Hierarchical compliance-control-based docking of a UVMS

Torje Nysæther
Department of Engineering Cybernetics
NTNU
Trondheim, Norway
torjesn@stud.ntnu.no

Kristin Y. Pettersen
Department of Engineering Cybernetics
NTNU
Trondheim, Norway
kristin.y.pettersen@ntnu.no

Jan Tommy Gravdahl
Department of Engineering Cybernetics
NTNU
Trondheim, Norway
jan.tommy.gravdahl@ntnu.no

***Abstract*—The viability of underwater vehicle-manipulator systems (UVMS) in underwater intervention missions is heavily reliant on safety. For safe operations, a UVMS should specifically be able to both avoid dangers and interact safely with the environment. While task-priority controllers have been extensively used to incorporate set-based safety-tasks for a UVMS, only recent controllers can ensure safe interaction of several tasks by shaping the impedance of the system. In this paper, a hierarchical impedance-based controller is extended with set-based tasks in a practical setting, to obtain a controller that can achieve both these desirable properties. To verify, the proposed solution is applied on an articulated intervention-AUV in a docking use case with an obstacle-avoidance task in simulation.**

I. Introduction

In the past decades, the interest for autonomous underwater operations has steadily increased, especially in areas such as marine biology, military, and offshore energy. Today, a large part of these operations are performed by remotely operated vehicles (ROV), which are generally costly, prone to human errors, and require extensive infrastructure to operate [1]. Although AUV technology has made tremendous leaps in survey-related mission, intervention-based tasks such as maintenance and repair remain open problems in practice. Therefore, more research into underwater vehicle-manipulator systems (UVMS) and controllers capable of handling interaction tasks, also with practical considerations, is needed.

During an intervention mission, a UVMS usually has many tasks to fulfil with different levels of importance. These can include primary tasks, such as turning a valve, safety-related tasks, such as obstacle avoidance, or optimization tasks, such as minimizing energy consumption. As a UVMS is kinematically redundant to most of these tasks, it is beneficial to use a task-priority controller. In such controllers, the different tasks are solved at distinct priority levels, so that a task never interferes with the execution of a higher level task.

During a mission, safe behavior should be ensured such that the robot does not harm itself or its surroundings. In particular, both excessive interaction forces and dangerous configurations should be avoided. To limit dangerous interaction forces, the system dynamics are commonly shaped by torque control. This is especially important for vehicles with complex dynamics, like an articulated intervention-AUV (AIAUV), as seen in Figure 1.

For torque control of redundant manipulators with several task definitions, the Operational Space Formulation (OSF) [2] is considered state-of-the-art. The controller can fully shape the system dynamics, but requires accurate force measurements to do so, and is also sensitive to modeling errors [3]. Although this may be negligible for other robots, underwater vehicles are susceptible to complex hydrodynamic forces that make both modeling and force sensing difficult. Force sensing is, however, not needed in a particular form of impedance control called compliance control. Here, the natural inertia is preserved, while the damping and stiffness can still be shaped. A task-priority compliance tracking controller is proposed in [4], and later extended to a UVMS in [5].

In a task-priority framework, a safe configuration is usually implemented with set-based tasks, where the goal is to keep the robot within the feasible task space. For a UVMS, common set-based safety tasks can include obeying field-of-view constraints, joint limits, or minimum distance to the seabed or underwater structures [6]. Although set-based tasks are available for many newer task-priority frameworks, this is not the case for the hierarchical compliance controller [4]. For the controller to be a viable alternative also for a UVMS, also for safety critical missions, the formulation should be extended with set-based tasks.

In this work, we want to utilize the strong properties of the hierarchical compliance controller, and we extend this to handle set-based tasks. Specifically, we propose to ”emulate” set-based tasks by activating regular tasks if a safe set is compromised. This is motivated by the approach in [6], where task activations are determined by an extended tangent cone. Here, the set-based tasks are only activated when strictly necessary, thus deteriorating other tasks as little as possible. Moreover, the proposed controller is applied in a docking maneuver, a safety-critical use case where restrained system dynamics are important. Emphasis is placed on how set-based tasks can be included to achieve obstacle avoidance while

This result is part of a project that 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 the Research Council of Norway through the Centres of Excellence funding scheme, project No. 223254 – NTNU AMOS, and by VISTA – a basic research program in collaboration between The Norwegian Academy of Science and Letters, and Equinor.

completing a terminal docking of an AIAUV.

The paper is structured as follows. First, the mathematical model of the UVMS is presented in Section II. Then, the hierarchical compliance controller is presented in Section III. The controller is further extended to emulate set-based tasks in Section IV. In Section V, the controller is applied to a practical use case of docking an AIAUV. The corresponding simulation results are presented in Section VI. Finally, conclusions and future work are presented in Section VII.

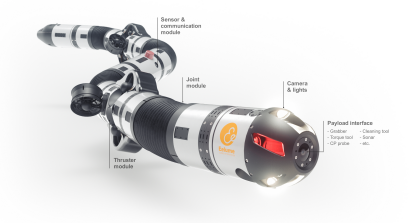

Fig. 1: An AIAUV made by Eelume.

II. Mathematical model

In this section, the general model of a UVMS is presented, as follows from [7]. A UVMS is a hovering underwater vehicle, with a base and a manipulator arm with n links. The state of the robot is defined in the inertial frame as

$$\boldsymbol{\xi} = \begin{bmatrix} \boldsymbol{\eta} \\ \boldsymbol{\theta} \end{bmatrix} \in \mathbb{R}^{6+n} \tag{1}$$

where $\boldsymbol{\eta} \in \mathbb{R}^6$ is the base pose in inertial frame and $\boldsymbol{\theta} \in \mathbb{R}^n$ the manipulator joint angles. The system velocities are defined in the base frame as

$$\boldsymbol{\zeta} = \begin{bmatrix} \boldsymbol{\nu} \\ \dot{\boldsymbol{\theta}} \end{bmatrix} \in \mathbb{R}^{6+n} \tag{2}$$

where $\boldsymbol{\nu}^\top = [\mathbf{v}^\top \quad \boldsymbol{\omega}^\top]$ is the velocity of the base frame, and $\mathbf{v}, \boldsymbol{\omega} \in \mathbb{R}^3$ the linear and angular component, respectively. The system of equations are given by

$$\begin{aligned} &\dot{\boldsymbol{\xi}} = \mathbf{T}_{\boldsymbol{\theta}}(\boldsymbol{\xi})\boldsymbol{\zeta} \\ &\mathbf{M}(\boldsymbol{\theta})\dot{\boldsymbol{\zeta}} + \mathbf{C}(\boldsymbol{\theta}, \boldsymbol{\zeta})\boldsymbol{\zeta} + \mathbf{D}(\boldsymbol{\theta}, \boldsymbol{\zeta})\boldsymbol{\zeta} + \mathbf{g}(\boldsymbol{\theta}, \boldsymbol{\eta}) = \boldsymbol{\tau} + \boldsymbol{\tau}^{\text{ext}} \end{aligned} \tag{3}$$

where $\mathbf{T}_{\boldsymbol{\theta}}(\boldsymbol{\xi}) \in \mathbb{R}^{(6+n)\times(6+n)}$ is the mapping between the system velocities in the inertial and base frame, $\mathbf{M}(\boldsymbol{\theta}) \in \mathbb{R}^{(6+n)\times(6+n)}$ is the inertia matrix, $\mathbf{C}(\boldsymbol{\theta}, \boldsymbol{\zeta}) \in \mathbb{R}^{(6+n)\times(6+n)}$ is the Coriolis matrix, $\mathbf{D}(\boldsymbol{\theta}, \boldsymbol{\zeta}) \in \mathbb{R}^{(6+n)\times(6+n)}$ is the matrix of the hydrodynamic damping, $\mathbf{g}(\boldsymbol{\theta}, \boldsymbol{\eta}) \in \mathbb{R}^{(6+n)}$ is the vector of generalized gravity forces and buoyancy, $\boldsymbol{\tau} \in \mathbb{R}^{(6+n)}$ is the control torque, and $\boldsymbol{\tau}^{\text{ext}} \in \mathbb{R}^{(6+n)}$ is the external torque on the system. Furthermore, the following properties are assumed:

$$\begin{aligned} &\dot{\mathbf{M}}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) - 2\mathbf{C}(\boldsymbol{\theta}, \boldsymbol{\zeta}) = -(\dot{\mathbf{M}}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) - 2\mathbf{C}(\boldsymbol{\theta}, \boldsymbol{\zeta}))^\top \\ &\mathbf{M}(\boldsymbol{\theta}) = \mathbf{M}(\boldsymbol{\theta})^\top > 0 \\ &\mathbf{z}^\top \mathbf{D}(\boldsymbol{\theta}, \boldsymbol{\zeta})\mathbf{z} > 0 \;\forall\; \mathbf{z} \in \mathbb{R}^{6+n} \\ &||\mathbf{D}(\boldsymbol{\theta}, \boldsymbol{\zeta}_a) - \mathbf{D}(\boldsymbol{\theta}, \boldsymbol{\zeta}_b)|| \leq \mathbf{D}_M ||\boldsymbol{\zeta}_a - \boldsymbol{\zeta}_b|| \end{aligned} \tag{4}$$

where $\mathbf{D}_M$ is some constant matrix. For the controller [5] it is also assumed that the nonlinear part of the hydrodynamic damping is negligible.

III. The hierarchical compliance controller

In this section, the key aspects of the hierarchical compliance controller [5] are presented. Consider a set of r prioritized tasks $\mathbf{x}_i \in \mathbb{R}^{m_i}$ in the task space. Each task is given by a mapping from the configuration space of the system,

$$\mathbf{x}_i = f_i(\boldsymbol{\xi}) \in \mathbb{R}^{m_i} \;\forall\; 1 \leq i \leq r \tag{5}$$

with task velocities

$$\begin{aligned} \dot{\mathbf{x}}_i &= \mathbf{J}_i'(\boldsymbol{\xi})\dot{\boldsymbol{\xi}}, \;\; \mathbf{J}_i'(\boldsymbol{\xi}) = \frac{\partial f_i(\boldsymbol{\xi})}{\partial \boldsymbol{\xi}} \\ \dot{\mathbf{x}}_i &= \mathbf{J}_i(\boldsymbol{\xi})\boldsymbol{\zeta}, \;\; \mathbf{J}_i(\boldsymbol{\xi}) = \mathbf{J}_i'(\boldsymbol{\xi})\mathbf{T}_{\boldsymbol{\theta}}(\boldsymbol{\xi}) \end{aligned} \tag{6}$$

in the inertial and base frame, respectively. The task accelerations are given by

$$\ddot{\mathbf{x}}_i = \mathbf{J}_i(\boldsymbol{\xi})\dot{\boldsymbol{\zeta}} + \dot{\mathbf{J}}_i(\boldsymbol{\xi})\boldsymbol{\zeta}. \tag{7}$$

The goal of the controller is to shape the damping and stiffness of each task while ensuring a dynamically consistent strict execution between tasks on different priority levels. This means that a task should never affect the performance of higher-level tasks. To achieve these properties, two assumptions about the tasks definitions are made:

1) All task are simultaneously feasible, and the task dimension is equal to the degrees of freedom(DOF) of the system:
$$\sum_{i=1}^{r} m_i = 6 + n \tag{8}$$
2) There are no singularities in the workspace.

To ensure strict priority between tasks while shaping the impedance of each task level independently, the control of each $\mathbf{x}_i$ must be decoupled. This is done by transforming all r tasks into a new state space where each task level can be independently controlled. The first step is to define the augmented Jacobians $\mathbf{J}_i^{\text{aug}}(\boldsymbol{\xi})$ at each task level, given by

$$\begin{aligned} \mathbf{J}_i^{\text{aug},\top}(\boldsymbol{\xi}) &= [\mathbf{J}_1^\top(\boldsymbol{\xi}) \; \mathbf{J}_2^\top(\boldsymbol{\xi}) \; ... \; \mathbf{J}_i^\top(\boldsymbol{\xi})] \\ \dot{\mathbf{x}}_i^{\text{aug}} &= \mathbf{J}_i^{\text{aug}}(\boldsymbol{\xi})\boldsymbol{\zeta}. \end{aligned} \tag{9}$$

To decouple the tasks and ensure strict priority, the augmented Jacobians of each task are mapped onto a dynamically consistent null-space $\mathbf{N}_i(\boldsymbol{\xi})$, creating an extended Jacobian $\bar{\mathbf{J}}_i(\boldsymbol{\xi})$ given by

$$\begin{aligned} &\bar{\mathbf{J}}_i(\boldsymbol{\xi}) = \mathbf{J}_i(\boldsymbol{\xi})\mathbf{N}_i(\boldsymbol{\xi})^\top \\ &\mathbf{N}_i(\boldsymbol{\zeta}) = \begin{cases} \mathbf{I}_{6+n}, & i = 1 \\ \mathbf{I}_{6+n} - \mathbf{J}_{i-1}^{\text{aug}}(\boldsymbol{\xi})^\top \mathbf{J}_{i-1}^{\text{aug}}(\boldsymbol{\xi})^{M+,\top}, & \text{otherwise} \end{cases} \end{aligned} \tag{10}$$

where $\mathbf{J}_{i-1}^{\text{aug}}(\boldsymbol{\xi})^{M+,\top}$ is the dynamically consistent pseudoinverse of the augmented Jacobian [8]. The new, decoupled tasks $v_i \in \mathbb{R}^{m_i}$ are then given by

$$\begin{aligned} v = \begin{bmatrix} v_1 \\ \vdots \\ v_r \end{bmatrix} &= \begin{bmatrix} \bar{\mathbf{J}}_1(\boldsymbol{\xi}) \\ \vdots \\ \bar{\mathbf{J}}_r(\boldsymbol{\xi}) \end{bmatrix} \boldsymbol{\zeta} = \bar{\mathbf{J}}(\boldsymbol{\xi})\boldsymbol{\zeta} \\ \dot{v} &= \dot{\bar{\mathbf{J}}}(\boldsymbol{\xi})\boldsymbol{\zeta} + \bar{\mathbf{J}}(\boldsymbol{\xi})\dot{\boldsymbol{\zeta}}. \end{aligned} \tag{11}$$

Inserted into the system dynamics (3), this gives us the decoupled system

$$\bar{\mathbf{M}}(\boldsymbol{\theta})\dot{v} = \bar{\mathbf{J}}^{-\top}(\boldsymbol{\xi})(\boldsymbol{\tau}+\boldsymbol{\tau}^{\text{ext}}-\mathbf{g}(\boldsymbol{\theta})-(\boldsymbol{\mu}(\boldsymbol{\theta},\boldsymbol{\zeta})+\boldsymbol{\delta}(\boldsymbol{\theta},\boldsymbol{\zeta})))v \quad (12)$$

where the transformed inertia, Coriolis and damping matrices are respectively given by

$$\begin{aligned} \bar{\mathbf{M}} &= \bar{\mathbf{J}}^{-\top}\mathbf{M}\bar{\mathbf{J}}^{-1} \\ \boldsymbol{\mu} &= \bar{\mathbf{J}}^{-\top}(\mathbf{C} - \mathbf{M}\bar{\mathbf{J}}^{-1}\dot{\bar{\mathbf{J}}})\bar{\mathbf{J}}^{-1} \\ \boldsymbol{\delta} &= \bar{\mathbf{J}}^{-\top}\mathbf{D}\bar{\mathbf{J}}^{-1} \end{aligned} \quad (13)$$

where the arguments are removed for readability. Note that $\bar{\mathbf{M}}$ is a block-diagonal, decoupled matrix, while $\boldsymbol{\mu}$ and $\boldsymbol{\delta}$ have coupling terms between the different task levels. To fully decouple the system, the controller

$$\boldsymbol{\tau} = \mathbf{g} + \boldsymbol{\tau}_{\boldsymbol{\mu}} + \boldsymbol{\tau}_{\boldsymbol{\delta}} + \sum_{i=1}^{r} \mathbf{N}_i\mathbf{J}_i\mathbf{F}_{i,\text{ctrl}} \quad (14)$$

is applied, where $\boldsymbol{\tau}_{\boldsymbol{\mu}}$ and $\boldsymbol{\tau}_{\boldsymbol{\delta}}$ cancels the cross-terms in $\boldsymbol{\mu}$ and $\boldsymbol{\delta}$, respectively, and $\mathbf{F}_{i,\text{ctrl}}$ is the virtual control input of each task to shape the compliance of the task space. The decoupled system can then be converted back to the original task space while preserving the decoupling by

$$\bar{\mathbf{M}}_i\ddot{\mathbf{x}}_i + (\boldsymbol{\mu}_i + \boldsymbol{\delta}_i)\dot{\mathbf{x}}_i + \boldsymbol{\gamma}_i \begin{bmatrix} \dot{\mathbf{x}}_{i-1}^{\text{aug}} \\ \ddot{\mathbf{x}}_{i-1}^{\text{aug}} \end{bmatrix} = \mathbf{F}_{i,\text{ctrl}} + \mathbf{F}_{v_i}^{\text{ext}} \quad (15)$$

where $\boldsymbol{\gamma}_i$ contains the top-down disturbances of the augmented higher prioritized tasks. Applying the controller

$$\mathbf{F}_{i,\text{ctrl}} = \bar{\mathbf{M}}\ddot{\mathbf{x}}_i^{\text{des}} + (\boldsymbol{\mu}_i+\boldsymbol{\delta}_i)\dot{\mathbf{x}}_i^{\text{des}} - \mathbf{D}_i\dot{\tilde{\mathbf{x}}}_i - \mathbf{K}_i\tilde{\mathbf{x}}_i + \boldsymbol{\gamma}_i \begin{bmatrix} \dot{\mathbf{x}}_{i-1,\text{des}}^{\text{aug}} \\ \ddot{\mathbf{x}}_{i-1,\text{des}}^{\text{aug}} \end{bmatrix} \quad (16)$$

where $\mathbf{D}_i$ and $\mathbf{K}_i$ are positive definite, tunable matrices to shape the damping and stiffness of the dynamics, $\dot{\mathbf{x}}_i^{\text{des}}, \ddot{\mathbf{x}}_i^{\text{des}}$ represent the reference trajectory of each task, and $\tilde{\mathbf{x}}_i, \dot{\tilde{\mathbf{x}}}_i$ the error between the reference and the actual task value. Finally, the closed loop dynamics of each task level are given by

$$\bar{\mathbf{M}}_i\ddot{\tilde{\mathbf{x}}}_i + (\boldsymbol{\mu}_i+\boldsymbol{\delta}_i)\dot{\tilde{\mathbf{x}}}_i + \mathbf{D}_i\dot{\tilde{\mathbf{x}}}_i + \mathbf{K}_i\tilde{\mathbf{x}}_i + \boldsymbol{\gamma}_i \begin{bmatrix} \dot{\tilde{\mathbf{x}}}_{i-1}^{\text{aug}} \\ \ddot{\tilde{\mathbf{x}}}_{i-1}^{\text{aug}} \end{bmatrix} = \mathbf{F}_{v_i}^{\text{ext}}. \quad (17)$$

As stated in Assumptions 1 and 2, the controller requires all tasks to be simultaneously feasible, with no singularities in the workspace. Such assumptions impose more rigid requirements than many other task-priority frameworks, but are not uncommon in controllers with stronger properties [4]. Null-space-based task-augmented controllers like this are, however, vulnerable to singular tasks [8], and their definitions must be carefully designed. Singularities can be both kinematic and algorithmic and arise when the extended Jacobian $\bar{\mathbf{J}}$ becomes singular. Kinematic singularities appear whenever a task $\mathbf{x}_i$ loses controllability due to the system configuration, which causes the task-Jacobian $\mathbf{J}_i$ to lose rank. On the other hand, algorithmic singularities arise when tasks on different priority levels conflict with each other. Then, the individual task Jacobians are linearly dependent, which results in loss of rank in the extended Jacobian, even if each individual task Jacobian has full rank. To use this controller, the designer thus needs to ensure that these singularities never occur in the workspace. The controller could also be extended with singularity robustness properties used in other task-priority controllers [4], but this will degrade the tracking performance, and are therefore not considered here.

IV. Extending to set-based tasks

To incorporate common UVMS safety tasks such as obstacle avoidance, the chosen controller is in this section extended with set-based task capabilities. The approach is heavily inspired by [9], and is here adapted to accommodate torque control. To introduce the set-based task for the hierarchical compliance controller, let us first revisit the definition of a single set-based task from [9]. Consider a scalar task variable $\sigma \in \mathbb{R}$ with a safe set D, where $\sigma \in D$ if $\sigma_{\min} \leq \sigma \leq \sigma_{\max}$. The extended tangent cone of the safe set is then given by

$$T_{\mathbb{R},D}(\sigma) = \begin{cases} [0,\infty), & \sigma \leq \sigma_{\min} \\ \mathbb{R}, & \sigma \in (\sigma_{\min}, \sigma_{\max}) \\ (-\infty, 0], & \sigma \geq \sigma_{\max}. \end{cases} \quad (18)$$

The set-based task is activated whenever its *absence* would make $\dot{\sigma} \notin T_{\mathbb{R},D}(\sigma)$. This will only happen when *both* $\sigma \notin D$ and the task velocity $\dot{\sigma}$ is heading away from D. This is intuitive; if the safety is already compromised, but the system moves toward safety anyway, it would be unnecessary to potentially impair other active tasks. Task activation is determined by calculating the control input assuming the task to be inactive, using the result and the task Jacobian (6) to calculate a *prior* task velocity $\dot{\sigma}_{\text{pri}}$, and finally checking whether $\dot{\sigma}_{\text{pri}} \in T_{\mathbb{R},D}$ or not.

In a torque-controlled system, a control input $\boldsymbol{\tau}$ will not affect $\dot{\sigma}$, but rather the task acceleration $\ddot{\sigma}$. The tangent cone approach from [9] is therefore extended, by using an acceleration prior $\ddot{\sigma}_{\text{pri}}$ instead to determine whether task activation is necessary to conserve the safe set or not. The prior is used to ensure that the system is never compromised while traveling along the border of the extended tangent cone (18), i.e. when $\sigma \notin D \cap \dot{\sigma} = 0$. To find $\ddot{\sigma}_{\text{pri}}$, a prior of the system accelerations $\dot{\boldsymbol{\zeta}}_{\text{pri}}$ must be found. This can be done by computing the desired control torque $\boldsymbol{\tau}$ with an inactive task and then solving for $\dot{\boldsymbol{\zeta}}_{\text{pri}}$ in the system equations (3). As $\mathbf{M}(\boldsymbol{\theta})$ is invertible (4), a solution is always available. Furthermore, $\ddot{\sigma}_{\text{pri}}$ can be found by inserting $\dot{\boldsymbol{\zeta}}_{\text{pri}}$ into (7). Note that this only works if $\boldsymbol{\tau}^{\text{ext}}$ is known, either by measurements or by assuming free movement where $\boldsymbol{\tau}^{\text{ext}} = 0$. The new set-based task activation scheme can be summarized in Algorithm 1, where the task is activated whenever *False* is returned.

Another option would be to extend the tangent cone once more, allowing the system to be outside the previous tangent cone $T_{\mathbb{R},D}(\sigma)$ whenever $\ddot{\sigma}$ is heading in the right direction. However, since a safety task usually is aggressively tuned to maintain the safety requirements, a task activation would likely produce a desirable higher "good" acceleration than whenever the system conveniently accelerates in the feasible direction.

As in all torque-controlled systems, a positional state like σ cannot be instantaneously controlled, opposed to inverse

Algorithm 1: Boolean activation function for a set-based task σ

Input: $\ddot{\sigma}_{\text{pri}}, \dot{\sigma}, \sigma, \sigma_{\min}, \sigma_{\max}$

if $\sigma_{\min} < \sigma < \sigma_{\max}$ **then**
  return True
else if $\sigma \leq \sigma_{\min}$ *and* $\dot{\sigma} > 0$ **then**
  return True
else if $\sigma \geq \sigma_{\max}$ *and* $\dot{\sigma} < 0$ **then**
  return True
else if $\sigma \leq \sigma_{\min}$ *and* $\dot{\sigma} = 0$ *and* $\ddot{\sigma}_{\text{pri}} \geq 0$ **then**
  return True
else if $\sigma \geq \sigma_{\max}$ *and* $\dot{\sigma} = 0$ *and* $\ddot{\sigma}_{\text{pri}} \leq$ **then**
  return True
else
  return False

kinematic based methods like [9]. A "safe zone" around D is therefore needed, to avoid dangerous configurations. However, defining this "safe zone" is not trivial. Revisiting the closed-loop dynamics of a task $\mathbf{x}_i$ (17), all the system matrices are configuration dependent and will thus change over time. The required size of a "safe zone" will therefore depend not only on the state and its derivatives, but also on how the physical system evolves. Consequently, a safe set D must be defined with task derivatives and system matrices in mind. This will be further discussed in Section V.

Another limitation is Assumption 1 in the controller, which requires a task space dimension equal to the dimension of the system. Consequently, when a set-based task is activated, another task must leave the task space. How this can be solved is application-dependable, but many highly redundant robots like a UVMS usually have "excessive" DOF not used in primary control, which easily can be swapped out for a high-priority set-based task. This will be illustrated in the case study in the next section.

V. Case study: AIAUV docking operation

A. Use case description

To demonstrate the proposed control approach, it is applied to a docking use case with an articulated intervention-AUV (AIAUV). An AIAUV is a class of UVMS, where the body size is similar to each of the manipulator links; see Fig. 1. This gives the robot more operational flexibility than normal in, for instance, confined spaces, but also makes it harder to control due to its strong couplings between the motion of its base and arm. Further descriptions can be found in [10]. In this use case, we consider an AIAUV with five links, connected by joints of two DOF. For modeling purposes, each joint is modeled as two joints with one DOF each, connected by a short link. As follows, the system has eight joints in total.

In the use case, the AIAUV docks towards a docking station (DS). The DS is modeled as the desired end-effector configuration in the inertial frame, representing the pose of the DS connector. To navigate to the DS, the AIAUV is visually guided, as is commonly done in the underwater docking literature. A camera is mounted on the end-effector of the AIAUV, pointing in the same direction as the dock connector placed on the vehicle. Four visual markers are placed around the DS connector in the vertical plane, a usual arrangement for a funnel-based DS. These markers can be mapped to the image frame of the camera by a calibrated pinhole model, and used to guide the robot. Additionally, an artificial light source is placed on the base link of the AIAUV to minimize camera backscattering. Therefore, the AIAUV should approach the DS in a C-shape, see Figure 2, to be able to point both the camera and the light source towards the DS.

In essence, the vehicle has two objectives, steering the camera/end-effector towards the DS connector and maintaining a C-shape configuration. This is implemented as two tasks in the hierarchical compliance controller, a high-level end-effector task $\mathbf{x}_c \in \mathbb{R}^6$ and a lower-level task $\mathbf{x}_\theta \in \mathbb{R}^8$ to regulate the eight AIAUV joints.

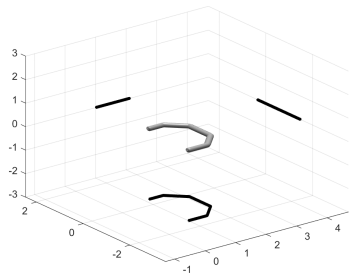

Fig. 2: An AIAUV in a C-shape.

Additionally, a potentially cluttered environment is assumed around the DS. To illustrate, a set-based obstacle-avoidance task $x_{\text{obs}} \in \mathbb{R}$ is defined for the base link. This could also trivially be extended to the other links. Whenever activated, x_{obs} has the highest rank in the task hierarchy to ensure safety at all times.

B. Visual servoing

To generate a desired task trajectory to guide the AIAUV towards the DS connector, image-based visual servoing (IBVS) is used. The method is mature and can be found in most textbooks on visual control, such as [11].

In IBVS, the goal is to generate a camera velocity reference $\boldsymbol{\nu}_c \in \mathbb{R}^6$ by controlling a number of (normalized) image features $\mathbf{p}_i \in \mathbb{R}^2$ to their desired image locations $\mathbf{p}_i^* \in \mathbb{R}^2$. In this use case, $\mathbf{p}_i^*$ represents the image position of each marker in the docked configuration, and can typically be found by testing or estimated with a known marker placement on the DS. Four feature points are assumed in this work.

To generate a camera trajectory from the image feature errors, $\tilde{\mathbf{p}}_i = \mathbf{p}_i^* - \mathbf{p}_i$, a relation between the camera velocity and the errors is needed. The following equations are sufficient for this purpose:

$$\underbrace{\begin{bmatrix}\dot{x}_i\\ \dot{y}_i\end{bmatrix}}_{\dot{\mathbf{p}}_i} = \underbrace{\begin{bmatrix}-\frac{1}{Z_i} & 0 & \frac{x_i}{Z_i} & x_iy_i & -(1+x_i^2) & y_i\\ 0 & -\frac{1}{Z_i} & \frac{y_i}{Z_i} & 1+y_i^2 & -x_iy_i & -x_i\end{bmatrix}}_{\mathbf{E}_i}\boldsymbol{\nu}_c \quad (19)$$

$$\boldsymbol{\nu}_c = \mathbf{E}^\dagger\dot{\mathbf{p}}, \quad \underbrace{\begin{bmatrix}\dot{\mathbf{p}}_1\\ \vdots\\ \dot{\mathbf{p}}_n\end{bmatrix}}_{\dot{\mathbf{p}}} = \underbrace{\begin{bmatrix}\mathbf{E}_1\\ \vdots\\ \mathbf{E}_n\end{bmatrix}}_{\mathbf{E}}\boldsymbol{\nu}_c. \quad (20)$$

The matrix $\mathbf{E}$ is often called the image Jacobian, $\mathbf{E}^\dagger$ is the Moore-Penrose inverse, and Z_i is the distance in Z-direction to the point $\mathbf{P}_i$ in the camera frame that is projected onto $\mathbf{p}_i$ in the image plane.

To be able to move to the desired camera configuration, the feature errors have to converge to zero. This is enforced by the proportional controller

$$\dot{\mathbf{p}} = \lambda(\mathbf{p}^* - \mathbf{p}) \quad (21)$$

where λ is the controller gain. Finally, by inserting (21) into (20), the camera velocity needed to execute the desired feature error dynamics from (21) can be found:

$$\boldsymbol{\nu}_c = \lambda\mathbf{E}^\dagger(\mathbf{p}^* - \mathbf{p}). \quad (22)$$

Note that $\boldsymbol{\nu}_c = \dot{\mathbf{x}}_c^{\text{des}}$ is the *velocity* reference of the task. The desired task acceleration $\ddot{\mathbf{x}}_c^{\text{des}}$ is further found by numerical differentiation of $\boldsymbol{\nu}_c$, and $\tilde{\mathbf{x}}_c$, the error between the desired and real position, is set to zero in (17) since the task velocity is generated on-line.

C. Obstacle avoidance

To avoid collision, the set-based obstacle-avoidance task is activated whenever the distance between the obstacle and the vehicle base becomes too small. The obstacle is modeled as a sphere with radius d_r and position $\mathbf{p}_o \in \mathbb{R}^3$ in the inertial frame. Collision is therefore avoided whenever

$$\sigma_{\text{obs}} \geq d_r, \quad \sigma_{\text{obs}} = \sqrt{(\mathbf{p}_o - \mathbf{p}_b)^\top(\mathbf{p}_o - \mathbf{p}_b)} \in \mathbb{R} \quad (23)$$

where σ_{obs} is the distance between the base and obstacle center, and $\mathbf{p}_b = \boldsymbol{\eta}_{1,2,3} \in \mathbb{R}^3$ is the position of the base.

The distance σ_{obs} is used for task activation as described in Section IV, with a corresponding safe set

$$D = [d_{\min}, \infty) \quad (24)$$

where $d_{\min} \in \mathbb{R}$ is a lower bound for how close the base can approach the obstacle and be still certain that task activation will prevent collision, i.e. (23) holds. A tight bound for $d_{\min}$ is difficult to find without analyzing the reachability of the system, for instance by using methods similar to those used in input-constrained control barrier functions [12]. Conservative estimates can, however, be determined practically, for instance with a "worst-case" estimate of the task dynamics. To find such an estimate, consider the obstacle-avoidance task x_{obs}. This is a regulation task with a desired task position σ_{obs}, and velocity and acceleration references equal to zero. Assuming no external forces, the error dynamics (17) are given by

$$\bar{M}_{\text{obs}}\ddot{\tilde{x}}_{\text{obs}} + (\mu_{\text{obs}} + \delta_{\text{obs}} + D_{\text{obs}})\dot{\tilde{x}}_{\text{obs}} + K_{\text{obs}}\tilde{x}_{\text{obs}} = 0. \quad (25)$$

As $\bar{M}_{\text{obs}} > 0$ [4] and K_{obs}, D_{obs} are tunable, lower bounds on $\bar{M}_{\text{obs}}$, μ_{obs}, δ_{obs} are sufficient to bound how far the system will travel with an active obstacle-avoidance task before the task velocity becomes zero. Such bounds can be found by considering the operational constraints of the vehicle in a given configuration. Furthermore, $d_{\min}$ can be found by determining a minimum initial value of σ_{obs} where the system (25) with conservative parameters avoids collision.

D. Joint regulation

When the obstacle-avoidance task is inactive, the joint angle task $\mathbf{x}_\theta$ regulates the vehicle to the desired C-shape, guided by the reference $\boldsymbol{\theta}_C \in \mathbb{R}^8$. However, when the obstacle-avoidance task is activated, the task dimension of the joint angles must be reduced by 1, as discussed in Section IV. To do so, a natural choice is to shrink the task space by removing the joint number k which is closest to its desired joint value, meaning

$$k = \arg\min_i |\theta_i - \theta_{C,i}|, \quad 1 \leq i \leq n. \quad (26)$$

The active set of controlled joints are therefore given by

$$\boldsymbol{\theta}_a = [\theta_1 ... \theta_i ... \theta_n]^\top, \quad 1 \leq i \leq n, \quad i \neq k. \quad (27)$$

To avoid unnecessary chattering in the task space, a small offset $\theta_\delta > 0$ is also introduced in the implementation, such that k only changes from its previous value k_p if

$$|\theta_{k_p} - \theta_{C,k_p}| > \theta_\delta + \min_i |\theta_i - \theta_{C,i}|, \quad 1 \leq i \leq n, \quad i \neq k_p. \quad (28)$$

VI. Simulation results

The proposed framework is simulated in the Eelume AIAUV simulator in Matlab Simulink [10], and extended with a simple visual framework. Seven thrusters are used to control the vehicle, in addition to the servos in the joints. Each thruster and joint can, respectively, produce a maximum of 60 N and 20 Nm. The task impedances are tuned to obey the control input restrictions, as saturation will dismantle the strict priority between the different tasks.

The results of the docking maneuver are given in Figures 3-7. In Figure 3, the Euclidean norm of the the feature errors are is shown. As the norm goes towards zero, the AIAUV accordingly docks towards the desired goal state. This can also be seen in the feature-point plot in Figure 4, where the blue feature points $\mathbf{p}$ converge towards the desired $\mathbf{p}*$ in red.

The plots related to the set-based obstacle-avoidance task can be seen in Figure 5. As expected, the set-based obstacle-avoidance constraint in red is violated. However, the *actual* safety limit in green is never compromised, and the AIAUV thus maintains safety at all times. In the middle plot, the inactive joint angle task can be seen. Noticeably, the excluded joint number changes over time, even when the obstacle-avoidance task is active. From the bottommost plot it is evident that the obstacle-avoidance task can be inactive even though D is violated. Interestingly, the solution travels along the

extended tangent cone at the end, as the task is repeatedly reactivated, as seen in the middle plot in Figure 5.

By comparing Figures 3 and 5, it is apparent that the object avoidance task activation only causes small ripples in the feature-error norm. The AIAUV is accordingly able to avoid an obstacle while simultaneously executing another task. However, the repeated task activation causes the control input to saturate, which can be seen in Figure 6. This chattering is also a problem in [9], and is later solved by smoothing the task activation [13], which could be an alternative here as well. While the obstacle-avoidance and the IBVS-task achieve acceptable performance while coexisting, this is however not the case for the joint-regulation task in Figure 7, which starts in the desired configuration and then deteriorates. As an accurate C-shape is not a core priority and lowest in the hierarchical design, this is acceptable. Furthermore, the system is well behaved at all times. For more complex operations, more delicate attention to singularities is still needed.

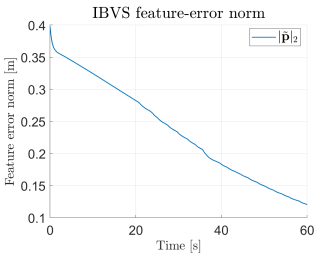

Fig. 3: The Eucledian norm of the feature-point errors.

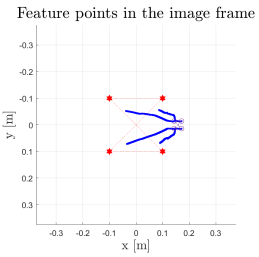

Fig. 4: The feature points in the image frame over time.

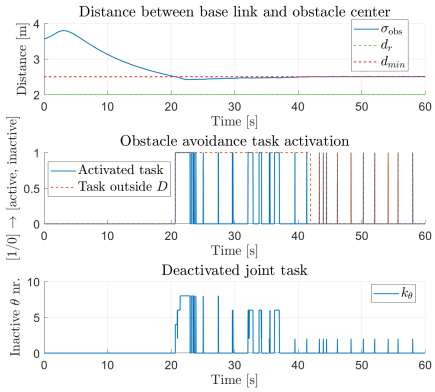

Fig. 5: The distance between the base and the obstacle center, the obstacle-avoidance task activation, and the index of the inactive joint task, respectively.

VII. Conclusions and Future Work

For the hierarchical compliance controller to be viable in complex underwater intervention-missions, common set-based safety tasks are needed. In this paper, the controller is extended with such capabilities. The control approach is demonstrated through a simulated use case, where an articulated intervention-AUV performs a docking maneuver while avoiding an obstacle. Both tasks are successfully executed, with only a small performance loss in the docking task when the obstacle-avoidance task is activated. Future work includes more complex intervention scenarios, physical experiments, and improving when to activate the safety task.

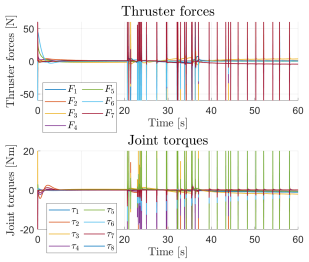

Fig. 6: The forces and torques produced by the AIAUV.

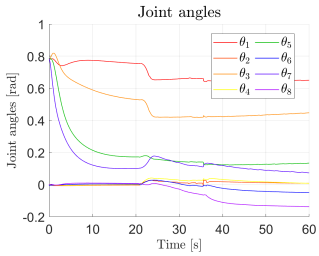

Fig. 7: The joint angles.

References

[1] P. Ridao, M. Carreras, D. Ribas, P. J. Sanz, and G. Oliver, "Intervention auvs: The next challenge," *Annual Reviews in Control*, vol. 40, pp. 227–241, 2015.

[2] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *IEEE Journal on Robotics and Automation*, vol. 3, no. 1, pp. 43–53, 1987.

[3] C. Ott, A. Dietrich, and A. Albu-Schäffer, "Prioritized multi-task compliance control of redundant manipulators," *Automatica*, vol. 53, pp. 416–423, 2015.

[4] A. Dietrich and C. Ott, "Hierarchical Impedance-Based Tracking Control of Kinematically Redundant Robots," *IEEE Transactions on Robotics*, vol. 36, no. 1, pp. 204–221, Feb. 2020.

[5] B. K. Sæbø, K. Y. Pettersen, and J. T. Gravdahl, "Robust Task-Priority Impedance Control for Vehicle-Manipulator Systems," in *Proc. 2022 IEEE Conference on Control Technology and Applications (CCTA)*, Aug. 2022, pp. 363–370.

[6] 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.

[7] G. Antonelli, T. I. Fossen, and D. R. Yoerger, "Modeling and control of underwater robots," *Springer Handbook of Robotics*, pp. 1285–1306, 2016.

[8] A. Dietrich, C. Ott, and A. Albu-Schäffer, "An overview of null space projections for redundant, torque-controlled robots," *The International Journal of Robotics Research*, vol. 34, no. 11, pp. 1385–1400, 2015.

[9] S. Moe, G. Antonelli, A. R. Teel, K. Y. Pettersen, and J. Schrimpf, "Set-Based Tasks within the Singularity-Robust Multiple Task-Priority Inverse Kinematics Framework: General Formulation, Stability Analysis, and Experimental Results," *Frontiers in Robotics and AI*, vol. 3, 2016.

[10] H. M. Schmidt-Didlaukies, A. J. Sørensen, and K. Y. Pettersen, "Modeling of Articulated Underwater Robots for Simulation and Control," in *2018 IEEE/OES Autonomous Underwater Vehicle Workshop (AUV)*, Nov. 2018, pp. 1–7.

[11] P. Corke, *Robotics, Vision and Control*, ser. Springer Tracts in Advanced Robotics. Cham: Springer International Publishing, 2017, vol. 118.

[12] T. Gurriet, A. Singletary, J. Reher, L. Ciarletta, E. Feron, and A. Ames, "Towards a framework for realizable safety critical control through active set invariance," in *2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS)*. IEEE, 2018, pp. 98–106.

[13] S. Moe, J. T. Gravdahl, and K. Y. Pettersen, "Set-based control for autonomous spray painting," *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 4, pp. 1785–1796, 2018.