Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnLine 56-2 (2023) 6703–6709

Task-Priority Operational Space Control for Vehicle-Manipulator Systems with Modelling Errors ⋆

Markus H. Iversflaten†∗ Bjørn Kåre Sæbø†∗ Erlend A. Basso∗
Kristin Y. Pettersen∗ Jan Tommy Gravdahl∗

∗ *Centre for Autonomous Marine Operations and Systems (NTNU AMOS), Department of Engineering Cybernetics, Norwegian University of Science and Technology, Trondheim, Norway (e-mail: {markus.h.iversflaten, bjorn.k.sabo, erlend.a.basso, kristin.y.pettersen, jan.tommy.gravdahl}@ntnu.no)*

Abstract: The dynamics of underwater vehicle-manipulator systems (UVMSs) are very hard to model, which reduces the feasibility of model-based control approaches. Even so, such strategies prove useful in redundancy resolution. In this paper, higher-order sliding mode control is combined with task-priority operational space control (OSC) in order to handle and utilize the redundancy of UVMSs despite the presence of dynamic model errors. At each task level, the generalized super-twisting algorithm is implemented to reject effects caused by model errors while maintaining a continuous control signal. The general problem of OSC with uncertain models is analyzed, and some of its challenges are highlighted, including an algebraic loop. We conduct a simulation study on a highly redundant UVMS, where we compare task-level higher-order sliding mode control to proportional-derivative control. Though this paper is motivated by challenges specific to UVMSs, the results also hold for other vehicle-manipulator systems.

Keywords: Marine robotics, control design, task-priority control, sliding mode control, operational space control

1. INTRODUCTION

Underwater vehicle-manipulator systems (UVMSs) are generally kinematically redundant with respect to their end-effector configuration. This redundancy can be leveraged in order to execute additional objectives simultaneously. Depending on the system at hand, it is preferable to resolve the redundancy either at the kinematic or the dynamic level. For systems with sufficiently quick dynamics, it is tractable to resolve the redundancy at the kinematic level. Conversely, for systems with slower dynamics, such an approach is generally ill-advised. Additionally, some classes of UVMSs suffer from significant coupling forces caused by joint motions (Borlaug et al., 2022). In order to instead resolve the redundancy on the dynamic level, it is possible to apply the results of Khatib et al. (2004) which extended the well-known operational space formulation (OSF) (Khatib, 1987) to a task-priority framework. In order to achieve effective redundancy resolution of UVMSs, we wish to apply task-priority operational space control (OSC). However, OSC relies heavily on knowledge of the kinematics and dynamics of the system. There exist many systems in which an accurate model cannot be obtained, particularly with respect to their dynamic parameters. It is in general not possible to maintain strict dynamic priority in the task hierarchy in this case.

The existing literature on this topic is limited, especially regarding theoretical results. A comparison of OSC strategies is given in Nakanishi et al. (2008), where the effects of modelling errors are mentioned but only analyzed empirically. Adaptive control has been applied to improve the model knowledge in task-priority methods in Lee et al. (2019) and Garofalo et al. (2021). The effects of imperfect model knowledge can also be dealt with through robust control, e.g. by using sliding mode control (SMC). In Slotine et al. (1988), first-order SMC is applied for an end-

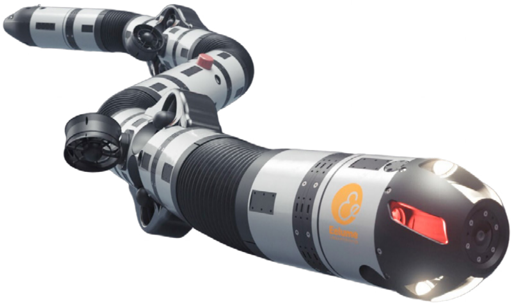

Fig. 1. The Eelume AIAUV, a highly redundant UVMS

⋆ 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.

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

10.1016/j.ifacol.2023.10.374

effector regulation task in the presence of uncertainties. To the authors' knowledge, there exists no literature on higher-order SMC in OSC. In Antonelli et al. (2018); Di Lillo et al. (2021), the effects of dynamic model errors in task-priority OSC is investigated, both with compatible and incompatible tasks. It is shown that only the restoring forces affect steady-state errors of a task. The work considers regulation control, and is limited to two tasks, implemented on a fixed-base robot. The present work is motivated by extending this analysis to task trajectory tracking for vehicle-manipulator systems (VMSs).

In this paper, the task-priority OSC problem with an arbitrary number of tasks is formulated for UVMSs subjected to dynamic model parameter uncertainties. The resulting task-level dynamics in the presence of imperfect cancellation of the system dynamics is analyzed for an arbitrary number of tasks. The analysis reveals that the dynamics suffer from an algebraic loop when the system inertia estimate is erroneous. An attempt to achieve task trajectory tracking in spite of this is made by implementing a super-twisting algorithm (STA) at each task level. The STA attenuates chattering compared to conventional SMC, which is important for systems with slow dynamics, while retaining robustness. Lastly, a simulation study with an articulated intervention autonomous underwater vehicle (AIAUV) (cf. Fig. 1) is conducted and supports the presented theory. The AIAUV is a subclass of UVMSs which is particularly challenging to model correctly. The results of this paper hold also for other VMSs in general.

The paper is organized as follows. The mathematical model of a general UVMS is presented in Section 2. In Section 3, the OSC problem is described with and without modelling errors. Furthermore, a task-level super-twisting controller is proposed to provide robustness to dynamic model errors. A theoretical stability analysis is given in Section 4. The proposed method is validated through a simulation study in Section 5. Lastly, Section 6 presents conclusions and future work.

2. MATHEMATICAL MODEL

The equations of motion for a UVMS can be expressed in a body-fixed reference frame as (From et al., 2014)

$$\dot{\boldsymbol{\xi}} = \boldsymbol{J}_{\boldsymbol{\xi}}(\boldsymbol{q})\boldsymbol{\zeta} \tag{1a}$$

$$\boldsymbol{M}(\boldsymbol{\theta})\dot{\boldsymbol{\zeta}} + \boldsymbol{C}(\boldsymbol{\theta},\boldsymbol{\zeta})\boldsymbol{\zeta} + \boldsymbol{D}(\boldsymbol{\theta},\boldsymbol{\zeta})\boldsymbol{\zeta} + \boldsymbol{g}(\boldsymbol{\xi}) = \boldsymbol{\tau}. \tag{1b}$$

The system configuration of a UVMS equipped with an n-joints manipulator arm is defined as $\boldsymbol{\xi} \triangleq \left[\boldsymbol{\eta}^T, \boldsymbol{\theta}^T\right]^T$, where $\boldsymbol{\eta} \triangleq \left[(\boldsymbol{p}_{IB}^I)^T, \boldsymbol{q}^T\right]^T \in \mathbb{R}^7$ is the position and attitude of the base frame with respect to the inertial frame and $\boldsymbol{\theta} \in \mathbb{R}^n$ is the vector of joint angles. The attitude is parametrized by the quaternion $\boldsymbol{q} \triangleq \left[\eta, \boldsymbol{\epsilon}^T\right]^T \in \mathbb{R}^4$ such that $\|\boldsymbol{q}\|_2 = 1$, where η denotes the real part and $\boldsymbol{\epsilon}$ denotes the vector part of $\boldsymbol{q}$. The body-fixed velocities are denoted $\boldsymbol{\zeta} \triangleq \left[(\boldsymbol{v}_{IB}^I)^T, (\boldsymbol{\omega}_{IB}^I)^T, \dot{\boldsymbol{\theta}}^T\right]^T$, where $\boldsymbol{v}_{IB}^I \in \mathbb{R}^3$ are the linear velocities, $\boldsymbol{\omega}_{IB}^I \in \mathbb{R}^3$ are the angular velocities, and $\dot{\boldsymbol{\theta}} \in \mathbb{R}^n$ are the joint velocities. The body-fixed and inertial velocities in (1a) are related through the kinematic relationship

$$\boldsymbol{J}_{\boldsymbol{\xi}} = \begin{bmatrix} \boldsymbol{R}_B^I(\boldsymbol{q}) & \boldsymbol{0}_{3\times3} & \boldsymbol{0}_{3\times n} \\ \boldsymbol{0}_{4\times3} & \boldsymbol{T}_{\boldsymbol{q}}(\boldsymbol{q}) & \boldsymbol{0}_{4\times n} \\ \boldsymbol{0}_{n\times3} & \boldsymbol{0}_{n\times3} & \boldsymbol{I}_n \end{bmatrix}, \; \boldsymbol{T}_{\boldsymbol{q}}(\boldsymbol{q}) = \frac{1}{2}\begin{bmatrix} -\boldsymbol{\epsilon}^T \\ \eta\boldsymbol{I}_3 + S(\boldsymbol{\epsilon}) \end{bmatrix}, \tag{2}$$

wherein $\boldsymbol{R}_B^I \in \mathrm{SO}(3)$ describes the rotation from the body-fixed frame to the inertial frame and $S : \mathbb{R}^3 \mapsto \mathfrak{so}(3)$ is the skew-symmetry map. The dynamics of (1b) are described by the positive definite inertia matrix $\boldsymbol{M}(\boldsymbol{\theta})$ which includes hydrodynamic added mass, the Coriolis-centripetal matrix $\boldsymbol{C}(\boldsymbol{\theta},\boldsymbol{\zeta})$, which also includes added mass effects, the damping matrix $\boldsymbol{D}(\boldsymbol{\theta},\boldsymbol{\zeta})$, the restoring forces $\boldsymbol{g}(\boldsymbol{\xi})$, and the control forces and moments $\boldsymbol{\tau}$. These equations also hold for terrestrial and space VMSs if the hydrodynamic effects are excluded.

Generally, the dynamics of a UVMS are very hard to model. This is mainly due to the effects of added inertia and hydrodynamic drag forces that are experienced underwater. These directly affect $\boldsymbol{M}$, $\boldsymbol{C}$, and $\boldsymbol{D}$. Additionally, its hydrostatics, collected in $\boldsymbol{g}$, may not be perfectly known, particularly if the vehicle is often reconfigured. As a result, model-based control of UVMSs loses some of its tractability when only poor estimates of the dynamics are available. Even so, model-based controllers may necessarily perform better than those that do not depend on the dynamic model. Indeed, this is often the case, especially for systems with a large degree of coupling at the dynamic level. Additionally, some effects due to model uncertainties may be alleviated by use of robust feedback control, and this will be investigated in this paper.

3. OPERATIONAL SPACE CONTROL

In this section, some background theory of OSC is recounted. Then, motivated by Di Lillo et al. (2021), we derive the task-level dynamics of a UVMS subjected to dynamic model errors in the task-priority OSC. An analysis of the dynamics is then performed, and this reveals that errors in the system inertia estimate may cause an algebraic loop to appear.

3.1 Background

Generally, OSC is a control approach where a system is controlled in the coordinates of some task, such as the end-effector coordinates of a robot manipulator. In task-priority OSC, we can define r tasks in a prioritized order. Tasks can be defined as any function of the system states, and several prioritized tasks form a hierarchy of tasks. The main idea behind the hierarchy is that tasks with higher priority will not be affected by motions generated by lower priority tasks. Let $\boldsymbol{x}_i(t) \in \mathbb{R}^{m_i}$ denote an m_i-dimensional task, with $i \in \{1, \ldots, r\}$. The task coordinates are related to the system configuration through the kinematic relationship

$$\boldsymbol{x}_i(t) = \boldsymbol{f}_i(\boldsymbol{\xi}(t)), \tag{3}$$

where $\boldsymbol{f}_i$ generally is a nonlinear function. The differential task kinematics are found through time differentiation of (3) and insertion of (1a) as

$$\dot{\boldsymbol{x}}_i(t) = \frac{\partial \boldsymbol{f}_i(\boldsymbol{\xi}(t))}{\partial \boldsymbol{\xi}}\dot{\boldsymbol{\xi}} = \bar{\boldsymbol{J}}_i(\boldsymbol{\xi})\dot{\boldsymbol{\xi}}(t) = \boldsymbol{J}_i(\boldsymbol{\xi}(t))\boldsymbol{\zeta}(t), \tag{4}$$

where $\boldsymbol{J}_i(\boldsymbol{\xi}) \triangleq \bar{\boldsymbol{J}}_i(\boldsymbol{\xi})\boldsymbol{J}_{\boldsymbol{\xi}}(\boldsymbol{q}) \in \mathbb{R}^{m_i \times (6+n)}$. Furthermore, omitting dependencies for readability, the task acceleration is given by

$$\begin{aligned}\ddot{\boldsymbol{x}}_i &= \boldsymbol{J}_i\dot{\boldsymbol{\zeta}} + \dot{\boldsymbol{J}}_i\boldsymbol{\zeta} \\ &= \boldsymbol{J}_i\boldsymbol{M}^{-1}(\boldsymbol{\tau} - \boldsymbol{C}\boldsymbol{\zeta} - \boldsymbol{D}\boldsymbol{\zeta} - \boldsymbol{g}) + \dot{\boldsymbol{J}}_i\boldsymbol{\zeta}.\end{aligned} \tag{5}$$

Then, by collecting dynamic terms as

$$\boldsymbol{n} = \boldsymbol{C}\boldsymbol{\zeta} + \boldsymbol{D}\boldsymbol{\zeta} + \boldsymbol{g}, \tag{6}$$

and defining the task inertia (assuming $\boldsymbol{J}_i$ has full rank)

$$\boldsymbol{\Lambda}_i = (\boldsymbol{J}_i\boldsymbol{M}^{-1}\boldsymbol{J}_i^T)^{-1} \in \mathbb{R}^{m_i \times m_i}, \tag{7}$$

we may express the dynamics of the task by

$$\boldsymbol{\Lambda}_i\ddot{\boldsymbol{x}}_i = \boldsymbol{\Lambda}_i\boldsymbol{J}_i\boldsymbol{M}^{-1}(\boldsymbol{\tau} - \boldsymbol{n}) + \boldsymbol{\Lambda}_i\dot{\boldsymbol{J}}_i\boldsymbol{\zeta}. \tag{8}$$

Systems that are redundant with respect to some task $\boldsymbol{x}_i$ are generally able to complete other tasks by projecting control forces and moments into the null space of that task. The dynamically consistent null space projectors associated with each task is given by (Khatib et al., 2004)

$$\boldsymbol{N}_1 = \boldsymbol{I}_{n+6}, \quad \boldsymbol{N}_{i+1} = \left(\boldsymbol{I}_{n+6} - \boldsymbol{N}_i\boldsymbol{J}_i^T(\boldsymbol{J}_i^{\dagger})^T\right)\boldsymbol{N}_i, \tag{9}$$

where

$$\boldsymbol{J}_i^{\dagger} = \boldsymbol{M}^{-1}\boldsymbol{J}_i^T\boldsymbol{\Lambda}_i \tag{10}$$

is the inertia-weighted pseudoinverse of $\boldsymbol{J}_i$. Using the null space projectors, it is possible to define the prioritized task inertia (Khatib et al., 2004) as

$$\boldsymbol{\Lambda}_i = (\boldsymbol{J}_i\boldsymbol{N}_i^T\boldsymbol{M}^{-1}\boldsymbol{N}_i\boldsymbol{J}_i^T)^{-1} = (\boldsymbol{J}_i\boldsymbol{M}^{-1}\boldsymbol{N}_i\boldsymbol{J}_i^T)^{-1}, \tag{11}$$

given that $\boldsymbol{N}_i\boldsymbol{J}_i^T$ has full rank (i.e. that the task is kinematically compatible with higher priority tasks). If the converse is true, $(\boldsymbol{J}_i\boldsymbol{M}^{-1}\boldsymbol{N}_i\boldsymbol{J}_i^T)$ will be singular. Eigenvalue decomposition of $(\boldsymbol{J}_i\boldsymbol{M}^{-1}\boldsymbol{N}_i\boldsymbol{J}_i^T)$ reveals the null space controllable dimensions of the task (Khatib et al., 2004).

Remark 1. Henceforth, m_i is considered to be equal to the number of controllable dimensions in the task.

For a system with r tasks, the control input is chosen as

$$\boldsymbol{\tau} = \boldsymbol{N}_1\boldsymbol{\tau}_1 + \boldsymbol{N}_2\boldsymbol{\tau}_2 + ... + \boldsymbol{N}_r\boldsymbol{\tau}_r + \boldsymbol{n} = \sum_{i=1}^{r}\boldsymbol{N}_i\boldsymbol{\tau}_i + \boldsymbol{n}, \tag{12}$$

where $\boldsymbol{\tau}_1, \boldsymbol{\tau}_2, ..., \boldsymbol{\tau}_r$ are control inputs corresponding to the r different tasks while the nonlinear dynamics (6) are cancelled directly. The task-level control inputs are chosen such that they linearize the task dynamics, i.e.

$$\boldsymbol{\tau}_i = \boldsymbol{J}_i^T\boldsymbol{\Lambda}_i\Big(\boldsymbol{a}_i - \dot{\boldsymbol{J}}_i\boldsymbol{\zeta} - \boldsymbol{J}_i\boldsymbol{M}^{-1}\sum_{j=1}^{i-1}\boldsymbol{N}_j\boldsymbol{\tau}_j\Big), \tag{13}$$

where $\boldsymbol{a}_i$ is the desired task acceleration to be defined later.

Remark 2. The sum in (13) is by definition zero for $i = 1$.

The dynamically consistent null space projectors satisfy

$$\boldsymbol{J}_i\boldsymbol{M}^{-1}\boldsymbol{N}_j = \boldsymbol{0}, \ i < j, \tag{14}$$

that is, they remove any effects from $\boldsymbol{\tau}_i$ on the higher prioritized tasks. Null space projectors constructed without using the inertia weighted Jacobian inverse in (10) will still be kinematically consistent. That is, they satisfy

$$\boldsymbol{J}_i\boldsymbol{N}_j = \boldsymbol{0}, \ i < j, \tag{15}$$

but may not be dynamically consistent and thus a coupling between tasks will remain at the dynamic level. The closed-loop task-level dynamics are found by insertion of (12) and (13) into (8) such that

$$\begin{aligned}\ddot{\boldsymbol{x}}_i &= \boldsymbol{J}_i\boldsymbol{M}^{-1}\boldsymbol{N}_i\boldsymbol{J}_i^T\boldsymbol{\Lambda}_i\Big(\boldsymbol{a}_i - \dot{\boldsymbol{J}}_i\boldsymbol{\zeta} - \boldsymbol{J}_i\boldsymbol{M}^{-1}\sum_{j=1}^{i-1}\boldsymbol{N}_j\boldsymbol{\tau}_j\Big) \\ &\quad + \boldsymbol{J}_i\boldsymbol{M}^{-1}\sum_{j=1, j\neq i}^{r}\boldsymbol{N}_j\boldsymbol{\tau}_j + \dot{\boldsymbol{J}}_i\boldsymbol{\zeta} \\ &= \boldsymbol{a}_i\end{aligned} \tag{16}$$

where (14) and $\boldsymbol{J}_i\boldsymbol{M}^{-1}\boldsymbol{N}_i\boldsymbol{J}_i^T\boldsymbol{\Lambda}_i = \boldsymbol{I}$ have been applied. It can be observed that the closed-loop dynamics are linear as long as the system model is perfectly known.

3.2 Effects of modelling errors

So far it has been assumed that terms in the system dynamics can be perfectly cancelled by the control law. A mathematical model of a real system, however, will inevitably contain parameter uncertainties (Nakanishi et al., 2008). Analysis of the effects of modelling errors has already been done by Antonelli et al. (2018); Di Lillo et al. (2021). With this section, we intend to extend their analysis. We derive the disturbance term in an explicit form and examine effects of modelling errors both up and down the task hierarchy. In the presence of model errors, it is reasonable to rewrite (13) to task-level control inputs in the form

$$\boldsymbol{\tau}_i = \boldsymbol{J}_i^T\hat{\boldsymbol{\Lambda}}_i\Big(\boldsymbol{a}_i - \dot{\boldsymbol{J}}_i\boldsymbol{\zeta} - \boldsymbol{J}_i\hat{\boldsymbol{M}}^{-1}\sum_{j=1}^{i-1}\hat{\boldsymbol{N}}_j\boldsymbol{\tau}_j\Big), \tag{17}$$

where the symbol $\hat{}$ implies an estimate. Analogously to Di Lillo et al. (2021), we will consider parameter uncertainties in the dynamics, while the system kinematics (such as $\boldsymbol{J}_i$) are considered known. The complete control law then takes the form of

$$\begin{aligned}\boldsymbol{\tau} &= \sum_{i=1}^{r}\hat{\boldsymbol{N}}_i\boldsymbol{\tau}_i + \hat{\boldsymbol{n}} \\ &= \sum_{i=1}^{r}\hat{\boldsymbol{N}}_i\boldsymbol{J}_i^T\hat{\boldsymbol{\Lambda}}_i\Big(\boldsymbol{a}_i - \dot{\boldsymbol{J}}_i\boldsymbol{\zeta} - \boldsymbol{J}_i\hat{\boldsymbol{M}}^{-1}\sum_{j=1}^{i-1}\hat{\boldsymbol{N}}_j\boldsymbol{\tau}_j\Big) + \hat{\boldsymbol{n}}.\end{aligned} \tag{18}$$

Inserting (18) into (8) gives

$$\ddot{\boldsymbol{x}}_i = \boldsymbol{J}_i\boldsymbol{M}^{-1}\Big(\sum_{j=1}^{r}\hat{\boldsymbol{N}}_j\boldsymbol{\tau}_j + \hat{\boldsymbol{n}} - \boldsymbol{n}\Big) + \dot{\boldsymbol{J}}_i\boldsymbol{\zeta}. \tag{19}$$

We proceed by splitting the control input in three different parts, namely when $j < i$, $j = i$, and $j > i$. This corresponds to control contributions from higher priority tasks, the current task, and lower priority tasks, respectively. The dynamics of the current task then becomes

$$\begin{aligned}\ddot{\boldsymbol{x}}_i &= \boldsymbol{J}_i\boldsymbol{M}^{-1}\sum_{j=1}^{i-1}\hat{\boldsymbol{N}}_j\boldsymbol{\tau}_j \\ &\quad + \boldsymbol{J}_i\boldsymbol{M}^{-1}\hat{\boldsymbol{N}}_i\boldsymbol{J}_i^T\hat{\boldsymbol{\Lambda}}_i\Big(\boldsymbol{a}_i - \dot{\boldsymbol{J}}_i\boldsymbol{\zeta} - \boldsymbol{J}_i\hat{\boldsymbol{M}}^{-1}\sum_{j=1}^{i-1}\hat{\boldsymbol{N}}_j\boldsymbol{\tau}_j\Big) \\ &\quad + \boldsymbol{J}_i\boldsymbol{M}^{-1}\sum_{j=i+1}^{r}\hat{\boldsymbol{N}}_j\boldsymbol{\tau}_j + \boldsymbol{J}_i\boldsymbol{M}^{-1}(\hat{\boldsymbol{n}} - \boldsymbol{n}) + \dot{\boldsymbol{J}}_i\boldsymbol{\zeta}.\end{aligned} \tag{20}$$

We have that

$$\boldsymbol{J}_i\boldsymbol{M}^{-1}\hat{\boldsymbol{N}}_i\boldsymbol{J}_i^T\hat{\boldsymbol{\Lambda}}_i \triangleq (\boldsymbol{\Lambda}_{i|\widehat{i-1}})^{-1}\hat{\boldsymbol{\Lambda}}_i \triangleq \tilde{\boldsymbol{\Lambda}}_i, \tag{21}$$

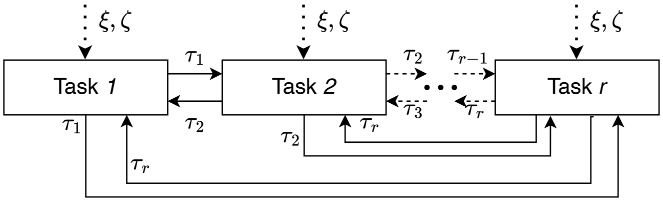

Fig. 2. Conceptual overview of connections between tasks

where $(\boldsymbol{\Lambda}_{i|\widehat{i-1}})^{-1}$ is an inverse task inertia based on the true system inertia. It is projected into an estimate of the dynamically consistent null space by $\hat{\boldsymbol{N}}_i$. By collecting terms and using definition (21) we then obtain

$$\begin{aligned}\ddot{\boldsymbol{x}}_i &= \tilde{\boldsymbol{\Lambda}}_i \boldsymbol{a}_i + \left(\boldsymbol{I} - \tilde{\boldsymbol{\Lambda}}_i\right) \dot{\boldsymbol{J}}_i \boldsymbol{\zeta} + \boldsymbol{J}_i \boldsymbol{M}^{-1} \tilde{\boldsymbol{n}} \\ &\quad + \left(\boldsymbol{J}_i \boldsymbol{M}^{-1} - \tilde{\boldsymbol{\Lambda}}_i \boldsymbol{J}_i \hat{\boldsymbol{M}}^{-1}\right) \sum_{j=1}^{i-1} \hat{\boldsymbol{N}}_j \boldsymbol{\tau}_j \\ &\quad + \boldsymbol{J}_i \boldsymbol{M}^{-1} \sum_{j=i+1}^{r} \hat{\boldsymbol{N}}_j \boldsymbol{\tau}_j \\ &\triangleq \tilde{\boldsymbol{\Lambda}}_i \boldsymbol{a}_i + \boldsymbol{d}_i(t, \boldsymbol{\xi}, \boldsymbol{\zeta}, \boldsymbol{\tau}),\end{aligned} \tag{22}$$

which describes the closed-loop task-level dynamics in the presence of modelling errors. Firstly, (22) has an uncertain input matrix $\tilde{\boldsymbol{\Lambda}}_i$. Several disturbances, collected in $\boldsymbol{d}_i$, affect the dynamics of each task. These include imperfect cancellation of the joint-space dynamics $\boldsymbol{n}$ defined in (6), the Coriolis term $\dot{\boldsymbol{J}}_i \boldsymbol{\zeta}$, and both *top-down* and *bottom-up* disturbances. The last two are caused by coupled task dynamics. Task controls from different tasks may thus propagate both up and down the task levels. The partially compensated top-down disturbances $\left(\boldsymbol{J}_i \boldsymbol{M}^{-1} - \tilde{\boldsymbol{\Lambda}}_i \boldsymbol{J}_i \hat{\boldsymbol{M}}^{-1}\right) \sum_{j=1}^{i-1} \hat{\boldsymbol{N}}_j \boldsymbol{\tau}_j$ appear since they can not be compensated exactly by (17). The bottom-up disturbances $\boldsymbol{J}_i \boldsymbol{M}^{-1} \sum_{j=i+1}^{r} \hat{\boldsymbol{N}}_j \boldsymbol{\tau}_j$ are generated by lower-prioritized tasks on tasks with higher priorities due to the fact that, when the correct inertia matrix is not used, the null space projectors are no longer dynamically consistent. That is, (14) does not hold, and the tasks affect each other at the dynamic level. Thus, all task-level control inputs may affect all tasks as illustrated in Fig. 2. This *coupling in control inputs*, may lead to an algebraic loop, where the task-level control input becomes part of the task-level disturbance term $\boldsymbol{d}_i$. The consequences of this will be further analyzed in Section 4.2.

3.3 Task-level control

A typical choice for the desired task acceleration is

$$\boldsymbol{a}_i = \ddot{\boldsymbol{x}}_i^d - \boldsymbol{K}_{D,i} \dot{\tilde{\boldsymbol{x}}}_i - \boldsymbol{K}_{P,i} \tilde{\boldsymbol{x}}_i \tag{23}$$

which is a proportional-derivative (PD) controller with a task acceleration feedforward term $\ddot{\boldsymbol{x}}_i^d$. The task error is defined as $\tilde{\boldsymbol{x}}_i \triangleq \boldsymbol{x}_i - \boldsymbol{x}_i^d$, where $\boldsymbol{x}_i^d$ is the desired task trajectory. The gains $\boldsymbol{K}_{P,i}$ and $\boldsymbol{K}_{D,i}$ are positive definite matrices. Without any modelling errors, the dynamics of the task tracking error are linear with an exponentially stable (ES) equilibrium. The presence of modelling errors, however, complicates the task dynamics significantly as seen in (22).

In order to compensate for this, we consider a control law with stronger robustness properties than (17), (23). SMC is known to be able to handle uncertain input coefficients and time- and state-dependent perturbations. Traditional first-order sliding mode requires knowledge of the upper bound of the disturbances and is highly prone to chattering. For these reasons, we will instead use the generalized STA (GSTA), which can theoretically compensate for unbounded disturbances and has a continuous control input.

In the scalar case, the GSTA control approach is described by the differential equation (Moreno, 2009)

$$u_{\text{GSTA}} = -\alpha \phi_1(\sigma) + v, \quad \dot{v} = -\beta \phi_2(\sigma), \tag{24}$$

where

$$\phi_1(\sigma) = \lceil \sigma \rfloor^{\frac{1}{2}} + L\sigma, \tag{25a}$$

$$\phi_2(\sigma) = \frac{1}{2} \lceil \sigma \rfloor^{0} + \frac{3}{2} L \lceil \sigma \rfloor^{\frac{1}{2}} + L^2 \sigma, \tag{25b}$$

and $\lceil a \rfloor^b \triangleq |a|^b \operatorname{sgn}(a)$. The control law has three gains $\alpha, \beta, L \in \mathbb{R}$, and uses a *sliding variable* $\sigma \in \mathbb{R}$, to be designed. The applied control signal is continuous due to the integration of the sign term in (24) ($u_{\text{GSTA}} \in \mathcal{C}^0$). We choose the task-level sliding variable as

$$\boldsymbol{\sigma}_i(\boldsymbol{x}) = \dot{\tilde{\boldsymbol{x}}}_i + \boldsymbol{\Omega}_i \tilde{\boldsymbol{x}}_i, \tag{26}$$

with $\boldsymbol{\Omega}_i \succ \boldsymbol{0}$. This choice is made so that on the surface $\boldsymbol{S} \triangleq \{\boldsymbol{x} : \boldsymbol{\sigma}_i(\boldsymbol{x}) = \boldsymbol{0}\}$ in the state space, the task dynamics become

$$\dot{\tilde{\boldsymbol{x}}}_i = -\boldsymbol{\Omega}_i \tilde{\boldsymbol{x}}_i \tag{27}$$

and converge exponentially to the equilibrium $\tilde{\boldsymbol{x}}_i = \boldsymbol{0}$. The scalar control law (24) can be applied independently on each degree of freedom (DoF) of a task by defining

$$\boldsymbol{a}_i = -\boldsymbol{A}_i \boldsymbol{\phi}_1(\boldsymbol{\sigma}_i) + \boldsymbol{v}_i + \ddot{\boldsymbol{x}}_i^r \quad \dot{\boldsymbol{v}}_i = -\boldsymbol{B}_i \boldsymbol{\phi}_2(\boldsymbol{\sigma}_i), \tag{28}$$

where $\boldsymbol{A}_i$ and $\boldsymbol{B}_i$ are diagonal gain matrices, $\ddot{\boldsymbol{x}}^r = \ddot{\boldsymbol{x}}_i^d - \boldsymbol{\Omega}_i \dot{\tilde{\boldsymbol{x}}}_i$, and $\boldsymbol{\phi}_1(\boldsymbol{\sigma}_i)$ and $\boldsymbol{\phi}_2(\boldsymbol{\sigma}_i)$ are the functions $\phi_1(\cdot)$ and $\phi_2(\cdot)$ applied element-wise on $\boldsymbol{\sigma}_i$, respectively.

4. STABILITY ANALYSIS

In this section, we analyze the task dynamics with the PD control law (23) and the STA (28), and discuss the properties of the resulting closed-loop system. Some fundamental challenges of controlling the system in the presence of modelling errors are discussed.

Assumption 3. A set of r tasks are considered, such that the total number of controllable task dimensions is equal to the number of DoFs of the system. Additionally, it is assumed that the trajectories of the system stay away from any singularities.

4.1 Without modelling errors

Without any model errors, the task dynamics are described by (16). This means that all tasks are completely decoupled, implying that any control stabilizing each task also stabilizes the whole system. Furthermore, the DoFs within each task are decoupled, so that each dimension can be controlled independently. The control problem thus simplifies to stabilizing scalar, linear systems of the form

$$\ddot{x}_{ij} = a_{ij}, \tag{29}$$

where $j \in \{1, \ldots, m_i\}$ denotes a dimension in task $\boldsymbol{x}_i$. As mentioned, using the PD control law (23) gives a linear closed-loop system with an ES origin. Similarly, it is well known that applying the STA control law (24) to (29) makes σ_{ij} go to zero in finite time, after which the origin $x_{ij} = 0$ is ES (Moreno, 2009).

Remark 4. As the task dynamics (16) depend on the total system configuration, global results cannot be obtained, due to potential kinematic singularities.

4.2 With modelling errors

Applying the PD control law (23) to the system with modeling errors (22) yields the closed-loop task dynamics

$$\ddot{\tilde{\boldsymbol{x}}}_i + \tilde{\boldsymbol{\Lambda}}_i(\boldsymbol{K}_{D,i}\dot{\tilde{\boldsymbol{x}}}_i + \boldsymbol{K}_{P.i}\tilde{\boldsymbol{x}}_i) = \boldsymbol{d}_i + (\tilde{\boldsymbol{\Lambda}}_i - \boldsymbol{I})\ddot{\boldsymbol{x}}_i^d. \tag{30}$$

When the system dynamics are imperfectly cancelled, the dimensions of each task are no longer decoupled, as discussed in Section 3.2. It has been demonstrated previously (Borlaug et al., 2022) that the GSTA (28) performs well in spite of this. Differentiating (26) with respect to time and inserting (22) with control law (28) yields the dynamics of the sliding variable as

$$\begin{aligned}\dot{\boldsymbol{\sigma}}_i &= \ddot{\tilde{\boldsymbol{x}}}_i + \boldsymbol{\Omega}\dot{\tilde{x}} \\ &= \tilde{\boldsymbol{\Lambda}}_i\big(- \boldsymbol{A}_i\boldsymbol{\phi}_1(\boldsymbol{\sigma}_i) + \boldsymbol{v}_i + \ddot{\boldsymbol{x}}_i^r\big) + \boldsymbol{d}_i - \ddot{\boldsymbol{x}}_i^r \\ &= \tilde{\boldsymbol{\Lambda}}_i\big(- \boldsymbol{A}_i\boldsymbol{\phi}_1(\boldsymbol{\sigma}_i) + \boldsymbol{v}_i\big) + (\tilde{\boldsymbol{\Lambda}}_i - \boldsymbol{I})\ddot{\boldsymbol{x}}_i^r + \boldsymbol{d}_i. \end{aligned} \tag{31}$$

This can be seen as m_i coupled scalar systems, given by

$$\dot{\sigma}_{ij} = \tilde{\lambda}_{ij}u_{\mathrm{GSTA},ij} + \ddot{x}_{ij}^r(\tilde{\lambda}_{ij} - 1) + d_{ij} + c_{ij}, \tag{32}$$

where $\tilde{\lambda}_{ij}$ is the j-th diagonal element of $\tilde{\boldsymbol{\Lambda}}_i$, d_{ij} the j-th element of $\boldsymbol{d}_i$, a_{ij} the j-th element of $\boldsymbol{a}_i$, and c_{ij} contains all coupling terms from the remaining task dimensions. That is, c_{ij} will contain off-diagonal elements in $\tilde{\boldsymbol{\Lambda}}_i$, $\boldsymbol{d}_i$, and so on.

An obvious issue arises if the inertia matrix $\tilde{\boldsymbol{\Lambda}}_i$ is completely unknown. In general, assumptions must be made about the properties of this matrix in order to apply our proposed control laws. An additional problem caused by incorrect knowledge of $\boldsymbol{M}$, is the coupling between control inputs across tasks, cf. Fig. 2. To examine the effects of this coupling, we consider what happens at any point where $\ddot{\tilde{\boldsymbol{x}}}_i = \boldsymbol{0}$. At such a point, the task-level control would have to match the disturbance $\boldsymbol{d}_i$ in order to keep perfect tracking, that is

$$\boldsymbol{0} = \ddot{\boldsymbol{x}}_i - \ddot{\boldsymbol{x}}_i^d \tag{33}$$

$$= \tilde{\boldsymbol{\Lambda}}_i\boldsymbol{a}_i + \boldsymbol{d}_i - \ddot{\boldsymbol{x}}_i^d \tag{34}$$

$$\implies \boldsymbol{a}_i = \tilde{\boldsymbol{\Lambda}}_i^{-1}\ddot{\boldsymbol{x}}_i^d - \tilde{\boldsymbol{\Lambda}}_i^{-1}\boldsymbol{d}_i \tag{35}$$

which can be written more explicitly as

$$\begin{aligned}\boldsymbol{a}_i = {} & \tilde{\boldsymbol{\Lambda}}_i^{-1}\ddot{\boldsymbol{x}}_i^d - (\tilde{\boldsymbol{\Lambda}}_i^{-1} - \boldsymbol{I})\dot{\boldsymbol{J}}_i\boldsymbol{\zeta} + \tilde{\boldsymbol{\Lambda}}_i^{-1}\boldsymbol{J}_i\boldsymbol{M}^{-1}\tilde{\boldsymbol{n}} \\ & + \Big(\tilde{\boldsymbol{\Lambda}}_i^{-1}\boldsymbol{J}_i\boldsymbol{M}^{-1} - \boldsymbol{J}_i\hat{\boldsymbol{M}}^{-1}\Big)\sum_{j=1}^{i-1}\hat{\boldsymbol{N}}_j\boldsymbol{\tau}_j \\ & + \tilde{\boldsymbol{\Lambda}}_i^{-1}\boldsymbol{J}_i\hat{\boldsymbol{M}}^{-1}\sum_{j=i+1}^{r}\hat{\boldsymbol{N}}_j\boldsymbol{\tau}_j, \end{aligned} \tag{36}$$

where all $\boldsymbol{\tau}_j$ are given by (13). Thus, the task-level control has to cancel both top-down and bottom-up disturbances caused by all other control inputs $\boldsymbol{\tau}_j$. Since the same is true for all other task levels, an algebraic loop appears. This holds regardless of which task-level control law is used. The effects of any higher- or lower-priority control $\boldsymbol{\tau}_j$ on $\boldsymbol{x}_i$ are scaled by $\Big(\boldsymbol{J}_i\boldsymbol{M}^{-1} - \tilde{\boldsymbol{\Lambda}}_i\boldsymbol{J}_i\hat{\boldsymbol{M}}^{-1}\Big)$ or $\boldsymbol{J}_i\tilde{\boldsymbol{M}}^{-1}\hat{\boldsymbol{N}}_j$, respectively, both of which vanishes with perfect knowledge of $\boldsymbol{M}$. Depending on the accuracy of $\hat{\boldsymbol{M}}$, this effect may cause the control input to grow unbounded and render any task-level control law useless. The effects caused by incorrect cancellation of the nonlinear effects $\boldsymbol{n}$ are easier to handle, since it is not directly dependent on the other task-level controls.

Since the terms in the disturbance $\boldsymbol{d}_i$ are generally nonvanishing, a PD controller will not be able to perfectly compensate for them. Therefore, the best possible performance using a PD control law would be to bring the tracking error $\tilde{\boldsymbol{x}}_i$ within some bound around $\boldsymbol{0}$, with the size of the bound depending on $\boldsymbol{d}_i$. The GSTA control law (28) has stronger robustness properties which should allow it to compensate for the disturbances to a higher degree. Application of GSTA, meanwhile, means that the disturbance term $\boldsymbol{d}_i$ becomes non-Lipschitz and even harder to compensate for.

The challenges detailed in this section prevent us from reaching any definite conclusions on the stability properties of the system (22) with the control law (23) or (28) without making inelegant assumptions. However, as will be demonstrated in Section 5 (and has been shown in a variety of other works), practical implementation of OSC tends to perform fairly well.

5. SIMULATIONS

In this section, we investigate the performance of the control law (12), (13) together with the PD law (23) and the GSTA law (28), respectively, through a simulation study. The UVMS used in the simulations is the AIAUV "Eelume" (Liljebäck and Mills, 2017; Schmidt-Didlaukies et al., 2018) depicted in Fig. 1. The simulations were run in Matlab with ODE1 and a fixed step size of 0.01s.

Task-priority control of AIAUVs has been investigated across several works. In Borlaug et al. (2022) and Iversflaten et al. (2022), inverse kinematics control was applied, and the dynamics were controlled by sliding mode control (SMC). In Basso and Pettersen (2020), feedback linearization in a task-priority OSC framework was investigated for and applied to an AIAUV but relied on perfect dynamic model knowledge. Model knowledge was utilized also in Sæbø et al. (2022), where the passivity-based method in Dietrich and Ott (2020) was applied to an AIAUV and first-order SMC was adopted at each task level.

Eelume is a modular, slender, and multi-articulated vehicle with no separate base. Due to its many DoFs, redundancy resolution is an important element of AIAUV control. The modularity allows the robot to be easily reconfigured, which affects its weight distribution and balance. Obviously, it is infeasible to derive new model parameters with every configuration change. Correctly modelling this robot for the purpose of simulation is considered "immensely challenging" (Schmidt-Didlaukies et al., 2018). These characteristics make Eelume a prime candidate for the simulation study.

The simulation model of the robot has been derived by perturbing the nominal controller model (1). The inertia matrix (and consequently the Coriolis matrix (Fossen, 2011)) has been modified by changing the added mass coefficients in all DoFs. Both linear and nonlinear damping coefficients have been perturbed as well. The nominal and perturbed parameters are shown in Table 1.

The simulated AIAUV has nine links and $n = 8$ joints, yielding a total of 14 DoFs. Seven thrusters are positioned along the robot. Since Eelume has no separate robot base, both of its ends can be viewed as end-effectors. Therefore, it is logical to attempt to control the configuration of both ends. We choose the “tail” (henceforth denoted “base” for familiarity) configuration as the highest-prioritized task, and “head” (denoted “end-effector”) as the second. The first task and its error can be expressed as

$$\boldsymbol{x}_1 = \begin{bmatrix} \boldsymbol{p}^i_{ib} \\ \boldsymbol{\epsilon} \end{bmatrix}, \tilde{\boldsymbol{x}}_1 = \begin{bmatrix} \boldsymbol{p}^i_{ib} - \boldsymbol{p}^i_{ib,d} \\ \eta\boldsymbol{\epsilon}_d - \eta_d\boldsymbol{\epsilon} + S(\boldsymbol{\epsilon})\boldsymbol{\epsilon}_d \end{bmatrix}, \tag{37}$$

where $\boldsymbol{p}^I_{IB,d}$ and $\boldsymbol{q}_d = \left[\eta_d, \boldsymbol{\epsilon}_d^T\right]^T$ is the desired base position and attitude, respectively. The vector part of the quaternion is sufficient for attitude representation due to its unit length constraint. Analogously, the end-effector configuration is defined as the second task, $\boldsymbol{x}_2$. At most, these two tasks consume $m_1 + m_2 = 6 + 6 = 12$ DoFs of the system. To ensure that no DoFs are left uncontrolled, a final task was chosen, which was to keep the joint velocities at zero. In practice, the eigenvalue decomposition detailed in Section 3.1 is implemented with a small threshold value ϵ_i to avoid inversion of arbitrarily small elements. The controller gains were chosen equal for the two main tasks. For the PD controller, they were chosen as $\boldsymbol{K}_{P,1} = \boldsymbol{K}_{P,2} = \operatorname{diag}\{0.5, 0.5, 0.5, 1, 1, 1\}$ and $\boldsymbol{K}_{D,1} = \boldsymbol{K}_{D,2} = \operatorname{diag}\{5, 5, 5, 3, 3, 3\}$. The STA proved to perform well with unit gains, so no further tuning was performed, i.e. $\boldsymbol{A}_1 = \boldsymbol{A}_2 = \boldsymbol{B}_1 = \boldsymbol{B}_2 = \boldsymbol{I}_6$, and $L = 1$. The final task only has a damping term, with gain $\boldsymbol{K}_{D,3} = 10\boldsymbol{I}_n$. Moreover, the STA control law was implemented with a continuous approximation of the sign term, namely $\operatorname{sgn}(a) \approx \tanh(a/\epsilon)$, where $\epsilon = 0.01$, in order to further attenuate any chattering.

We initialized the system in an S-shape (see Fig. 1) ($\boldsymbol{\theta} = \left[0, \frac{\pi}{3}, 0, \frac{\pi}{3}, 0, -\frac{\pi}{3}, 0, -\frac{\pi}{3}\right]^T$), since it yields compatibility between the base and end-effector tasks. The reference trajectories for the base and end-effector configurations are shown in Fig. 3 and chosen such that the tasks eventually cannot be completed simultaneously, i.e. that $m_2 < 6$. Figs. 4 and 5 show the tracking error of the base and end-effector tasks both with the PD control law (23) and the GSTA (28). It can be seen that the x-position and pitch angle of the end-effector become uncontrollable just before $t = 40s$. This happens since the end-effector tries to move too far away from the base, such that $\boldsymbol{N}_2\boldsymbol{J}_2^T$ drops rank. It is clear that the GSTA exhibits superior tracking to the PD controller in the controllable dimensions of both tasks. Lastly, Fig. 6 shows the control forces and moments generated by both control laws. Notably, the sliding mode control signal in the upper plot oscillates compared to the PD control input. The control signal is however still continuous, and the oscillations are fairly small with a frequency of around 1 Hz and an amplitude less than 1 N. Thus the increased robustness compared to the PD controller may be worth the additional chattering depending on the system.

Table 1. Dynamic model parameters

	Nominal	Perturbed
Added mass coeff.	1	3
Nonlinear drag coeff. in surge	0.2	0.6
Nonlinear drag coeff. in roll	0.1	0.7
Nonlinear crossflow drag coeff.	0.5	0.1
Linear cross drag coeff.	0.1	0.8
Added mass ratio, surge	0.2	0.5
Linear drag parameter, surge	0.1	0.5
Linear drag parameter, roll	0.1	0.7

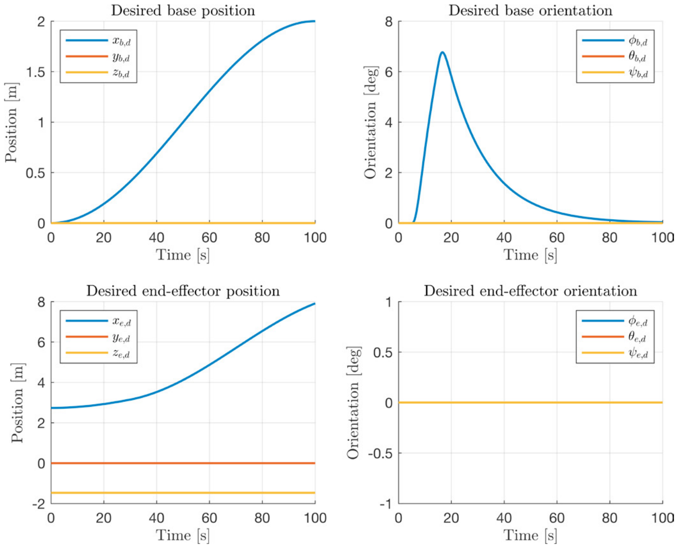

Fig. 3. The desired base and end-effector trajectories

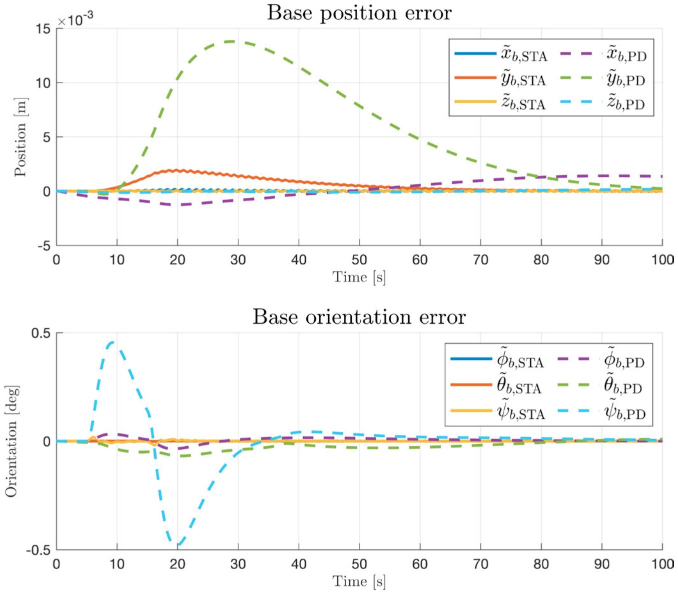

Fig. 4. Comparison of the base configuration errors using PD and STA control

Remark 5. The transition that happens as m_2 first decreases causes a discontinuity in the control just before $t = 40$s but is of no significance in the context of this work.

6. CONCLUSIONS AND FUTURE WORK

In this paper, the task-priority OSC problem was formulated for UVMSs whose dynamic model has parameter

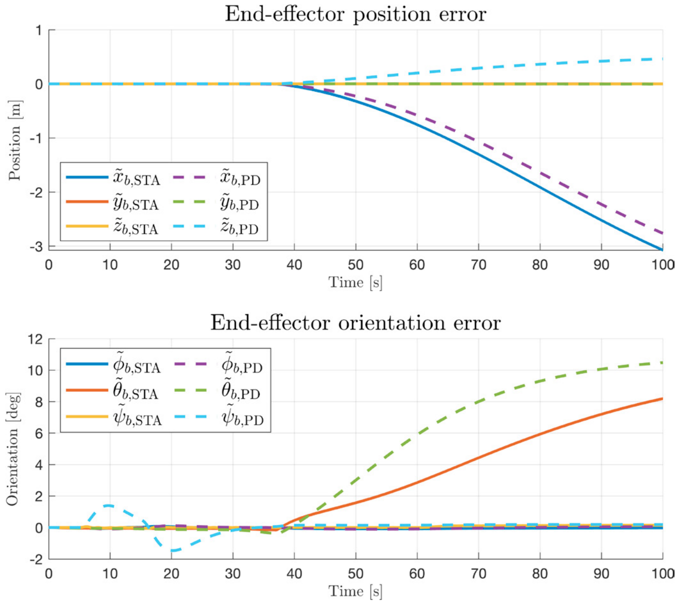

Fig. 5. Comparison of the end-effector configuration errors using PD and STA control

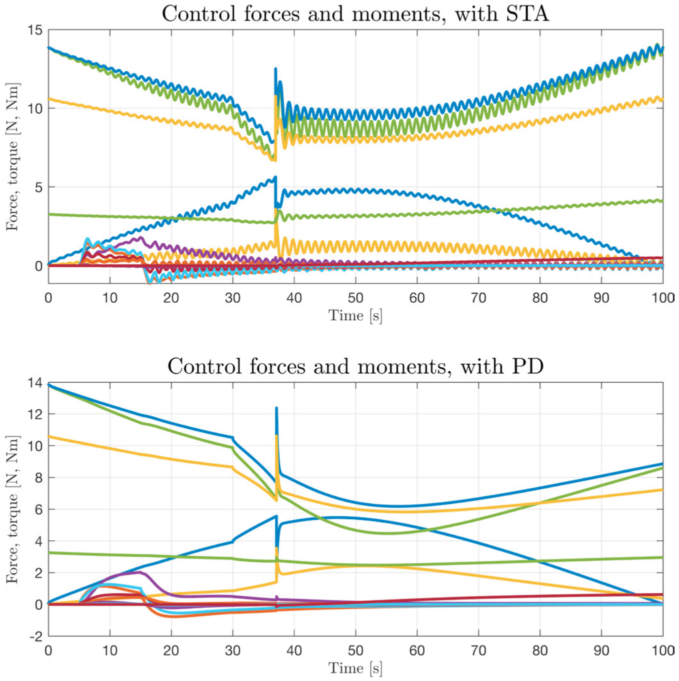

Fig. 6. Control forces and moments in both control laws

uncertainties. The resulting task-level dynamics were analyzed for the case with an arbitrary number of tasks. The analysis revealed that every task level may suffer from an algebraic loop due to the presence of both top-down and bottom-up disturbances. The analysis is applicable also for other VMSs, which can be viewed as special cases of the UVMS model (1). A simulation study using an AIAUV was conducted which validated the aforementioned analysis. We implemented a task-level GSTA in an attempt to achieve satisfactory tracking in the presence of large errors in the dynamic model parameters. This control approach was compared to a standard PD controller and displayed significantly improved robustness. Future work will focus on experimental validation.

REFERENCES

Antonelli, G., Di Lillo, P., and Natale, C. (2018). Modeling errors analysis in inverse dynamics approaches within a task-priority framework. In *Proc. 2018 IEEE Conf. on Control Technology and Applications*. Copenhagen, Denmark.

Basso, E.A. and Pettersen, K.Y. (2020). MIMO feedback linearization of redundant robotic systems using task-priority operational space control. In *Proc. 21st IFAC World Congress*.

Borlaug, I.G., Pettersen, K.Y., and Gravdahl, J.T. (2022). The generalized super-twisting algorithm with adaptive gains. *Int. J. Robust and Nonlinear Control*, 7240–7270.

Di Lillo, P., Antonelli, G., and Natale, C. (2021). Effects of dynamic model errors in task-priority operational space control. *Robotica*, 1642–1653.

Dietrich, A. and Ott, C. (2020). Hierarchical impedance-based tracking control of kinematically redundant robots. *IEEE Trans. on Robotics*, 204–221.

Fossen, T.I. (2011). *Handbook of Marine Craft Hydrodynamics and Motion Control*. John Wiley & Sons, Ltd.

From, P., Gravdahl, J., and Pettersen, K. (2014). *Vehicle-Manipulator Systems*. Springer-Verlag London.

Garofalo, G., Wu, X., and Ott, C. (2021). Adaptive passivity-based multi-task tracking control for robotic manipulators. *IEEE Robotics and Aut. Letters*, 6(4).

Iversflaten, M.H., Haraldsen, A., and Pettersen, K.Y. (2022). Kinematic and dynamic control of cooperating underwater vehicle-manipulator systems. In *Proc. 14th IFAC Conf. on Control Applications in Marine Systems, Robotics and Vehicles*. Kongens Lyngby, Denmark.

Khatib, O. (1987). A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal on Robotics and Automation*, 43–53.

Khatib, O., Sentis, L., Park, J., and Warren, J. (2004). Whole-body dynamic behavior and control of human-like robots. *Int. Journal of Humanoid Robotics*.

Lee, J., Dallali, H., Jin, M., Caldwell, D.G., and Tsagarakis, N.G. (2019). Robust and adaptive dynamic controller for fully-actuated robots in operational space under uncertainties. *Auton. Robots*, 43(4), 1023–1040.

Liljebäck, P. and Mills, R. (2017). Eelume: A flexible and subsea resident IMR vehicle. In *Proc. OCEANS*, 1–4.

Moreno, J.A. (2009). A linear framework for the robust stability analysis of a generalized super-twisting algorithm. In *Proc. 2009 6th Int. Conf. on Electrical Engineering, Computing Science and Automatic Control*. Toluca, Mexico.

Nakanishi, J., Cory, R., Mistry, M., Peters, J., and Schaal, S. (2008). Operational space control: A theoretical and empirical comparison. *Int. J. on Robotics Research*.

Schmidt-Didlaukies, H.M., Sørensen, A.J., and Pettersen, K.Y. (2018). Modeling of articulated underwater robots for simulation and control. In *Proc. 2018 IEEE/OES AUV Workshop*. Porto, Portugal.

Slotine, J.J., Khatib, O., and Ruth, D. (1988). Robust control in operational space for goal-positioned tasks. *Int. Journal of Robotics and Automation*.

Sæbø, B.K., Pettersen, K.Y., and Gravdahl, J.T. (2022). Robust task-priority impedance control for vehicle-manipulator systems. In *Proc. 2022 IEEE Conf. on Control Technology and Applications*. Trieste, Italy.