Framework for Fair Comparisons of Underwater Vehicle Controllers

Showcasing the Robustness Properties of a Model-free Sliding Mode Controller Tuned with a Random-forest-based Bayesian Optimization Approach

Musa Morena Marcusso Manhães[1], Sebastian A. Scherer[1], Luiz Ricardo Douat[1],
Martin Voss[1] and Thomas Rauschenbach[2]

[1] *Corporate Sector for Research and Advance Engineering, Robert Bosch GmbH,*
Robert-Bosch-Campus 1, 71272, Renningen, Baden-Württemberg, Germany

[2] *Advanced System Technology (AST), Fraunhofer-Institute of Optronics, Systems Technologies and Image Exploitation,*
Am Vogelherd 50, 98693, Ilmenau, Thüringen, Germany

Keywords: Underwater Robotics, Robust Control, Parameter Optimization, Sliding Mode Control, SMAC, Underwater Robotics Simulation, Gazebo.

Abstract: Tasks with underwater vehicles present several challenges that include complex and hazardous environments, and unmodeled and/or unknown uncertainties. The setup of positioning controllers is therefore a difficult and laborious task and very often leads to suboptimal performance results on the field. This paper shows a framework for methodical evaluation and setup of dynamic positioning controllers for underwater vehicles with simulation-based optimization method using performance metrics. The proposed method can be configured to be mission specific and delivers a controller configuration that also allows a fair numerical comparison between control algorithms on similar scenarios.

1 INTRODUCTION

From underwater inspection of off-shore wind turbine structures in depths of a few dozen meters to oil and gas interventions in thousands of meters, remotely operated vehicles (ROVs) constitute an ubiquitous enabler of industrial subsea activities.

The specialized literature provides several advanced model-based control strategies for underwater vehicles (Fossen, 2011; Zhu and Gu, 2011; Molero et al., 2011; Khadhraoui et al., 2014; Soylu et al., 2016). Nevertheless, many algorithms still rely on PID-like controllers due to their simple implementation (Hosseini and Seyedtabaii, 2016; Garcia-Valdovinos et al., 2009). PID controllers can be easily tuned even when the vehicle model is unknown, but at the expense of not intrinsically assuring robustness properties against system parameter variations and external disturbances. A promising model-free robust alternative, based on variable structure sliding mode control, was presented in (García-Valdovinos et al., 2014).

In (García-Valdovinos et al., 2014), as well as in numerous other sources in the same field, the proposed controller has its parameters selected following a trial and error procedure, which not only cannot guarantee a good performance in the presence of uncertain perturbations but also makes the numerical comparison with other dynamic positioning controllers, even under equal scenarios, rather difficult (Antonelli, 2014). Furthermore, considering the difficult environment in which underwater vehicles have to be deployed regarding costs and sensor systems, a preliminary simulation-based evaluation can be used to decrease the number of iterations on experiments related to controller configuration by providing a parameter set that has shown good performance metrics in simulated scenarios.

To the best knowledge of the authors, there are today no published results on optimal parameter search applied to dynamic positioning controllers with particular focus on performance analysis, but examples can be found in other fields such as for safe optimization of controller parameters for quadrotors (Berkenkamp et al., 2016) and gait optimization for bipedal locomotion (Calandra et al., 2014). Both (Berkenkamp et al., 2016) and (Calandra et al., 2014) rely on Gaussian processes to learn the system's performance metrics from a set of experimental data and use this fitted model in the search for optimal param-

Manhães, M., Scherer, S., Douat, L., Voss, M. and Rauschenbach, T.
Framework for Fair Comparisons of Underwater Vehicle Controllers - Showcasing the Robustness Properties of a Model-free Sliding Mode Controller Tuned with a Random-forest-based Bayesian Optimization Approach.
DOI: 10.5220/0006427001020113
In *Proceedings of the 7th International Conference on Simulation and Modeling Methodologies, Technologies and Applications (SIMULTECH 2017)*, pages 102-113
ISBN: 978-989-758-265-3

eters for the system's controller and walking gait. As discussed in (Calandra et al., 2014), the strategy has been shown to converge to a near-optimal solution using a much smaller number of system runs in comparison with algorithms that rely, for example, on gradient descent methods.

The objective of this paper is to create a methodical simulation-based procedure to search for the optimal parameter set for a generic controller through minimization of a performance cost function to enable a fair comparison of dynamic positioning controllers in equal scenarios and mission plan. The minimizer used for the parameter search, SMAC (”Sequential Model-based Algorithm Configuration”) (Hutter et al., 2011; Lindauer et al., 2017), fits a random forest model on data extracted from the performance cost function computed after each simulation run to search for best controller parameter set.

The performance cost function is based on performance metrics that are derived from simulated pose and velocity error in a scenario where the vehicle is subjected to different types of disturbances while following a pre-defined trajectory. The objective is to find parameters that will allow the vehicle to complete a mission with focus on the performance for trajectory tracking and disturbance rejection. The controllers to be parametrized and later compared against each other for the use-case presented in this paper are a conventional MIMO PID controller and a model-free sliding mode controller (García-Valdovinos et al., 2014). Both were implemented using ROS and integrated into the UUV Simulator (Manhães et al., 2016a) to control a fully actuated work-class ROV.

The paper is further structured as follows. In Section 2 an overview of the controller optimization is depicted. The controllers used in this use-case are presented in Section 3. The simulated scenario is shown in detail in Section 4. The results for the optimal controller parametrization are presented in 5 and a validation experiment for a different scenario is presented in 6. Section 7 includes conclusion and discussion of the results. The source code and the simulation environment used to generate the results presented in the following sections can be found at (Manhães et al., 2016a).

2 PERFORMANCE-BASED CONTROLLER PARAMETRIZATION

Performance analysis to evaluate controllers is already a subject of interest in fields with hard security constraints, such as flight control (Heise et al., 2013; Stepanyan et al., 2009). The procedure is independent of the system model and controller algorithm, and tries to guarantee that the system will present a safe performance during its tasks. The establishment of a clear method for controller design and evaluation can also benefit the creation of safety and control design standards (Stepanyan et al., 2009).

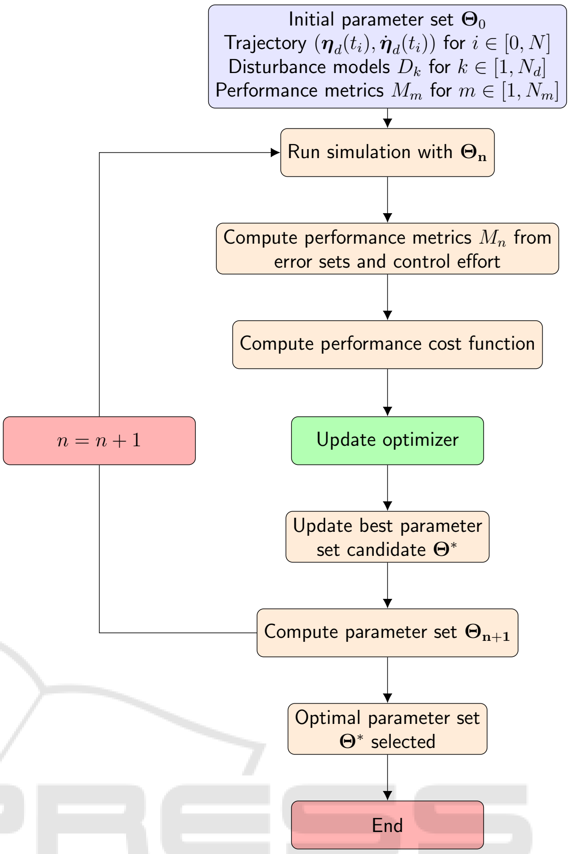
Initial parameter set $\mathbf{\Theta}_0$
Trajectory $(\boldsymbol{\eta}_d(t_i), \dot{\boldsymbol{\eta}}_d(t_i))$ for $i \in [0, N]$
Disturbance models D_k for $k \in [1, N_d]$
Performance metrics M_m for $m \in [1, N_m]$
Run simulation with $\mathbf{\Theta_n}$
Compute performance metrics M_n from error sets and control effort
Compute performance cost function
$n = n + 1$
Update optimizer
Update best parameter set candidate $\mathbf{\Theta}^*$
Compute parameter set $\mathbf{\Theta_{n+1}}$
Optimal parameter set $\mathbf{\Theta}^*$ selected
End

Figure 1: Flow chart to the parameter optimization procedure.

As shown in Figure 1, the procedure is very straightforward. The parameter optimization is executed for a specific scenario using pre-defined disturbances models D_k. The disturbance models are chosen in order to represent a worst case scenario for the mission objective in order to allow the optimal parameter set to provide a good trajectory tracking performance for a large range of unmodeled disturbances.

There is, however, the possibility of controller parameters converging to a local minima without exploring the parameter set or the resulting set will be a result of overfitting the controller parameters to the given simulated scenario. It was therefore decided upon the use of the algorithm configuration tool SMAC (Hutter et al., 2011). This global black-box algorithm configuration method calculates the next parameter sets by fitting a random forest model to the defined cost function output. Due to the use of random forest models, the optimizer is less likely to get stuck in local minima and is capable of exploring unseen regions of the parameter space (Hutter et al., 2011).

2.1 Performance Cost Function

The performance of a controller applied to a scenario is a very broad subject to tackle and highly dependent on the application and/or mission. For this use-case, the performance metrics used are focused on the position and orientation errors. Considering a trajectory $(\eta_d(t_i), \dot{\eta}_d(t_i))$ for $i \in [0,N]$ and t_N being the final time stamp, set as reference for pose (η_d) and velocity ($\dot{\eta}_d$) to the controller, pose and velocity error vectors can be computed after each simulated scenario with a controller parameter set Θ_n.

The cost function takes then into account a weighted sum of root mean square errors for position and orientation as shown in Equation 1:

$$
\begin{aligned}
C_n = \sum_{m=1}^{N_m} w_m M_m & \\
= w_1 \sqrt{\frac{1}{N}\sum_{i=0}^{N} \left\|\eta_{1_d}(t_i) - \eta_1(t_i))\right\|^2} & \\
+ w_2 \sqrt{\frac{1}{N}\sum_{i=0}^{N} (\phi_d(t_i) - \phi(t_i))^2} & \quad (1) \\
+ w_3 \sqrt{\frac{1}{N}\sum_{i=0}^{N} (\theta_d(t_i) - \theta(t_i))^2} & \\
+ w_4 \sqrt{\frac{1}{N}\sum_{i=0}^{N} (\psi_d(t_i) - \psi(t_i))^2} &
\end{aligned}
$$

being w_m set as the inverse of the maximum accepted value for the metrics in case, η_{1_d} the reference position vector $(x_d, y_d, z_d)^T$, and ϕ_d, θ_d and ψ_d the orientation reference in roll, pitch and yaw, respectively.

2.2 Disturbance Models

Two types of disturbance models are considered in this paper. The first basic model is the application of wrench vector defined on the WORLD frame to the vehicle's center of mass for a limited time range $[t_i, t_f]$. The disturbance can be formally described in Equation 2.

$$
\mathbf{F} = \begin{cases} (F_x, F_y, F_z, \tau_x, \tau_y, \tau_z)^T, & \text{for } t_i \leq t \leq t_f \\ \mathbf{0} \in \mathbb{R}^6, & \text{otherwise} \end{cases} \quad (2)
$$

In a similar way, a constant current velocity $\mathbf{V}_c$ also defined in the WORLD frame subject to the horizontal angle θ_c can also be scheduled to be added to the simulation for a given time range in a similar manner following the model in Eq. 3.

$$
\mathbf{V}_c = \begin{cases} (V_c \cos\theta_c, V_c \sin\theta_c, 0)^T, & \text{for } t_i \leq t \leq t_f \\ \mathbf{0} \in \mathbb{R}^3, & \text{otherwise} \end{cases} \quad (3)
$$

3 CONTROLLERS

In this study case, two model-free controllers were subject to parameter optimization: a conventional MIMO PID controller and a model-free sliding mode controller. Both controllers were optimized in the same scenario with the same disturbances to allow a fair comparison later on.

3.1 MIMO PID Controller

Consider ξ and $\dot{\xi}$ to be, respectively, the pose and velocity error vectors in $\mathbb{R}^6$ with respect to the BODY-frame, the PID controller's control force output is described by the following equation:

$$
\tau_{PID} = \mathbf{K}_P \xi + \mathbf{K}_D \dot{\xi} + \mathbf{K}_I \int_0^t \xi(\tau) d\tau \quad (4)
$$

where $\mathbf{K}_P, \mathbf{K}_D, \mathbf{K}_I \in \mathbb{R}^{6\times 6}$ are chosen to be positive semi-definite diagonal matrices.

3.2 Model-free Sliding Mode Controller

The model-free sliding mode controller presented below is based on the work of (García-Valdovinos et al., 2014) but differs in two aspects from the original algorithm, namely that the errors are represented in the BODY-frame coordinates and, according to our convention, errors are represented with the opposite sign.

Assume BODY-frame velocities and WORLD-frame poses to be represented, respectively, as $\nu = [\nu_1, \nu_2]^T = [u, v, w, p, q, r]^T$ and $\eta = [\eta_1, \eta_2]^T = [x, y, z, \phi, \theta, \psi]^T$, where $\nu_1 \in \mathbb{R}^3$ and $\nu_2 \in \mathbb{R}^3$ are the linear and angular velocity vectors with respect to the BODY-frame, and $\eta_1 \in \mathbb{R}^3$ and $\eta_2 \in \mathbb{R}^3$ are position and orientation vectors represented in the WORLD-frame.

The equation of motion expressed in BODY-frame (Fossen, 2011) is given as:

$$
\mathbf{M}\dot{\nu} + \mathbf{C}(\nu)\nu + \mathbf{D}(\nu)\nu + \mathbf{g}(\eta) = \tau \quad (5)
$$

with $\mathbf{M} \in \mathbb{R}^{6\times 6}$, $\mathbf{C} \in \mathbb{R}^{6\times 6}$, $\mathbf{D} \in \mathbb{R}^{6\times 6}$, $\mathbf{g} \in \mathbb{R}^6$, and $\tau \in \mathbb{R}^6$ representing respectively the inertia and added mass matrix, the Coriolis and centripetal terms accounting with added mass terms, the damping matrix, the gravitational forces and the control inputs. The transformation $\dot{\eta} = \mathbf{J}(\eta)\nu$ is used to convert linear and angular velocity from BODY to WORLD frame, $\mathbf{J}(\eta)$ being defined as in (Garcia-Valdovinos et al., 2009).

The left-hand side of Equation 5 can be linearly parametrized, in terms of a nominal reference $(\nu_r, \dot{\nu}_r)$, by the product of a regressor $\mathbf{Y}_r(\nu, \eta, \nu_r, \dot{\nu}_r) \in \mathbb{R}^{6\times p}$ (Lantos and Márton, 2011), composed of known nonlinear terms, and a constant parameter vector $\Theta \in \mathbb{R}^p$:

$$\mathbf{M}\dot{\nu}_r + \mathbf{C}(\nu)\nu_r + \mathbf{D}(\nu)\nu_r + \mathbf{g}(\eta) = \mathbf{Y}_r(\nu, \eta, \nu_r, \dot{\nu}_r)\Theta \tag{6}$$

Subtracting (6) from (5) leads to the representation of Equation 5 in error coordinates (Parra-Vega and Arimoto, 1995; García-Valdovinos et al., 2014):

$$\mathbf{M}\dot{\mathbf{s}}_r + \mathbf{C}(\nu)\mathbf{s}_r + \mathbf{D}(\nu)\mathbf{s}_r = \tau - \mathbf{Y}_r(\nu, \eta, \nu_r, \dot{\nu}_r)\Theta \tag{7}$$

with the extended error $\mathbf{s}_r = \nu - \nu_r$.

Defining a change of coordinates to the nominal reference (Parra-Vega et al., 2003; García-Valdovinos et al., 2014) leads to:

$$\nu_r = \nu_d + \alpha\xi + \mathbf{s}_d - \mathbf{K}_I \int_0^t \mathrm{sign}(\mathbf{s}_\nu)\, d\sigma \tag{8}$$

where

$$\mathbf{s}_d = \mathbf{s}(t_0)e^{-\kappa t} \tag{9}$$

$$\mathbf{s}_\nu = \mathbf{s} - \mathbf{s}_d \tag{10}$$

with α and $\mathbf{K}_I$ being diagonal positive definite 6×6 matrices, κ a positive scalar, $\mathrm{sign}(x)$ the input-wise discontinuous signum function for the vector x, and the sliding surface $\mathbf{s}$ defined as $\mathbf{s} = -\dot{\xi} - \alpha\xi$. $\mathbf{s}_d$ is responsible for a smooth initialization of the controller output and $\boldsymbol{\nu}_d$ is the velocity reference with respect to the BODY frame. The extended error $\mathbf{s}_r$ can be written as:

$$\mathbf{s}_r = \mathbf{s}_\nu + \mathbf{K}_I \int_0^t \mathrm{sign}(\mathbf{s}_\nu)\, d\sigma \tag{11}$$

The control law $\tau_{SM} = -\mathbf{K}_D\mathbf{s}_r$, with $\mathbf{K}_D$ as a diagonal positive 6×6 matrix, in closed-loop with system presented in Equation 7 yields to:

$$\begin{aligned}\mathbf{M}\dot{\mathbf{s}}_r = &- \mathbf{K}_D\mathbf{s}_r - \mathbf{C}(\nu)\mathbf{s}_r - \mathbf{D}(\nu)\mathbf{s}_r \\ &- \mathbf{Y}_r(\nu, \nu_r, \dot{\nu}_r, \eta)\Theta.\end{aligned} \tag{12}$$

To ensure the convergence of the system, let us consider a Lyapunov candidate function defined as follows, as given by (Garcia-Valdovinos et al., 2009):

$$V = \frac{1}{2}\mathbf{s}_r^T\mathbf{M}\mathbf{s}_r \tag{13}$$

Its corresponding time derivative is:

$$\begin{aligned}\dot{V} = &\mathbf{s}_r^T[-\mathbf{K}_D\mathbf{s}_r - \mathbf{C}(\nu)\mathbf{s}_r - \mathbf{D}(\nu)\mathbf{s}_r - \mathbf{Y}_r\Theta] + \\ &\frac{1}{2}\mathbf{s}_r^T\dot{\mathbf{M}}\mathbf{s}_r\end{aligned} \tag{14}$$

Now, using the fact that $\mathbf{x}^T[\dot{\mathbf{M}} - 2\mathbf{C}(\nu)]\mathbf{x} = 0, \forall \mathbf{x} \in \mathbb{R}^6$, $\mathbf{x} \neq \mathbf{0}$, we have:

$$\dot{V} = -\mathbf{s}_r^T[\mathbf{K}_D + \mathbf{D}(\nu)]\mathbf{s}_r - \mathbf{s}_r^T\mathbf{Y}_r\Theta \tag{15}$$

Since $\mathbf{D}(\nu)$ is a positive definite matrix and $\mathbf{Y}_r\Theta$ is upper bounded (Parra-Vega et al., 2003) by $\rho(t)$:

$$\begin{aligned}\dot{V} &= -\mathbf{s}_r^T[\mathbf{K}_D + \mathbf{D}(\nu)]\mathbf{s}_r - \mathbf{s}_r^T\mathbf{Y}_r\Theta \\ &\leq -\mathbf{s}_r^T\mathbf{K}_D\mathbf{s}_r - \mathbf{s}_r^T\mathbf{Y}_r\Theta \\ &\leq -\mathbf{s}_r^T\mathbf{K}_D\mathbf{s}_r + \|\mathbf{s}_r\|\,\rho(t)\end{aligned} \tag{16}$$

Equation 16 allows to conclude that for sufficiently large $\mathbf{K}_D$ values and small initial error conditions, the time derivative of the Lyapunov function is negative semidefinite, implying converge of $\mathbf{s}_r$ to a set-bounded set ε, i.e. $\mathbf{s}_r \to \varepsilon$ as $t \to \infty$. Rewriting Equation 12 as:

$$\begin{aligned}\dot{\mathbf{s}}_r = &-\mathbf{M}^{-1}[\mathbf{K}_D\mathbf{s}_r + \mathbf{C}(\nu)\mathbf{s}_r \\ &+ \mathbf{D}(\nu)\mathbf{s}_r + \mathbf{Y}_r(\nu, \nu_r, \dot{\nu}_r, \eta)\Theta]\end{aligned} \tag{17}$$

and using the fact that all right-hand side terms of this equation are upper bounded (Parra-Vega et al., 2003), it follows that:

$$\|\dot{\mathbf{s}}_r\| \leq \varepsilon_1 \tag{18}$$

Now, deriving Equation 11 and rearranging the terms:

$$\dot{\mathbf{s}}_\nu = -\mathbf{K}_I\mathrm{sign}(\mathbf{s}_\nu) + \dot{\mathbf{s}}_r \tag{19}$$

Multiplying Equation 19 by $\mathbf{s}_\nu^T$:

$$\begin{aligned}\mathbf{s}_\nu^T\dot{\mathbf{s}}_\nu &= -\mathbf{s}_\nu^T\mathbf{K}_I\mathrm{sign}(\mathbf{s}_\nu) + \mathbf{s}_\nu^T\dot{\mathbf{s}}_r \\ &\leq -\lambda_{min}(\mathbf{K}_I)\left|\mathbf{s}_\nu^T\right| + \left|\mathbf{s}_\nu^T\right|\left|\dot{\mathbf{s}}_r\right| \\ &\leq \left|\mathbf{s}_\nu^T\right|(-\lambda_{min}(\mathbf{K}_I) + \varepsilon_1) \\ &\leq -\mu\left|\mathbf{s}_\nu^T\right|\end{aligned} \tag{20}$$

$\lambda_{min}(\mathbf{K}_I) > \varepsilon_1 \to \mu > 0$, assuring sliding mode at $t_s \leq (|\mathbf{s}_\nu(t_0)/\mu|)$ and, since $\mathbf{s}_\nu(t_0) = 0$ for any initial condition, sliding mode in $\mathbf{s}_\nu(t) = 0$ is enforced for all time.

From Equation 9, $\mathbf{s}_d \to \mathbf{0}$ exponentially, and from Equation 10:

$$\mathbf{s}_\nu = -\dot{\xi} - \alpha\xi - \mathbf{s}_d \Rightarrow \dot{\xi} = -\alpha\xi \tag{21}$$

implying exponential convergence of the tracking errors. The full control law after $\mathbf{s}_d$ converges to zero after initialization is shown in Equation 22.

$$\tau_{SM} = \mathbf{K}_D\alpha\xi + \mathbf{K}_D\dot{\xi} + \mathbf{K}_D\mathbf{K}_I \int_0^t \text{sign}(\dot{\xi} + \alpha\xi) d\sigma \tag{22}$$

4 SIMULATION SCENARIO DESCRIPTION

The simulation was built and configured using the UUV Simulator (Manhães et al., 2016b), an open-source underwater simulation package for the robotics simulator Gazebo (Koenig and Howard, 2004) using ROS ("Robot Operating System") as a system application and communication framework. The simulator is built in a modular fashion, which allows an easy setup of the scenarios for different controller algorithms. The information flow between the modules is shown in Figure 3. A view of the 3D visualization tool during the simulation of the scenario with the test trajectory is shown in Figure 2.

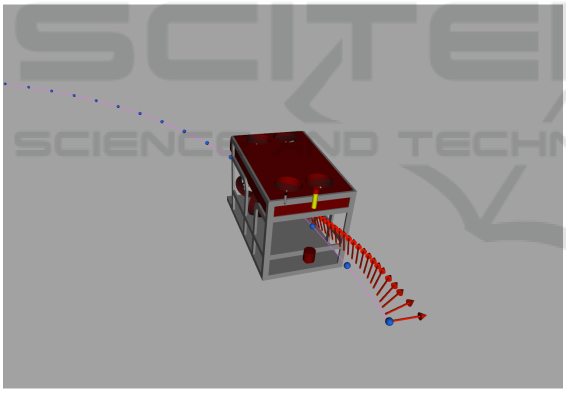

Figure 2: Snapshot from the RViz environment (ROS 3D visualization tool) for the first few seconds after the trajectory following started. The red arrows show the heading for past samples and the pink line represents the path to be followed.

4.1 Vehicle Model

The vehicle considered in this paper is a fully actuated work-class ROV with parameters retrieved from (Berg, 2012) based on the Sperre SF 30k ROV with a slightly different thruster configuration and thruster models. The vehicle's dynamic model, visual and collision geometries can be found as part of the UUV Simulator package (Manhães et al., 2016b).

The thruster units (see Figure 3) include a steady-state curve described by $\tau_i = 0.00031|\Omega|\Omega$, where Ω is the rotor's angular velocity, and a first-order dynamic model to represent the propeller dynamics (Manhães et al., 2016a). Ω_{C_i} is the rotor command velocity is obtained through the inverse of the steady-state angular velocity to thrust function applied to the desired thrust force τ_{C_i}.

4.2 Trajectory and Disturbances

The pose and velocity reference were computed from a helical trajectory. The WORLD frame follows in this case the ENU (East-North-Up) convention since it is the standard for the Gazebo simulator, and is therefore the reference frame for the trajectory generated. The trajectory parameters are listed in Table 1.

Table 1: Reference trajectory parameterized with the controller benchmark.

Parameter	Value
Type of trajectory	Helical
Radius	20 m
Δz	10 m
# of turns	2
Duration	200 s
Center point	$(0, 0, -20)$ m
Start time	5 s

The disturbances are scheduled and generated by a disturbance manager node using the models described in Section 2.2 with respect to Gazebo's WORLD frame in the ENU convention. The disturbance model description and their respective activation and deactivation times can be seen in Table 2.

4.3 Controller Parameter Optimization

A fair comparison of the performance properties of a PID and a model-free sliding mode control is proposed, where both controller parameters are optimized using SMAC (Hutter et al., 2011), a random-forest-based Bayesian optimization algorithm. Bayesian optimization approaches are an adequate choice when the cost function evaluations are expensive to obtain (Brochu et al., 2010).

For both the controllers the SMAC task had to be configured with the controller parameters, its ranges and initial values, the cost function and the maximum number of simulation runs. In both cases, the maximum number of simulation runs was set to 130. The initial values for both controllers have been chosen to deliver a stable closed-loop system.

For the MIMO PID controller presented in Section 3.1, the parameter matrices are defined in Equation

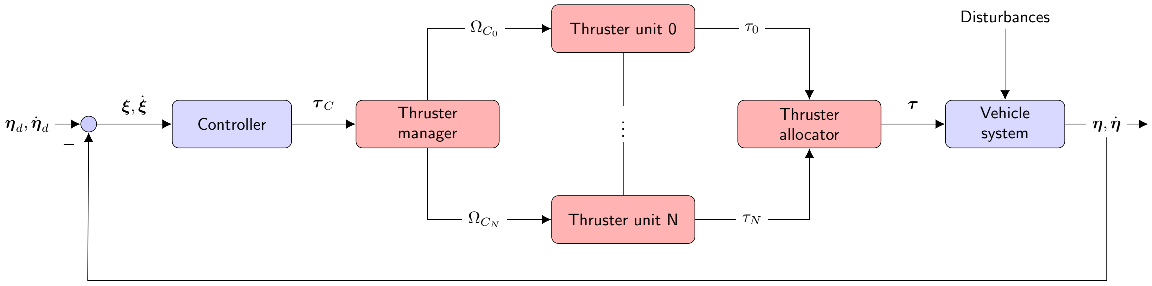

Figure 3: Information flow using the simulation environment.

Table 2: Parameters for the disturbance models used in the optimization process.

Disturbance model	**Value**	**Start time**	**End time**
Wrench	$(3000\text{ N}, 0, -3000\text{ N}, 0, 0, 0)$	25 s	75 s
Current	$(1.2, 0, 0)$ m/s	75 s	125 s
Wrench	$(0, 0, 0, 0, 0, 3000\text{ Nm})$	125 s	150 s
Wrench	$(0, 0, 0, 0, 0, -3000\text{ Nm})$	150 s	175 s

23. The parameter search ranges and initial values are listed in Table 3.

Table 3: Configuration parameters for the MIMO PID parameter optimization.

Parameter	**Range**	**Initial value**
$K_{P_{lin}}$	[100, 20000]	5000
$K_{P_{ang}}$	[100, 20000]	5000
$K_{I_{lin}}$	[0, 5000]	0
$K_{I_{ang}}$	[0, 5000]	0
$K_{D_{lin}}$	[0, 20000]	0
$K_{D_{ang}}$	[0, 20000]	0

$$\begin{aligned} \mathbf{K}_P &= \operatorname{diag}(K_{P_{lin}}\mathbf{I}_{3\times3}, K_{P_{ang}}\mathbf{I}_{3\times3}) \\ \mathbf{K}_I &= \operatorname{diag}(K_{I_{lin}}\mathbf{I}_{3\times3}, K_{I_{ang}}\mathbf{I}_{3\times3}) \\ \mathbf{K}_D &= \operatorname{diag}(K_{D_{lin}}\mathbf{I}_{3\times3}, K_{D_{ang}}\mathbf{I}_{3\times3}) \end{aligned} \tag{23}$$

The model-free sliding mode controller described in Section 3.2 has its matrices defined as shown in Equation 24. The setup used for SMAC is listed in Table 4.

Table 4: Configuration parameters for the model-free sliding mode parameter optimization.

Parameter	**Range**	**Initial value**
$K_{D_{lin}}$	[100, 10000]	2000
$K_{D_{ang}}$	[100, 10000]	200
$K_{I_{lin}}$	[0, 10]	0.005
$K_{I_{ang}}$	[0, 10]	0.3
α_{lin}	[0.01, 5]	1.5
α_{ang}	[0.01, 5]	1.5

$$\begin{aligned} \kappa &= 5 \\ \mathbf{K}_D &= \operatorname{diag}(K_{D_{lin}}\mathbf{I}_{3\times3}, K_{D_{ang}}\mathbf{I}_{3\times3}) \\ \mathbf{K}_I &= \operatorname{diag}(K_{I_{lin}}\mathbf{I}_{3\times3}, K_{I_{ang}}\mathbf{I}_{3\times3}) \\ \alpha &= \operatorname{diag}(\alpha_{I_{lin}}\mathbf{I}_{3\times3}, \alpha_{I_{ang}}\mathbf{I}_{3\times3}) \end{aligned} \tag{24}$$

5 PARAMETRIZATION RESULTS

For the parametrization process, a task scheduler was used to start all needed processes at each optimizer iteration with the new set of parameters and process the simulated data for the computation of the performance metrics needed in the cost function described in Section 2.1. The vehicle starts at the position $(x, y, z) = (20, 0, -20)$ m in the ENU frame of the Gazebo simulator world and with initial orientation set as $(\phi, \theta, \psi) = (0, 0, 0)$ rad, with a initial heading error of $\pi/2$ rad with respect to the initial trajectory reference heading.

SMAC does not have a stopping criteria, it will therefore search for the best parameter set until the maximum number of simulation runs is reached, storing partial results during the process.

The weights of the cost function depicted in Section 2.1 was set as $w_i = 1/RMSE_{i_{MAX}}$, $RMSE_{i_{MAX}}$ being an stipulated maximum acceptable value for the corresponding metric. The weight values used in the cost function according to the nomenclature used in Section 2.1 are $w_1 = 1/5$ m^{-1}, $w_2 = 10/\pi$ rad^{-1}, $w_3 = 10/\pi$ rad^{-1} and $w_4 = 6/\pi$ rad^{-1}.

5.1 MIMO PID Controller Parametrization

The optimal controller parameters were found with a corresponding cost function derived from SMAC's fitted model of $C^{*}_{PID} = 0.66059$ after 28 evaluations, 1.8 hours after the optimizer was started (see Figure 4). The controller parameters found are listed in Table 5.

Table 5: Optimal parameters for the MIMO PID controller after optimization with SMAC.

Parameter	Optimal values
KP_{lin}	11993.888
KP_{ang}	19460.069
KI_{lin}	321.417
KI_{ang}	2096.951
KD_{lin}	9077.459
KD_{ang}	18880.925

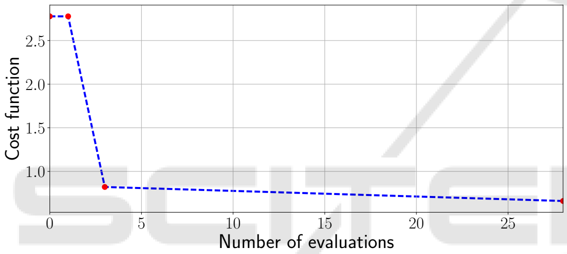

Figure 4: Evolution of performance cost function for the MIMO PID controller.

In Figure 5, the paths generated with the initial and optimal parameter sets are compared against the reference path. It illustrates the improvement of the vehicle during its trajectory tracking task even in the presence of disturbances. The position and heading error curves for initial and optimal parameter sets are shown in Figure 6.

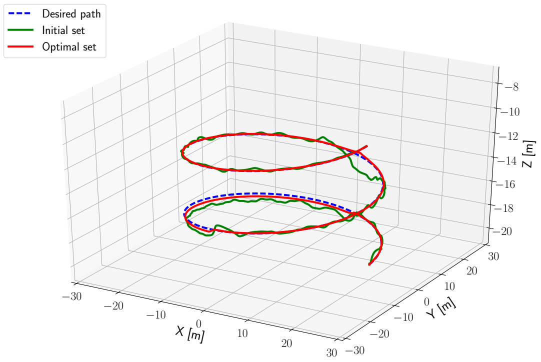

Figure 5: Comparison between desired and actual trajectories for the initial and optimal parameter sets for the MIMO PID controller under the influence of disturbances.

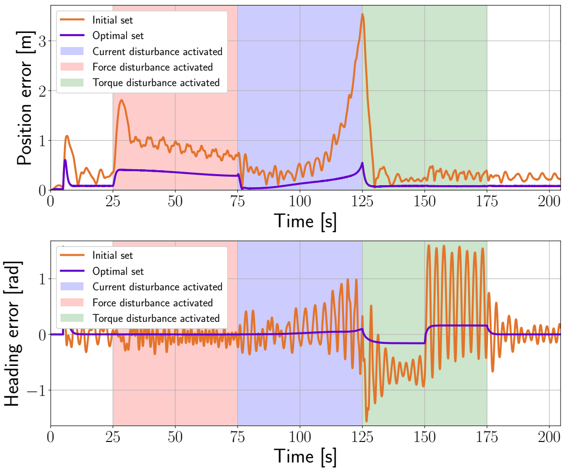

Figure 6: Error curves for the best parameter set for the MIMO PID controller.

5.2 Model-free Sliding Mode Controller Parametrization

The optimal controller parameters were found with a corresponding cost function being $C^{*}_{SM} = 0.378156$ after 98 evaluations, 6.52 hours after the optimizer was started (see Figure 7). The controller parameters found are listed in Table 6.

Table 6: Optimal parameters for the model-free sliding mode controller after optimization with SMAC.

Parameter	Optimal values
KD_{lin}	3243.315
KD_{ang}	5602.003
KI_{lin}	0.134
KI_{ang}	0.169
α_{lin}	0.733
α_{ang}	4.833

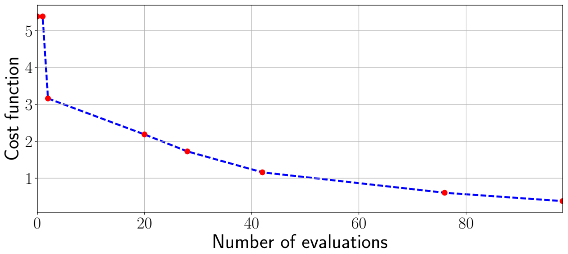

Figure 7: Evolution of the performance cost function for the model-free sliding mode controller.

In Figure 8, the paths generated with the initial and optimal parameter sets are compared with the reference path. The position and heading errors for both the optimal and initial controller parameter sets can be seen in Figure 9.

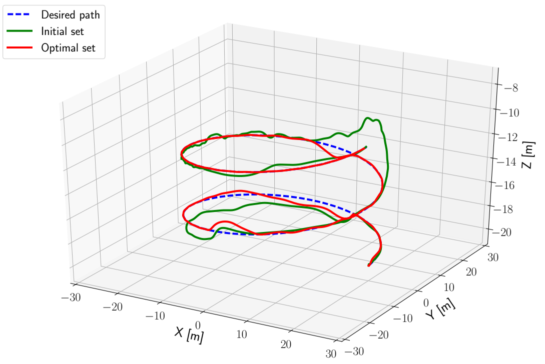

Figure 8: Comparison between desired and actual paths for the initial and optimal parameter sets for the model-free sliding mode controller under the influence of disturbances.

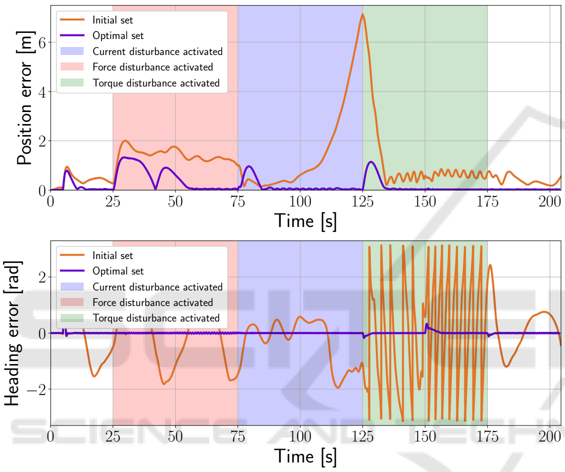

Figure 9: Position error for the best and initial parameter set for the model-free sliding mode controller under the influence of disturbances.

5.3 Comparison of the Resulting Optimized Controller Performances

Upon running both resulting parameter sets in the simulated scenario, the performance metrics used for the computation of the cost function (see Section 2.1) for each use-case using the optimized controller parameters are presented in Table 7. The cost function computed from using the data Table 7 may differ slightly from the SMAC's output presented in Sections 5.1 and 5.2 since they were computed from its calculated model.

Upon analysis of the generated paths and performance metrics, both controllers achieve good results both for position and heading tracking in the presence of disturbances, with the PID controller test showing a better position tracking performance and lower maximum position error (see Figures 6 and 9). It is, however, important to note that the sliding mode controller is inherently robust and can show better results when subjected to different and/or unknown uncertainties. This aspect will be addressed on the next section.

Table 7: Comparison of performance metrics after simulation runs using the optimized controller parameters.

Metric	Sliding Mode	MIMO PID
$RMSE_{\eta_1}$ [m]	0.411	0.212
$RMSE_{\phi}$ [rad]	0.009	0.014
$RMSE_{\theta}$ [rad]	0.306	0.16
$RMSE_{\psi}$ [rad]	0.089	0.122

6 PERFORMANCE COMPARISON WITH A CHANGING CURRENT DISTURBANCE MODEL

As a next analysis, both controllers are put again to the task of tracking a helical trajectory under a different setup of current disturbances (see Figure 10) to test their robustness to a different scenario.

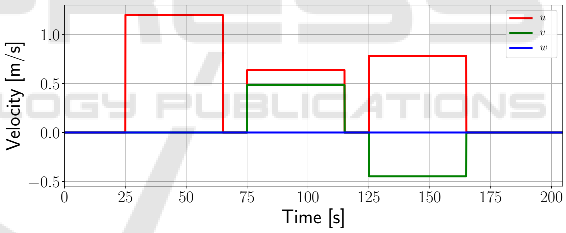

Figure 10: New setup of current velocity disturbances.

Table 8: Parameters for the new current disturbance models used in the validation process.

Current velocity vector	Start time	End time
$(1.2, 0, 0)$ m/s	25 s	65 s
$(0.67, 0.48, 0)$ m/s	75 s	115 s
$(0.78, -0.45, 0)$ m/s	125 s	165 s

The sliding mode controller shows, for the new disturbance set, very similar maximum position errors to those presented on the optimized scenario (see Figures 11 and 12).

The MIMO PID controller, however, offers a poor tracking error performance, particularly during the occurance of the first current disturbance (see Figures 13 and 14).

As stated in Section 5.3, the sliding mode controller has indeed shown better behavior under the

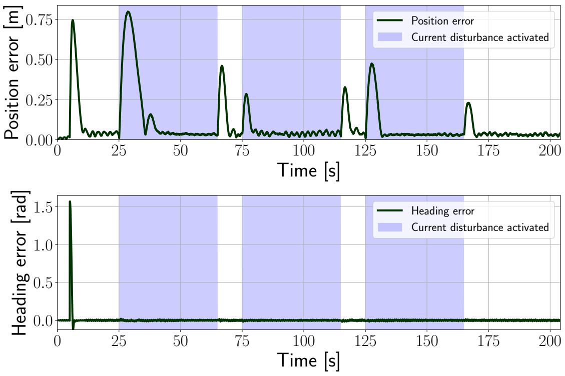

Figure 11: Position and heading error for the model-free sliding mode controller with the new current disturbance model.

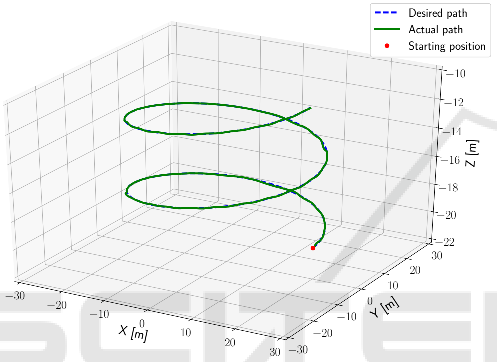

Figure 12: Desired and actual paths generated in the simulation scenario using the optimized model-free sliding mode controller with the new current disturbance model.

new set of disturbances in comparison to the PID controller. The performance metrics for this new scenarios are presented in Table 9 and show that the simulation using the PID controller lead to an $RMSE_{\eta_1}$ almost 20 times higher and a maximum position error 10 times higher than the sliding mode controller.

In an additional example, the same set of disturbances listed in Table 8 was used and the path was generated through linear interpolation with polynomial blends of set of waypoints with the vehicle's maximum forward speed set to 0.5 m/s (see Figure 15). The initial position and orientation of the vehicle were set in both simulation runs as $\eta_1 = (20, 0, -20)^T$ m and $\eta_2 = (0, 0, \pi)^T$ rad with respect to Gazebo's WORLD frame, respectively.This scenario requires the vehicle to change depth and heading several times at designated points instead of the constant heading and depth rate provided by the helical trajectory. The resulting reference and actual trajectories for the sliding mode and the PID controller can be seen in Figures 16 and 17, respectively.

As it can be observed from the two cases, in this scenario of waypoint following, both optimized controller seem fit to be used with for the task regarding the trajectory tracking result. As it can be seen in Table 10, the PID controller showed overall better metrics for the trajectory following aspect even subjected to the same disturbances.

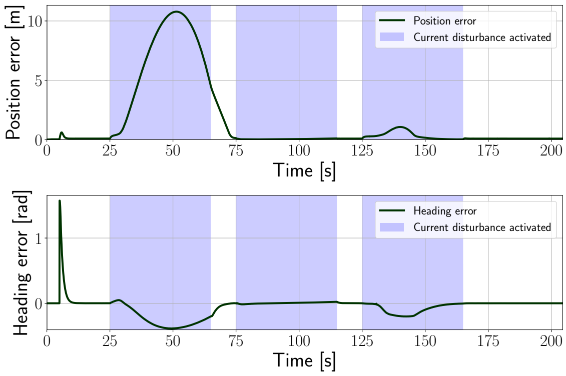

Figure 13: Position and heading error for the PID controller with the new current disturbance model.

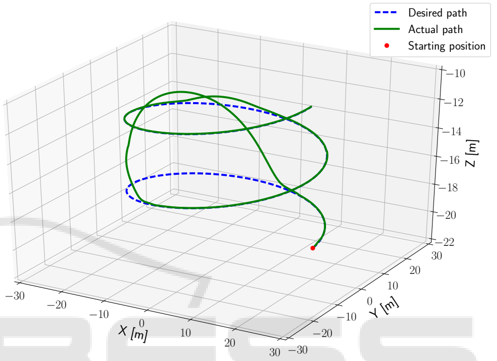

Figure 14: Desired and actual paths generated in the simulation scenario using the optimized PID controller with the new current disturbance model.

It can then be concluded that using the sliding mode controller will lead to a robust system behavior for unmodeled disturbances with a good trajectory tracking performance guaranteed by the optimization step via the consideration of tracking performance metrics in the cost function. However, the optimized PID controller can also provide a good trajectory tracking performance with the knowledge that

Table 9: Comparison of performance metrics using the new disturbance setup.

Metric	Sliding Mode	MIMO PID
$RMSE_{\eta_1}$ [m]	0.177	3.415
$RMSE_{\phi}$ [rad]	0.007	0.157
$RMSE_{\theta}$ [rad]	0.117	0.565
$RMSE_{\psi}$ [rad]	0.083	0.164

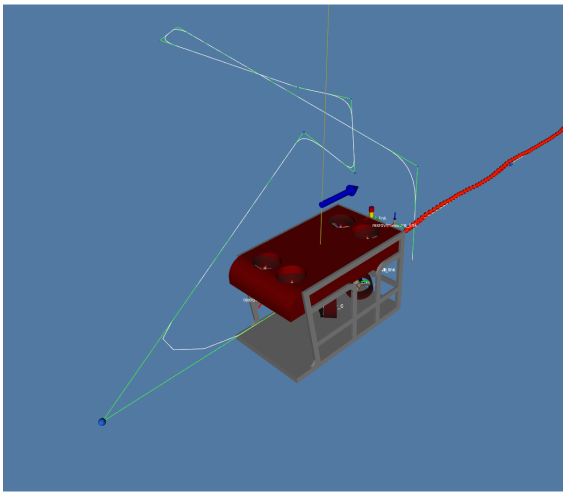

Figure 15: View of the RViz display during the waypoint following scenario. The green path shows the lines connecting the waypoint sequence, the white path represents the result of the path interpolation with polynomial blends, the blue arrow shows the direction of the current velocity vector and the blue spheres represent the waypoints.

Table 10: Comparison of performance metrics using the new disturbance setup for the waypoint following scenario.

Metric	Sliding Mode	MIMO PID
$RMSE_{\eta_1}$ [m]	0.154	0.059
$RMSE_{\phi}$ [rad]	0.007	0.005
$RMSE_{\theta}$ [rad]	0.069	0.036
$RMSE_{\psi}$ [rad]	0.010	0.006

it might lead to high deviations or unstable results under certain environmental conditions and unmodeled model uncertainties considering that it is not inherently robust. The optimization of these control strategies using scenarios where the disturbances models tend to reproduce worst-case examples of possible perturbations that could occur will help to gain certainty on the capability of the control algorithm to be deployed with a specific vehicle before going to field tests with the real system. More extensive tests, such as sensitivity analysis against a range of disturbances, can also be employed to gain more information on the safe operational limits of the closed-loop system and will be a subject of future research.

7 CONCLUSIONS

This paper explores the advantages of a methodical simulation-based controller setup based on its performance evaluation. The optimization takes into account a weighted sum of performance metrics from the simulated scenario and can be setup to also consider possible disturbances that could occur during the mission. The performance analysis can also be further extended, e.g. considering the control efforts and velocity errors.

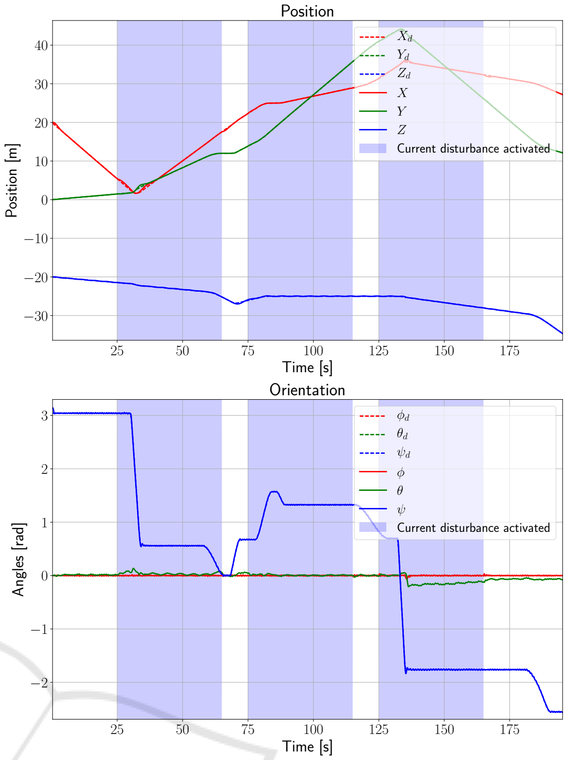

Figure 16: Reference and actual trajectories using the interpolated waypoint set for the model-free sliding mode controller.

This framework is facilitated by the modular structure of the simulation environment used based on ROS and Gazebo. Its structure allows the combination of different vehicles, controller, trajectory generators and disturbance models, that combined with a task scheduler can be easily initialized with the different configuration parameters by, in this case, by an optimization process, allowing a high level of automation of the parameter search method.

After the optimal setup of the two controllers in the use case presented, a numerical comparison between their performances can be fairly evaluated under the same conditions, an aspect that can highly benefit the mission planning phases. This framework will be further developed to consider model-based controllers and a benchmarking procedure to be used on the mission planning for different vehicles and scenarios, also focusing on energy consumption and mission time along with other sources of perturbation, such as thruster failure and model mismatch, and extending the validation process to the real underwater vehicle.

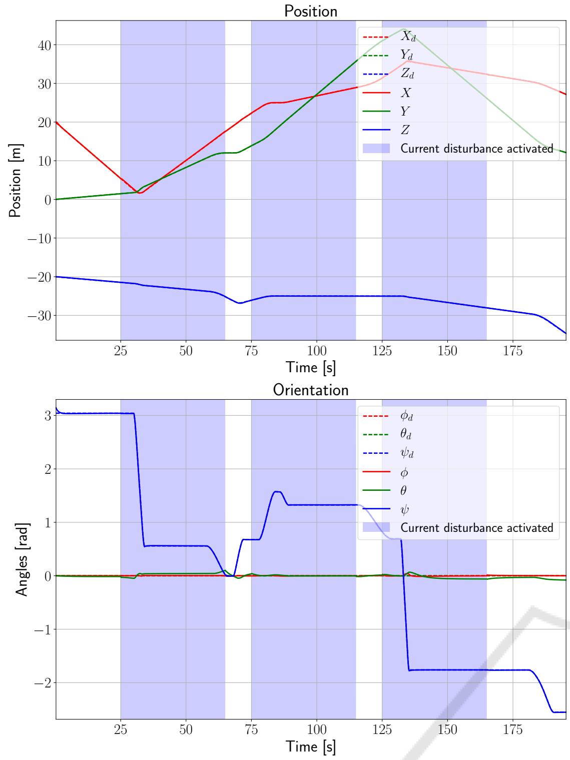

Figure 17: Comparison of results for the trajectory tracking from an interpolated waypoint set for the sliding mode and PID controllers.

ACKNOWLEDGEMENTS

The work in this paper has been developed as part of the work undertaken for the EU-funded SWARMs (Smart and Networking Underwater Robots in Cooperation Meshes) research project (ECSEL project number 662107).

REFERENCES

Antonelli, G. (2014). *Underwater Robots*. Springer International Publishing.

Berg, V. (2012). Development and commissioning of a DP system for ROV SF 30k. Master's thesis, Institutt for marin teknikk, NTNU.

Berkenkamp, F., Schoellig, A. P., and Krause, A. (2016). Safe controller optimization for quadrotors with gaussian processes. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*. Institute of Electrical and Electronics Engineers (IEEE).

Brochu, E., Cora, V. M., and de Freitas, N. (2010). A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. Technical report, arXiv.org.

Calandra, R., Gopalan, N., Seyfarth, A., Peters, J., and Deisenroth, M. P. (2014). Bayesian gait optimization for bipedal locomotion. In *Lecture Notes in Computer Science*, pages 274–290. Springer Nature.

Fossen, T. I. (2011). *Handbook of Marine Craft Hydrodynamics and Motion Control*. Wiley-Blackwell.

García-Valdovinos, L. G., Salgado-Jiménez, T., Bandala-Sánchez, M., Nava-Balanzar, L., Hernández-Alvarado, R., and Cruz-Ledesma, J. A. (2014). Modelling, design and robust control of a remotely operated underwater vehicle. *International Journal of Advanced Robotic Systems*, 11(1):1.

Garcia-Valdovinos, L. G., Salgado-Jimenez, T., and Torres-Rodriguez, H. (2009). Model-free high order sliding mode control for ROV: Station-keeping approach. In *OCEANS 2009, MTS/IEEE Biloxi-Marine Technology for Our Future: Global and Local Challenges*, pages 1–7. IEEE.

Heise, C. D., Leitao, M., and Holzapfel, F. (2013). Performance and robustness metrics for adaptive flight control - available approaches. In *AIAA Guidance, Navigation, and Control (GNC) Conference*. American Institute of Aeronautics and Astronautics (AIAA).

Hosseini, M. and Seyedtabaii, S. (2016). Robust ROV path following considering disturbance and measurement error using data fusion. *Applied Ocean Research*, 54:67–72.

Hutter, F., Hoos, H. H., and Leyton-Brown, K. (2011). Sequential model-based optimization for general algorithm configuration. In *Lecture Notes in Computer Science*, pages 507–523. Springer Nature.

Khadhraoui, A., Beji, L., Otmane, S., and Abichou, A. (2014). Observer-based controller design for remotely operated vehicle ROV. In *Proceedings of the 11th International Conference on Informatics in Control, Automation and Robotics*. Scitepress.

Koenig, N. and Howard, A. (2004). Design and use paradigms for Gazebo, an open-source multi-robot simulator. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*. Institute of Electrical and Electronics Engineers (IEEE).

Lantos, B. and Márton, L. (2011). *Nonlinear Control of Vehicles and Robots*. Springer London.

Lindauer, M., Eggensperger, K., Feurer, M., Falkner, S., Biedenkapp, A., and Hutter, F. (2017). SMAC v3: Algorithm Configuration in Python. https://github.com/automl/SMAC3.

Manhães, M. M. M., Scherer, S. A., and Douat, L. R. (2016a). Unmanned Underwater Vehicle (UUV) Simulation Package for Gazebo. https://github.com/uuvsimulator/uuv_simulator.

Manhães, M. M. M., Scherer, S. A., Voss, M., Douat, L. R., and Rauschenbach, T. (2016b). UUV simulator: A Gazebo-based package for underwater intervention and multi-robot simulation. In *OCEANS 2016 MTS/IEEE Monterey*. Institute of Electrical and Electronics Engineers (IEEE).

Molero, A., Dunia, R., Cappelletto, J., and Fernandez, G. (2011). Model predictive control of remotely operated underwater vehicles. In *IEEE Conference on Decision and Control and European Control Conference*. Institute of Electrical and Electronics Engineers (IEEE).

Parra-Vega, V. and Arimoto, S. (1995). Adaptive control for robot manipulators with sliding mode error coordinate system: free and constrained motions. In *Proceedings of 1995 IEEE International Conference on Robotics and Automation*. Institute of Electrical and Electronics Engineers (IEEE).

Parra-Vega, V., Arimoto, S., Liu, Y.-H., Hirzinger, G., and Akella, P. (2003). Dynamic sliding PID control for tracking of robot manipulators: theory and experiments. *IEEE Transactions on Robotics and Automation*, 19(6):967–976.

Soylu, S., Proctor, A. A., Podhorodeski, R. P., Bradley, C., and Buckham, B. J. (2016). Precise trajectory control for an inspection class ROV. *Ocean Engineering*, 111:508–523.

Stepanyan, V., Krishnakumar, K., Nguyen, N., and Eykeren, L. V. (2009). Stability and performance metrics for adaptive flight control. In *AIAA Guidance, Navigation, and Control Conference*. American Institute of Aeronautics and Astronautics (AIAA).

Zhu, K. and Gu, L. (2011). A MIMO nonlinear robust controller for work-class ROVs positioning and trajectory tracking control. In *2011 Chinese Control and Decision Conference (CCDC)*. Institute of Electrical and Electronics Engineers (IEEE).