, kristin.y.pettersen, jan.tommy.gravdahl}@ntnu.no.

[†]B.K. Sæbø and M. H. Iversflaten contributed equally to this work and should be considered co-first authors.

Task-priority control addresses scenarios where multiple control objectives must be achieved simultaneously, each with an assigned priority reflecting its relative importance. It is a widely used method across robotic systems, including fixed-base manipulators [3], humanoids [4], and free-floating robots such as UVMSs [1]. These objectives, or *tasks*, are expressed in suitable task coordinates (e.g., end-effector pose). The idea of controlling robots in task coordinates dates back to [5], with the extension to multiple prioritized tasks introduced in [3]. Among the most influential kinematic formulations are the *projected-residual* [3] and *post-projection* [6] laws, which differ in how secondary tasks are resolved within the redundancy left by higher-priority tasks. Comparative studies in other domains exist, for example, [7] in humanoid robots. This paper complements that line of work by focusing on free-floating robots, particularly AIAUVs, comparing the projected-residual and post-projection laws experimentally, and highlighting phenomena that are less pronounced in fixed-base or humanoid settings while remaining consistent with prior findings.

Only a limited number of works have reported experimental studies of task-priority control in underwater robots. Notable examples include the Ocean One humanoid-robotic diver [8], the MARIS project [9], and the TWINBOT project [10]. Earlier experiments specifically on AIAUVs are limited only to [11]. While these studies demonstrate significant progress in applying task-priority control to underwater robotics, they do not consider alternative task-priority formulations, focusing instead on a single approach; consequently, there is no direct comparison of formulations. In particular, no such analysis exists for AIAUVs, where body-frame placement and body-velocity sharing play a decisive role; [11], for instance, focuses mainly on low-level control with only a basic treatment of task-priority aspects. This paper addresses this gap by presenting the first experimental comparison of two classical task-priority formulations on an AIAUV, supported by a theoretical analysis of task compatibility and body-velocity sharing.

AIAUVs are a subclass of UVMSs in which the vehicle itself is an articulated, slender, multi-link structure with thrusters distributed along the body, rather than a heavy hull carrying a separate manipulator. A representative platform is the Eelume-M robot [12], which is a modular, snake-like design that enables efficient transit, as well as fully actuated hovering and precise interaction, effectively treating the entire vehicle as a manipulator without a separate base. For several years, its mathematical model has served as a reference platform for simulation studies of advanced

control methods [13]–[17]. Recently, the vehicle has been made available for broader research use, making the Eelume-M well suited for experimental evaluation of whole-body controllers and task-priority schemes.

In this paper, the problem of kinematic-level task-priority control for AIAUVs is considered. Due to their unique structure, with multiple links of comparable inertia and thrusters distributed along the body, they differ significantly from traditional UVMSs, and thus pose new control challenges. We discuss the placement of the body frame, which is a nontrivial design choice for AIAUVs, and its implications for controller design and performance. This choice determines how task motions are split between the system's body and joint velocities. Practical AIAUV task pairs are often sensitive to the same body-frame motions, leading to what we call body-velocity sharing. This sensitivity makes the choice of kinematic control law consequential, which motivates our focus on two classical formulations: [3] and [6], which we denote the projected-residual and the post-projection methods, respectively. Moreover, we introduce a task-compatibility taxonomy which guides the choice of control law. When tasks fall outside the two extremes (orthogonal or strictly incompatible), a situation common in AIAUVs because the free-floating base appears in most task Jacobians, the two laws can differ significantly. We analyze how body-velocity sharing induces these intermediate regimes and identify when each control law is preferable. Finally, we present results from open-water field trials with the snake-like Eelume-M AIAUV (cf. Fig. 1) to compare the two control laws experimentally.

The remainder of the paper is organized as follows. Section II describes the mathematical model of the AIAUV and discusses body-frame placement. Section III introduces the notion of body-velocity sharing and compares the two kinematic-level task-priority control laws from [3] and [6]. Section IV presents the experimental setup and results and Section V concludes the paper.

II. Mathematical model

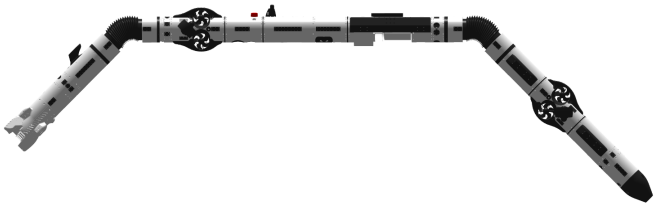

Fig. 1. Render of the Eelume 500 AIAUV (Eelume-M). Its "head" is to the left and its "tail" is to the right.

In this section, we present the kinematic model of UVMSs and then specialize the formulations to the subclass of AIAUVs. Furthermore, we discuss the implications of the placement of the body frame, which is a nontrivial design choice for AIAUVs.

Consider a UVMS with n revolute joints. Its configuration can be described by the triple $\xi := (p, q, \theta)$. It consists of a body frame position $p := p_b^n \in \mathbb{R}^3$ and orientation $q := q_b^n \in \mathbb{S}^3$, as well as a vector of joint angles $\theta \in \mathbb{R}^n$. We denote the body frame by $\{b\}$ and relate it to a local north-east-down (NED) frame $\{n\}$, assumed to be inertial. We express the orientation via the unit quaternion, which we split into its scalar and vector parts as $(\rho, \epsilon) := q$. The body-fixed velocities of a UVMS are denoted $\zeta := (\nu, \dot{\theta}) := (v_{nb}^b, \omega_{nb}^b, \dot{\theta}) \in \mathbb{R}^3 \times \mathbb{R}^3 \times \mathbb{R}^n$, representing the linear, angular, and joint velocities, respectively. We can then express the system's governing kinematics as

$$\dot{\xi} = J(q)\zeta, \tag{1}$$

where $J := J_b^n : \mathbb{S}^3 \to \mathbb{R}^{(7+n)\times(6+n)}$ is the Jacobian matrix

$$J(q) = \begin{bmatrix} R(q) & 0_{3\times3} & 0_{3\times n} \\ 0_{4\times3} & T(q) & 0_{4\times n} \\ 0_{n\times3} & 0_{n\times3} & I_n \end{bmatrix}, T(q) = \frac{1}{2}\begin{bmatrix} -\epsilon^\top \\ \rho I_3 + S(\epsilon), \end{bmatrix}. \tag{2}$$

The matrix $R(q) \in \mathrm{SO}(3)$ describes the rotation from $\{b\}$ to $\{n\}$, $S : \mathbb{R}^3 \to \mathfrak{so}(3)$ is the skew-symmetry map, and I_n is the $n \times n$ identity matrix.

UVMSs are in general actuated by m thrusters and n joint motors. The commanded inputs $u = (u_\eta, u_\theta)$ consist of thruster forces $u_\eta \in \mathbb{R}^m$ and joint torques $u_\theta \in \mathbb{R}^n$. These inputs generate generalized forces and moments $\tau \in \mathbb{R}^{6+n}$ through the map $\tau = B(\theta)u$, where

$$B(\theta) = \begin{bmatrix} B_\eta(\theta) & 0_{6\times n} \\ B_\theta(\theta) & I_n \end{bmatrix} \in \mathbb{R}^{(6+n)\times(m+n)}. \tag{3}$$

The blocks on the diagonal of B comprise the most common actuator mapping for traditional UVMSs. B_η describes how the thrusters affect the body-frame link, while I_n directly maps each commanded joint torque to its associated joint. The block B_θ maps thruster forces to joint torques. This term only appears when there exist thrusters attached to links other than the base. In most traditional UVMSs, $B_\theta = 0$, whereas in AIAUVs it can be nonzero and configuration-dependent. For control, desired generalized forces and moments can be converted to thruster commands and joint torques via

$$u = B(\theta)^\dagger \tau, \tag{4}$$

where $\dagger$ denotes the Moore-Penrose pseudoinverse.

The choice of body frame origin is an inherent part of the kinematic formulations in (1), as all quantities are expressed relative to this frame. While the placement of $\{b\}$ does not alter the physical motion of the system directly, it influences how the control effort is partitioned between the vehicle's base and its manipulator. Thus, the placement of $\{b\}$ is not merely a modeling convention but a design decision with direct implications for controller behavior and closed-loop performance.

III. Kinematic-level task-priority control

In this section, we introduce two classical task-priority methods and compare their theoretical properties. To this end, we develop a task-compatibility taxonomy and relate it to the notion of body-velocity sharing, which is especially pronounced in free-floating robots.

In this context, all kinematic control objectives are referred to as *tasks*. Each task $x_i \in \mathbb{R}^{m_i}$ is defined as the image of the configuration ξ under a mapping $f_i : \mathbb{R}^{7+n} \to \mathbb{R}^{m_i}$:

$$x_i \coloneqq f_i(\xi), \tag{5}$$

where $i \in \mathcal{K} \coloneqq \{1, \ldots, k\}$ indexes a task. The kinematic goal is to find a desired configuration trajectory $\xi_d(t)$ such that $f_i(\xi_d(t)) = x_{i,d}(t),\ \forall i \in \mathcal{K},\ \forall t \geq 0$.

Assumption 1.
The desired task trajectories are defined for all $t \geq 0$, i.e. $x_{i,d}, \dot{x}_{i,d} : [0, \infty) \to \mathbb{R}^{m_i}$, for each $i \in \mathcal{K}$.

We find $\xi_d(t)$ by first resorting to the time derivative of (5), as

$$\dot{x}_i = \frac{\partial f_i(\xi)}{\partial \xi}\dot{\xi} = \frac{\partial f_i(\xi)}{\partial \xi} J(\xi)\zeta \coloneqq J_i(\xi)\zeta, \tag{6}$$

where we use (1). If the task Jacobian J_i is non-singular, we can solve (6) for ζ. Tasks are ordered by priority (with higher priority corresponding to lower index $i \in \mathcal{K}$). The control methods utilize null-space projections to enforce their priorities.

Consider a hierarchy of two tasks, x_1 and x_2. We highlight two methods for calculating the contribution of the secondary task, which are the subject of [6], [7]. The first method, which we refer to as the projected-residual law, originates from early task-priority control work and yields the solution [3], [6, eq. (8)]

$$\zeta_d = J_1^{\dagger}\dot{x}_{1,d} + (J_2 N_1)^{\dagger}(\dot{x}_{2,d} - J_2 J_1^{\dagger}\dot{x}_{1,d}), \tag{7}$$

where $N_1(\xi) \coloneqq I_{6+n} - J_1^{\dagger}J_1$ is the projector onto the null space of J_1. The second method, which we call the post-projection law, can be written as [6, eq. (34)]

$$\zeta_d = J_1^{\dagger}\dot{x}_{1,d} + N_1 J_2^{\dagger}\dot{x}_{2,d}. \tag{8}$$

Since (7) and (8) are prone to drifting when integrated to obtain ξ_d, task feedback is added to the control laws as in [18]. Furthermore, to generalize each control law to accommodate an arbitrary number of tasks, we use [19] and [20]. We choose to represent the extensions for both control laws in a recursive form, as it highlights their differences. The extended forms are then given by

Projected–residual law

$$\begin{cases} \zeta_1 = J_1^{\dagger}\dot{x}_{1,r} & \text{(9a)} \\ \zeta_i = \zeta_{i-1} + (J_i \bar{N}_{i-1})^{\dagger}(\dot{x}_{i,r} - J_i\zeta_{i-1})\,, & \text{(9b)} \end{cases}$$

Post–projection law

$$\begin{cases} \zeta_1 = J_1^{\dagger}\dot{x}_{1,r} & \text{(10a)} \\ \zeta_i = \zeta_{i-1} + \bar{N}_{i-1} J_i^{\dagger}\,(\dot{x}_{i,r})\,. & \text{(10b)} \end{cases}$$

Here, $\dot{x}_{i,r} \coloneqq \dot{x}_{i,d} + \Lambda_i \tilde{x}_i$ with $\Lambda_i \succ 0$ a positive-definite gain matrix for each $i \in \mathcal{K}$, $\tilde{x}_i$ is the minimal-representation task error, and $\bar{N}_i \coloneqq I - \bar{J}_i^{\dagger}\bar{J}_i$ for $\bar{J}_i \coloneqq [J_1^{\top}\ J_2^{\top} \cdots J_i^{\top}]^{\top}$. The reference velocities ζ_d produced by the kinematic controllers need to be tracked by a separate low-level controller, which we describe in Section IV-B. These methods form the basis for the following discussion and for the experimental comparison in Section IV. For clarity, we focus on the case of two tasks, though the arguments extend to more tasks.

It was shown in [6] that the two laws are equivalent iff

$$J_2 J_1^{\dagger} = 0 \quad \text{or} \quad J_2 N_1 = 0$$

which occurs in two cases: either the tasks are orthogonal or strictly incompatible. For free-floating systems, and particularly for AIAUVs, tasks rarely fall into either of these extremes. Instead, most practical task pairs belong to an intermediate regime in which the two laws may differ substantially. An important reason is a phenomenon we term *body-velocity sharing*, where tasks depend on overlapping body-twist directions. In the following, we will first introduce a taxonomy of four compatibility modes, two of which describe this intermediate regime, and relate them to the notion of body-velocity sharing. This taxonomy will then serve as the basis for analyzing the differences between (9) and (10).

Definition 1: Strictly incompatible tasks.
Two tasks x_1 and x_2 are strictly incompatible if the row space of the second task's Jacobian is a subset of the first's, that is $\mathrm{row}(J_2) \subseteq \mathrm{row}(J_1)$. In this case $J_2 N_1 = 0$.

Definition 2: Partially incompatible tasks.
Two tasks x_1 and x_2 are partially incompatible if the row spaces of their Jacobians are linearly dependent, that is $\mathrm{row}(J_1) \cap \mathrm{row}(J_2) \neq \{0\}$. In this case $\mathrm{rank}(J_2 N_1) < \mathrm{rank}(J_2) = m_2$.

Definition 3: Compatible tasks.
Two tasks x_1 and x_2 are compatible if the row spaces of their Jacobians are linearly independent, that is $\mathrm{row}(J_1) \cap \mathrm{row}(J_2) = \{0\}$. In this case $\mathrm{rank}(J_2 N_1) = \mathrm{rank}(J_2) = m_2$.

Definition 4: Orthogonal tasks.
Two tasks x_1 and x_2 are orthogonal if the row spaces of their Jacobians are orthogonal, that is $\mathrm{row}(J_1) \perp \mathrm{row}(J_2)$. In this case $J_2 N_1 = J_2$, and $J_1 J_2^{\top} = J_2 J_1^{\top} = 0$.

Remark 1.
Task compatibility can be configuration-dependent. When that is the case, the row spaces of J_1 and J_2 change with ξ, so the overlap dimension can increase or decrease. This can cause $\mathrm{rank}(J_2 N_1)$ to drop or recover and the task pair x_1, x_2 to move between compatibility modes.

When the secondary task relies solely on motions that would affect the primary task, they are strictly incompatible; the secondary task is completely suppressed by the null-space projection. At the other extreme, if two tasks utilize disjoint sets of system velocities, which means that they cannot affect each other kinematically. In this case, the null-space projection leaves the secondary task entirely unchanged. These modes represent the trivial cases where both control approaches (9) and (10) are equivalent.

Between these two extremes lie two intermediate modes, which are both more common and interesting to analyze for AIAUVs. In these modes, the two controllers exhibit different behaviors. In particular, task pairs that belong to these modes can be either *compatible* or *partially incompatible*. When both tasks are simultaneously achievable despite depending on some of the same system velocities, they are compatible.

On the other hand, when the secondary task can only be partially realized without hindering the primary, the tasks are partially incompatible. Moreover, in this mode the product J_2N_1 is rank deficient, which by definition constitutes an algorithmic singularity. The prevalence of these intermediate modes is primarily due to a phenomenon we term body-velocity sharing, where tasks depend on overlapping body-frame velocities. We formalize this in the following.

Definition 5: Body-velocity sharing.
Partition each task Jacobian as

$$J_i(\xi) = \begin{bmatrix} J_i^b(\xi) & J_i^\theta(\xi) \end{bmatrix}, \quad J_i^b \in \mathbb{R}^{m_i \times 6}, \tag{11}$$

and the body-sensitivity subspace of task i as

$$\mathcal{S}_i \coloneqq \mathrm{row}(J_i^b) \subseteq \mathbb{R}^6, \tag{12}$$

i.e., the set of body-twist elements that affect task i. Two tasks x_1, x_2 *share body velocities* if $\dim(\mathcal{S}_1 \cap \mathcal{S}_2) > 0$. A convenient way to compute this dimension is

$$\dim(\mathcal{S}_1 \cap \mathcal{S}_2) = \mathrm{rank}(J_1^b) + \mathrm{rank}(J_2^b) - \mathrm{rank}\left(\begin{bmatrix} J_1^b \\ J_2^b \end{bmatrix}\right). \tag{13}$$

In floating-base systems, such as AIAUVs, any task expressed in inertial coordinates, such as maintaining or tracking a pose, necessarily depends on body velocities. As a result, tasks often rely on overlapping components of the body twist, meaning that their body-sensitivity subspaces $\mathcal{S}_i$ are typically nontrivial and that $\dim(\mathcal{S}_1 \cap \mathcal{S}_2)$ can be large. This makes strictly orthogonal tasks rare; instead, one more often encounters compatible or partially incompatible tasks when the overlap dimension is large relative to m_2.

The difference between the control laws (9) and (10) can be understood in terms of how they compute the secondary-task contributions. The projected-residual method, (9), solves a constrained least-squares problem, explicitly taking into account the primary task's null space by inverting the product J_2N_1. This inversion makes the method sensitive to algorithmic singularities, i.e. cases where J_2N_1 loses rank even though both J_1 and J_2 are full rank. While potentially problematic, this inversion also enables compensation for the influence of the primary task on the secondary task through the term $J_2J_1^\dagger\dot{x}_{1,d}$. This compensation is not possible in (10), which first computes the unconstrained least-squares solution for the secondary task and only then projects it into the null space of the primary task. That is, the optimization is blind to the directions that will later be removed by the projection. This approach may result in reduced performance for the secondary task during the completion of the primary task. However, it is robust to algorithmic singularities because it avoids inverting the potentially singular product J_2N_1.

In the case of non-orthogonal, compatible tasks, this unconstrained solution of the secondary task contribution can often lead to a suboptimal solution, as the task Jacobian may be significantly modified by the null-space projection. If the unconstrained solution relies heavily on directions that lie outside the null space of the primary task, much of that effort is discarded by the projection. As the product J_2N_1 becomes singular, the post-projection approach remains well-defined. Any incompatible elements of the secondary task will be projected away after a numerically sound inversion of the unconstrained task Jacobian. On the other hand, the projected-residual approach will in these cases attempt to solve an ill-posed optimization problem and may request arbitrarily high system velocities to compensate for the loss of compatibility. At worst, this may lead to a collapse of the strict priority levels, causing a situation where the controller increases the errors in the primary task in an attempt to realize the secondary [6], [7].

In the extreme case of strictly incompatible tasks, $J_2N_1 = 0$, the two methods are theoretically equivalent. Because task Jacobians generally vary with configuration (Remark 1), strict incompatibility is rarely sustained in practice; even small perturbations or numerical effects tend to shift the system away from this regime, except in edge cases where constant Jacobians enforce permanent blocking.

The problem of algorithmic singularities in (9) cannot be eliminated and must instead be managed. This is typically addressed by adding damping to the pseudoinverse calculation $(J_2N_1)^\dagger$. The damped pseudoinverse is defined as

$$A^\# \coloneqq A^\top(AA^\top + \lambda^2 I)^{-1} \tag{14}$$

for some damping factor $\lambda > 0$. This ensures that the matrix to be inverted is always of full rank. However, it comes at the cost of introducing a bias: directions associated with small singular values of A are attenuated, reducing the aggressiveness of secondary-task tracking. While algorithmic singularities only affect the projected-residual law, kinematic singularities, when a task Jacobian J_i loses rank, affect both formulations. Hence, the post-projection method's singularity robustness is limited to algorithmic singularities and does not remove the need for damping at kinematic singularities.

Remark 2.
One might be tempted to consider a composite approach that seemingly combines the benefits of both methods, which is to add the compensation term from (7) into (8), as

$$\zeta_d = J_1^\dagger\dot{x}_{1,d} + N_1J_2^\dagger\left(\dot{x}_{2,d} - J_2J_1^\dagger\dot{x}_{1,d}\right). \tag{15}$$

However, since the top-down compensation is calculated in the full joint space through $J_2^\dagger$ and only afterward projected by N_1, the resulting compensation term may at best only partially eliminate the influence of the higher-priority task, and at worst contribute velocity components that actively counteract the desired motion of the lower-priority task.

Unlike most robot classes, AIAUVs offer an important design parameter in the choice of the body-frame location. For fixed-base manipulators, the base frame is fixed by construction; for work-class ROVs/UVMSs it naturally resides on the hull; for humanoids and legged robots it is standardized on the pelvis or torso; and for aerial or mobile platforms with arms it is attached to the vehicle base. In contrast, the articulated body and distributed thrusters of an AIAUV make it reasonable to place the body frame anywhere along the structure.

The choice of body frame is consequential: it determines which joint coordinates influence a given task. If the body frame is attached to link r and a task x_i is attached to link s, then the joint block J_i^θ of the task Jacobian contains nonzero columns only for the joints that lie on the unique kinematic path $r \to s$. Placing the body frame farther from the task (so that the path $r \to s$ includes more joints), the task effectively gains controllable directions through additional joint columns. Conversely, placing the body frame closer to the task reduces the number of joints on the path, making the task rely more heavily on the body twist ν. Therefore, the placement of the body frame and the definition of the tasks should be considered together, as they directly determine the degree of base reliance and joint involvement.

Remark 3.
Relocating the body frame does not change the dimension of a task's body-sensitivity subspace. Consider a change of the body frame by $g \in SE(3)$ with adjoint $T = \mathrm{Ad}_g^{-1}$, then $J_i^{b'} = J_i^b T$ and $\mathcal{S}_i' = \mathrm{row}(J_i^{b'}) = T^\top \mathcal{S}_i$. Since T is invertible, $\dim(\mathcal{S}_i') = \dim(\mathcal{S}_i)$.

In summary, our analysis indicates that the projected-residual law (9) is preferable in most applications. It provides markedly better secondary-task tracking, especially under body-velocity sharing, while the additional cost is relatively small, since algorithmic singularities can be handled effectively with damping (14). In the *compatible-task* regime, (9) outperforms the post-projection law (10) without drawbacks. This view aligns with the findings of [7]. In the *partially incompatible* regime ($\mathrm{rank}(J_2 N_1) < m_2$), the trade-off becomes notable: damping typically renders (9) reliable while preserving much of its secondary-tracking advantage, whereas (10) offers inherently predictable and safe primary behavior across such trajectories at the expense of secondary performance. Note that kinematic singularities of a task Jacobian affect both formulations and likewise benefit from damping. Accordingly, we recommend using the projected-residual approach by default when accurate secondary tracking matters and damping can be applied to manage singularity effects, reserving the post-projection method for cases where conservative, singularity-robust primary behavior is paramount.

IV. Experiments

This section presents experimental results verifying the discussion in Section III. The results are based on open-water experiments conducted in the Trondheimsfjord in Norway. We begin by presenting the Eelume-M AIAUV and the experimental setup before comparing the two control laws (9) and (10).

A. System description

Here we will briefly describe the configuration and the capabilities of our Eelume-M AIAUV [21].

In its current configuration, the robot is 6 m long and weighs 200 kg, comprising eight thrusters and four revolute joints (see Fig. 1). The thruster configuration enables full 6-DOF control. The thrusters are mounted across two identical modules, each containing four thrusters. Within each module, the thrusters are symmetrically arranged in pairs about the module's center. Two ducted thrusters provide lateral thrust and are vertically offset, while the other two are angled 45°, providing both longitudinal and vertical thrust with lateral offsets. Adding the four joints, the robot has a total of 10 degrees of freedom (DOFs). The revolute joints come in pairs; each joint module contains a pitch-actuated (y-revolute) and yaw-actuated (z-revolute) joint that are nearly superimposed. Joint motions are limited to $\pm 70°$. The joint motors can provide up to 16 Nm of torque, while the thrusters provide up to 50 N of force.

Our Eelume-M AIAUV is equipped with a high-performance navigation system. It fuses measurements from a Doppler velocity log (DVL), three ring-laser gyros, three accelerometers, and more. Combined, they yield accuracies of $0.01°$ for roll and pitch, $0.04°$ for heading, and very low drift in the position estimate.

B. Experimental setup

During experiments, the robot was connected to a topside computer via a fiber-optic tether for real-time control and data collection. To minimize environmental disturbances from waves and salinity variations, the vehicle was operated at a depth of approximately 20 m. Other environmental effects, including sea currents, were not measured directly and were not explicitly compensated for. The two experimental trials were conducted in rapid succession, such that environmental conditions can be assumed comparable.

The experiments were designed to compare the performance of the control laws (9) and (10). The first design decisions are the choice of tasks and the placement of the body frame, which, as discussed in Section III, should be considered together. To highlight the differences between the control laws, we selected tasks with a high degree of body-velocity sharing and placed the body frame in the middle link of the robot, ensuring an equal number of joints between the body frame and each end. Specifically, we chose the tips of the end links as task locations. This makes the tasks largely depend on the same body directions, i.e., $\dim(\mathcal{S}_1 \cap \mathcal{S}_2)$ is large, without any joint coupling. Lastly, controlling the orientation of the body frame was chosen as the third task in order to avoid large deviations from its nominal orientation. The tasks can then be listed as

$$x_1 \coloneqq p_t \in \mathbb{R}^3 \qquad \text{Tail link position} \tag{16a}$$
$$x_2 \coloneqq p_h \in \mathbb{R}^3 \qquad \text{Head link position} \tag{16b}$$
$$x_3 \coloneqq \epsilon \in \mathbb{R}^3 \qquad \text{Body-frame orientation.} \tag{16c}$$

The desired task trajectories for x_1 and x_2 in the xz-plane are shown in Fig. 2. The reference for task x_3 is simply the zero vector. The reference trajectories are relative to the nominal initial configuration of the robot, and from there, both ends follow arcs in the xz-plane, moving down- and inward. The signals start after 10 seconds and then move over a duration of 20 seconds with constant speed. They were designed such that they were always physically reachable

by both ends simultaneously (i.e. the references were never further apart than the length of the robot). Additionally, these references can be tracked in a straightforward manner by using the joints while keeping the body-frame orientation fixed. This provokes very different solutions from the two control laws.

We implemented both control laws (9), and (10) with $\Lambda_i = 0.1\boldsymbol{I}_{m_i},\ \forall i \in \mathcal{K}$. Moreover, constant damping was added to the projected Jacobian inversions in (9), i.e. $\lambda = 0.01$ to attenuate the effects of algorithmic singularities. The tasks generated reference positions and velocities for a lower-level PD+ controller, implemented as

$$\tau = K_p \begin{bmatrix} R(q)^\top (p_d - p) \\ -\mathrm{sgn}(\tilde{\rho})\tilde{\epsilon} \\ \theta_d - \theta \end{bmatrix} + K_d \tilde{\zeta} - \hat{g} \begin{bmatrix} R(q)^T e_3 \\ 0_{3\times 1} \\ 0_{n\times 1} \end{bmatrix}, \quad (17)$$

where $K_p, K_d \succ 0$ are gain matrices, $\tilde{\zeta} = \zeta_d - \zeta$ is the velocity error, $\hat{g} \in \mathbb{R}$ is an estimate of the restoring forces acting on the body frame, and $e_3 := [0\ 0\ 1]^T$. The orientation feedback term $-\mathrm{sgn}(\tilde{\rho})\tilde{\epsilon}$ is taken from [22] where

$$\begin{bmatrix} \tilde{\rho} \\ \tilde{\epsilon} \end{bmatrix} := \begin{bmatrix} \rho_d \rho + \epsilon_d^\top \epsilon \\ \rho \epsilon_d - \rho_d \epsilon + S(\epsilon)\epsilon_d \end{bmatrix}, \ \mathrm{sgn}(\tilde{\rho}) := \begin{cases} 1, & \tilde{\rho} \geq 0 \\ -1, & \tilde{\rho} < 0. \end{cases} \quad (18)$$

This definition of the sgn function avoids an unstable equilibrium point at $\tilde{\rho} = 0$. The controller gains are given by

$$\begin{aligned} K_p &= \mathrm{diag}\big(150,\ 160,\ 120,\ 2\tfrac{180}{\pi},\ 20\tfrac{180}{\pi}, 22\tfrac{180}{\pi},\ 1.5\tfrac{180}{\pi}\ 1_{4\times 1}\big) \\ K_d &= \mathrm{diag}\big(20,\ 0,\ 40,\ 2\tfrac{180}{\pi},\ 0,\ 0,\ \tfrac{180}{\pi}\ 1_{4\times 1}\big). \end{aligned} \quad (19)$$

The gains were chosen based on extensive testing in open water for the straight configuration (i.e. $\theta = 0_{n\times 1}$). Notably, the proportional roll gain is very small, due to the robot's low inertia and damping in roll.

This inertia is also partly the reason why roll is the only rotational DOF with added damping. Pitch and yaw, on the other hand, have large inertias and high innate damping through hydrodynamic drag. The salinity in the fjord varies and we set $\hat{g} = -45$ N based on empirical data on the day of the experiments.

C. Experimental results

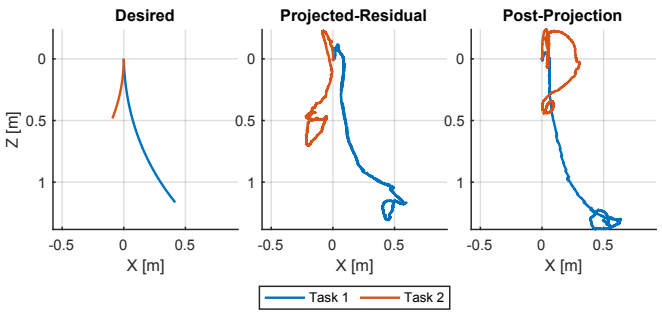

Fig. 2. Relative desired and actual task trajectories in the xz-plane. The trajectories are plotted in a task-relative coordinate frame with the initial task coordinates shifted to the origin

The trajectories of the robot's ends in the xz-plane is seen in Fig. 2. The tracking error norms for each task, using both

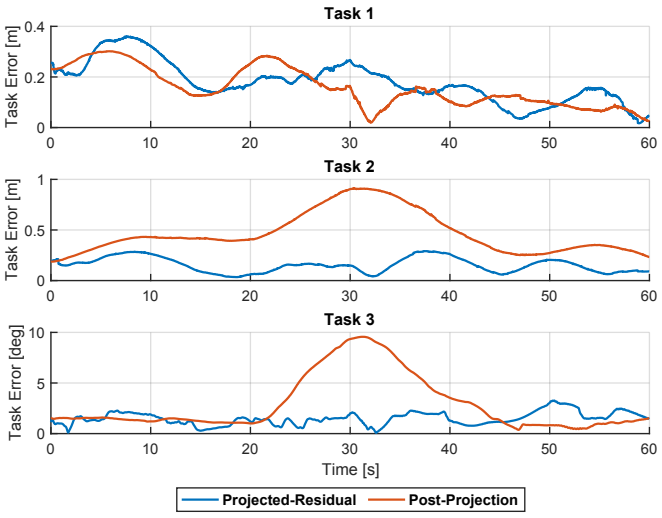

Fig. 3. Task tracking error norms

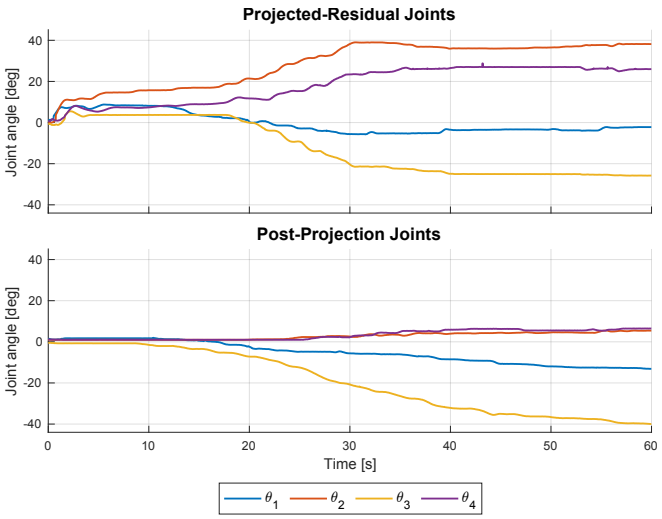

Fig. 4. Joint angles

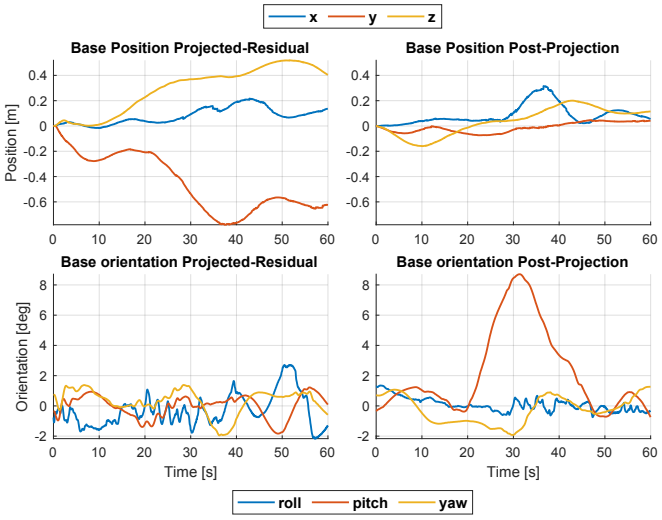

Fig. 5. Position and orientation of the center link

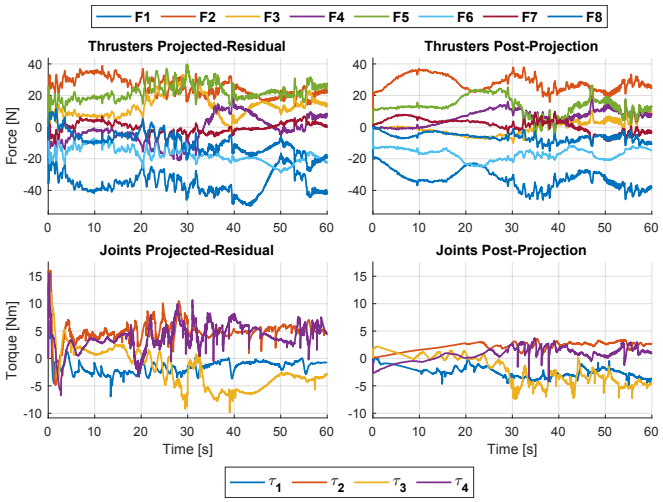

Fig. 6. Control inputs (thruster forces and joint torques)

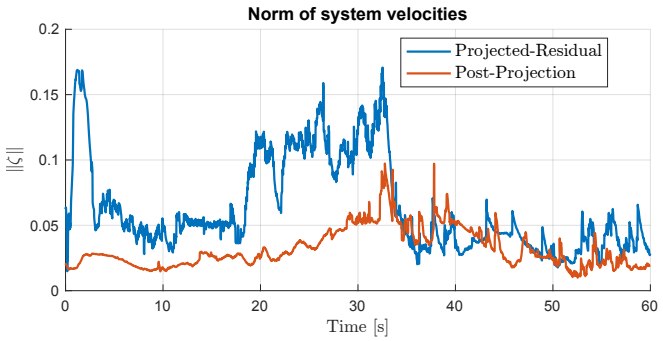

Fig. 7. Euclidian norm of the system velocities

control laws, are shown in Fig. 3. They have very similar initial error norms in all cases but start to differ eventually. We observe that their performance with respect to task x_1 is comparable, with their errors trending downward. However, both tasks x_2 and x_3 show clear differences between the two control laws. The projected-residual law (9) keeps the errors low while the post-projection law (10) struggles to track the secondary tasks. After the reference signal settles (at $t = 30$ s), (10) drives the the secondary task errors toward zero. Together, Figs. 4 and 5 show how ξ evolves for both laws. It is evident that law (9) produces more motion in almost every DOF, and in particular, the base position changes significantly. The control inputs in Fig. 6 show that both laws produce noisy or oscillating commands, though (10) only exhibits this behavior after around 30 seconds. We also note that the commanded joint torque from (9) is briefly saturated at 16 Nm initially. The norm of the system velocities ζ in Fig. 7 are supported by the previous observations: the projected-residual approach produces higher and noisier velocities before the laws converge toward similar behaviors. Lastly, Fig. 8 shows the smallest singular values of the secondary task Jacobian J_2 and its projection onto N_1 for both laws. It is clear that J_2 remains well-posed for the duration of the motion, but its projection has at least one singular value close to zero for some time.

We start by analyzing the discrepancies between (9) and

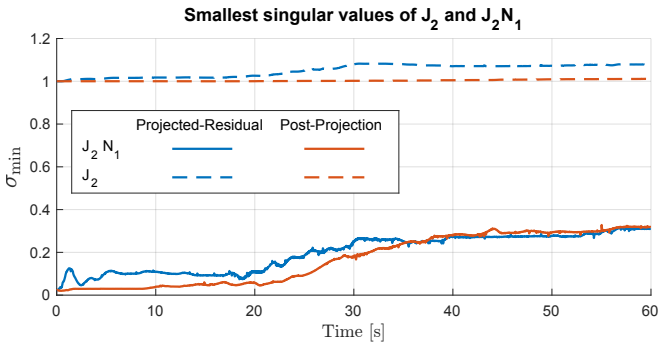

Fig. 8. Minimum singular values of J_2 and J_2N_1

(10) for Figs. 2 and 3. Both laws track $x_{1,d}$ comparably well, but the post-projection law fails to track either of the lower-priority tasks during the transient. The reason is body-velocity sharing: to realize x_1, the controller exploits base motion, most notably pitch, which simultaneously drives x_2 and x_3. This is evident in Fig. 5, where the pitch angle exceeds 8° during the transient. By construction, (10) allows this shared body motion to pass into the lower-priority tasks, whereas (9) can compensate its propagation, whenever sufficient redundancy remains, e.g., by canceling the disturbance introduced by the base pitch used for x_1.

The control inputs in Fig. 6 reveal the projected-residual law's sensitivity to algorithmic singularities, in contrast to the post-projection law. The initial joint-torque spike coincides with a low $\sigma_{min}(J_2N_1)$ (i.e. low smallest singular value of J_2N_1) in Fig. 8, indicating operation near an algorithmic singularity. The initial configuration lies in an algorithmic-singularity neighborhood; (9) then drives the system away via aggressive joint motions, which is supported by Fig. 7. After the transient, the commanded torques decrease as $\sigma_{min}(J_2N_1)$ increases, i.e., as the projected secondary task Jacobian becomes better conditioned. By contrast, (10) maintains moderate torques. The high-frequency content in Fig. 6 can likely be attributed to the low-level controller: its gains are not specifically tuned for large-joint-angle configurations, which causes oscillations that propagate into the control inputs. The joint angles in Fig. 4 support this: (9) reaches high joint angles early and oscillate throughout, while (10) reaches similar angles more slowly, oscillating notably only after 30 seconds.

Note that the large base translation in Fig. 5 under the projected-residual law has a clear explanation. The base position is not a controlled task; its motion is exploited to realize x_1 and x_2. Starting in an algorithmic-singularity neighborhood, the poor conditioning of J_2N_1 induces an unstructured initial transient that manifests as large base and joint motions. Once displaced, the configuration alters the Jacobians $J_i(\xi)$, improves the conditioning of J_2N_1, making additional directions feasible for task execution. This also explains the large yaw-joint angles (θ_2, θ_4): although their columns had little alignment with $\dot{x}_{1,d}$ and $\dot{x}_{2,d}$ at the start, the new configuration increases their task sensitivity, making yaw motion more effective.

Importantly, the present scenario, starting in a poorly conditioned configuration and quickly moving to a better one, can be considered relatively benign. Trajectories that drive the system deeper into singular regions are potentially more challenging for the projected-residual approach (9), as an initial transient will not improve the conditioning of the projected Jacobian's. In such cases, stronger damping or the post-projection method may be required. For the case considered here, however, the projected-residual approach generally outperformed the post-projection, with only a brief problematic transient that remained well within the system's capabilities.

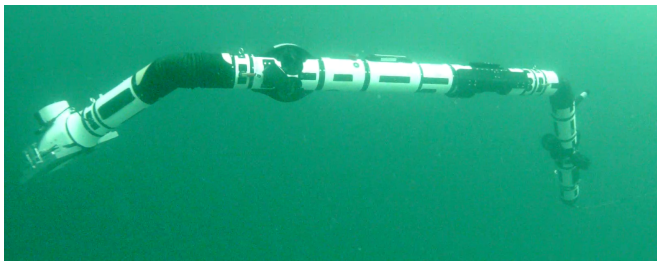

Fig. 9. Eelume-M during the experiments, captured by a Blueye ROV.

V. CONCLUSIONS AND FUTURE WORK

This paper presented the first experimental comparison of two classical task-priority formulations on AIAUVs. We introduced and formalized *body-velocity sharing*, showing that it often leads to an intermediate task-compatibility regime in which tasks are neither fully orthogonal, nor strictly independent. In this regime, which is frequently encountered in AIAUVs, the projected-residual law (9) and the post-projection law (10) yield distinct behaviors. The projected-residual law achieves superior secondary-task tracking and uniquely handles the top-down disturbance term $J_2 J_1^{\dagger} \dot{x}_{1,d}$, but it is sensitive to algorithmic singularities; the post-projection law is more robust to such singularities, at the cost of reduced secondary-task performance. Open-water experiments with the Eelume-M confirmed these predictions: under pronounced body-velocity sharing, (9) tracked secondary objectives more accurately, whereas (10) yielded more predictable behavior near algorithmic singularities. Thus, no single law is universally superior; the choice should be guided by the specific tasks, the degree of body-velocity sharing, and the desired robustness–performance trade-off.

Future work will extend the analysis to more challenging sets tasks and trajectories, including the effects of kinematic singularities, and do further comparisons with other redundancy-resolution approaches and more nuanced methods for adding damping in the presence of singularities.

ACKNOWLEDGMENT

We thank the Applied Underwater Robotics Laboratory (AURLab) at NTNU, and in particular Dr. Pedro de la Torre, for enabling and supporting the experiments. We also thank Håkon Bårsaune for the initial development of code used in the tests.

REFERENCES

[1] G. Antonelli and S. Chiaverini, "Task-priority redundancy resolution for underwater vehicle-manipulator systems," in *Proc. 1998 IEEE Int. Conf. on Robotics and Automation (ICRA)*, Leuven, Belgium, 1998.

[2] E. Simetti, G. Casalino, F. Wanderlingh, and M. Aicardi, "Task priority control of underwater intervention systems: Theory and applications," *Ocean Engineering*, vol. 164, pp. 40–54, Sep. 2018.

[3] Y. Nakamura, H. Hanafusa, and T. Yoshikawa, "Task-Priority Based Redundancy Control of Robot Manipulators," *Int. J. Robotics Research*, vol. 6, no. 2, pp. 3–15, Jun. 1987.

[4] M. Mistry, J. Nakanishi, G. Cheng, and S. Schaal, "Inverse kinematics with floating base and constraints for full body humanoid robot control," in *Proc. 8th IEEE-RAS Int. Conf. Humanoid Robots*, Daejeon, South Korea, Dec. 2008.

[5] D. E. Whitney, "Resolved motion rate control of manipulators and human prostheses," *IEEE Trans. Man-Machine Syst.*, vol. 10, no. 2, pp. 47–53, Jun. 1969.

[6] S. Chiaverini, "Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators," *IEEE Trans. Rob. Autom.*, vol. 13, no. 3, pp. 398–410, Jun. 1997.

[7] P. Baerlocher and R. Boulic, "Task-priority formulations for the kinematic control of highly redundant articulated structures," in *Proc. 1998 IEEE/RSJ Int. Conf. Int. Robots and Systems (IROS).*, Oct. 1998.

[8] G. Brantner and O. Khatib, "Controlling Ocean One: Human–robot collaboration for deep-sea manipulation," *J. Field Robotics*, vol. 38, no. 1, pp. 28–51, Jan. 2021.

[9] E. Simetti, F. Wanderlingh, S. Torelli, M. Bibuli, A. Odetti, G. Bruzzone, D. L. Rizzini, J. Aleotti, G. Palli, L. Moriello, and U. Scarcia, "Autonomous underwater intervention: Experimental results of the MARIS project," *IEEE J. Oceanic Eng.*, vol. 43, no. 3, pp. 620–639, Jul. 2018.

[10] R. Pi, P. Cieslak, P. Ridao, and P. J. Sanz, "TWINBOT: Autonomous underwater cooperative transportation," *IEEE Access*, vol. 9, pp. 37 668–37 684, Mar. 2021.

[11] I.-L. G. Borlaug, K. Y. Pettersen, and J. T. Gravdahl, "The generalized super-twisting algorithm with adaptive gains," *Int. J. Robust and Nonlinear Control*, vol. 32, no. 13, pp. 7240–7270, 2022.

[12] P. Liljebäck and R. Mills, "Eelume: A flexible and subsea resident IMR vehicle," in *Proc. OCEANS 2017*, Aberdeen, Scotland, Jun. 2017.

[13] M. H. Iversflaten, A. Haraldsen, and K. Y. Pettersen, "Kinematic and dynamic control of cooperating underwater vehicle-manipulator systems," *IFAC-PapersOnLine*, vol. 55, no. 31, pp. 110–117, 2022.

[14] E. Tveter, K. Y. Pettersen, and J. T. Gravdahl, "Power-based safety constraint for redundant robotic manipulators," *IFAC-PapersOnLine*, vol. 58, no. 20, pp. 347–353, 2024.

[15] M. H. Iversflaten, B. K. Sæbø, E. A. Basso, K. Y. Pettersen, and J. T. Gravdahl, "Task-priority operational space control for vehicle-manipulator systems with modelling errors," *IFAC-PapersOnLine*, vol. 56, pp. 6703–6709, 2023.

[16] B. K. Sæbø, K. Y. Pettersen, and J. T. Gravdahl, "Robust task-priority impedance control for vehicle-manipulator systems," in *Proc. 2022 IEEE Conf. Control Technology and Applications*, Trieste, Italy, 2022.

[17] M. Wrzos-Kaminska, B. K. Sæbø, K. Y. Pettersen, and J. T. Gravdahl, "Unifying the Generalized Jacobian Matrix and prioritized task hierarchies, with application to free-floating VMSs," *IFAC-PapersOnLine*, vol. 58, no. 20, pp. 79–86, 2024.

[18] L. Sciavicco and B. Siciliano, "A solution algorithm to the inverse kinematic problem for redundant manipulators," *IEEE J. Rob. Autom.*, vol. 4, no. 4, pp. 403–410, Aug. 1988.

[19] B. Siciliano and J. J.-E. Slotine, "A general framework for managing multiple tasks in highly redundant robotic systems," in *Proc. 5th Int. Conf. Advanced Robotics*, Pisa, Italy, 1991.

[20] G. Antonelli, "Stability analysis for prioritized closed-loop inverse kinematic algorithms for redundant robotic systems," *IEEE Trans. Rob.*, vol. 25, no. 5, pp. 985–994, Apr. 2009.

[21] Eelume AS. (2025) Eelume m-series. [Accessed: 2025-09-05]. [Online]. Available: https://www.eelume.com/eelume-m-series

[22] O.-E. Fjellstad and T. Fossen, "Position and attitude tracking of AUV's: A quaternion feedback approach," *IEEE J. Oceanic Eng.*, vol. 19, no. 4, pp. 512–518, Oct. 1994.