Empowering Autonomous Underwater Vehicles Using Learning-based Model Predictive Control With Dynamic Forgetting Gaussian Processes

Abdelhakim Amer, *Graduate Student Member, IEEE*, Mohit Mehndiratta, Yury Brodskiy and Erdal Kayacan, *Senior Member, IEEE*

***Abstract*—Autonomous underwater vehicles (AUVs) present several challenges due to the complex and simultaneous interplay of various factors, including but not limited to unmodeled dynamics, highly nonlinear behaviour, inter-couplings, communication delays, and environmental disturbances. In particular, environmental disturbances degrade trajectory tracking performance for model-based controllers, e.g., model predictive control (MPC) algorithms. Data-driven methods such as Gaussian process (GP) are effective at learning disturbances in real-time, however, the underlying offline hyperparameter tuning process limits their overall effectiveness. To overcome this limitation, we propose a novel dynamic forgetting Gaussian process (DF-GP) methodology that compensates for operational disturbances, thus circumventing the need for hyperparameter retuning. In essence, the proposed method optimally combines the predictions of individual GPs – designed with handcrafted forgetting factors –, rendering precise disturbance estimation of varying timescales. What is more, the predicted disturbances update the model parameters in MPC, facilitating a learning-based control framework that ensures accurate tracking performance in different underwater scenarios. Rigorous simulation and real-world experiments demonstrate the efficiency and efficacy of the proposed framework. The results show a 25% improvement in disturbance estimation and tracking performance, demonstrating that the proposed framework outperforms its direct competitors.**

***Index Terms*—Autonomous underwater vehicle (AUV), model predictive control (MPC), learning-based control, model learning, Gaussian processes (GPs), trajectory tracking.**

I. Introduction

Autonomous underwater vehicles (AUVs) are gaining popularity for pipe inspection and underwater-terrain mapping tasks. As such, fully autonomous operation requires advanced control strategies to navigate accurately and safely in complex underwater environments. In this regard, the optimization-based model predictive control (MPC) algorithm is among the most adopted candidates as it elegantly caters to actuators and states constraints [1, 2]. However, the tracking performance of the MPC algorithm heavily relies on the accuracy of the underlying model. It poses a challenge for AUVs as the estimated models often contain inaccuracies originating from time-varying parameters, unmodeled effects including tether dynamics, and sudden disturbances such as currents. These inaccuracies lead to a suboptimal performance from MPC in AUV navigation.

To address the critical need for a precise system model by MPC, we adopt a learning-based framework that utilizes Gaussian process (GP) models to accurately estimate external disturbances. The main reason to utilize GP over other regression techniques is its non-parametric nature that also furnishes uncertainty distribution along with the prediction value. Conventionally, the hyperparameter tuning of GP models is conducted offline for real-time disturbance estimation. However, pre/offline tuning limits the overall accuracy, especially with time-varying disturbances. To overcome the limitation mentioned above, we propose a novel dynamic forgetting Gaussian process (DF-GP) methodology that imparts adaptive forgetting capability to the underlying GP. In essence, the external disturbances are predicted by several GP models designed with different forgetting factors. The final estimation is the weighted prediction of each GP that is obtained via online performance optimization over a horizon consisting of past measurements. Thanks to this optimal combination, the DF-GP methodology can efficiently learn disturbances with various timescales, thus eliminating the need for hyperparameter retuning. As such, the key outcomes of this study are outlined as follows:

- A learning-based MPC framework built upon a GP architecture. The presented GP structure facilitates efficient disturbance learning of varying time scales along with the unmodeled uncertainties.
- An online adaptive framework that optimally weighs the predictions of GPs with varying forgetting factors.
- Validation of the proposed framework through simulations and real-world experiments.

The subsequent sections are organized as follows: Section II overviews model learning methods. Section III outlines the mathematical model of the AUV, followed by the formulation of MPC in Section IV. The proposed DF-GP approach is then detailed in Section V. Subsequently, Section VI presents the obtained simulation and real-world experimental results. Finally, Section VII derives some conclusions from this work.

Submitted on July 5, 2024. This research receives support from Innovation Fund Denmark grant number 2040-00032B and EIVA a/s.

A. Amer is with the Artificial Intelligence in Robotics Laboratory (AiR Lab), Department of Electrical and Computer Engineering, Aarhus University, 8000 Aarhus C, Denmark abdelhakim@ece.au.dk.

Mohit Mehndiratta is with GIM Robotics, Espoo, Finland, mohit.mehndiratta@gimrobotics.fi.

Y. Brodskiy is with EIVA a/s, 8660 Skanderborg, Denmark. ybr@eiva.com.

E. Kayacan is with the Automatic Control Group (RAT), Department of Electrical Engineering and Information Technology, Paderborn University, Paderborn, Germany. erdal.kayacan@uni-paderborn.de.

II. RELATED WORK

Model learning is a critical component of robot control, enabling precise and robust model-based controller design. One approach focuses on addressing errors in individual parameters separately [3, 4, 5], while another consolidates all unmodeled effects into a single lumped disturbance term. In this work, we adopt the latter method as it renders a more generic model description. Multiple strategies exist for lumped disturbance estimation.

One such approach underlies characterizing the unknown disturbances by a parametric model, for instance, a neural network (NN) [6, 7]. However, large NN models suffer from issues including insufficient excitation, estimation windup, and the necessity for a large amount of training data. Alternative methods for learning disturbances involve utilizing a disturbance observer, such as an extended Kalman filter [8], or adopting stochastic models like GP [9].

Unlike other models, GP is a non-parametric regression technique that does not impose structure-specific requirements on the learned nonlinear models. Instead, it relies on general assumptions regarding continuity and differentiability. Moreover, a GP model provides uncertainty quantification, enabling the implementation of conservative, thus safer control strategies [10, 11]. GP has also proven real-time applicability, as shown in [12, 13]. In [14], a cascaded GP architecture accurately learns wind disturbances on a drone based on a historical sequence of disturbance observations. A hierarchical control scheme, as presented in [15], uses constrained GP-based MPC for translational and rotational subsystems, effectively simplifying nonlinear MPC problems into convex ones. Additionally, [16] presents a framework that integrates output feedback MPC with evolving GP, ensuring constraint satisfaction and stability while reducing computational load. Within high-speed indoor flight, [17] utilized a GP model to capture unmodeled complex drag effects. [18] integrated a moving horizon estimator (MHE) with a GP to achieve simultaneous state estimation and disturbance rejection.

Although the methods mentioned above have demonstrated significant enhancements over conventional MPC algorithms and have shown practical viability in real-time applications, a notable challenge they encounter is their adaptability to unforeseen conditions. This challenge is associated with GP models that rely on pre/offline tuned hyperparameters, facilitating real-time application but potentially limiting their ability to respond to dynamic environments. Recently, a fast online adaptive GP incorporating a forgetting factor into the GP formulation has been proposed [19]. In essence, it facilitates the adaptation of time series models that exhibit rapid variations. While this approach has shown promising results, selecting the appropriate forgetting factor online remains unresolved.

Hence, this work builds on [14] and [19], introducing a partitioned dataset to explicitly capture mean and transient disturbance components, while employing a forgetting factor to balance their contribution to disturbance prediction. Inspired by locally weighted models GP [20], which weighs GP models with different training data and hyperparameters, a weighted average is applied to GP models with shared data and hyperparameters but differing forgetting factors. The weight of each GP is then optimized over a horizon, enabling adaptation to disturbances without the need for an online hyperparameter optimization.

III. AUV MODELLING

The equations of motion for the rigid body are described in a body-fixed coordinate system in the North-EastDown (NED) frame (See Fig. 1) [21]. Neglecting roll and pitch motion due to the restoring moment action significantly simplifies the equations of motion. This assumption, as adopted in [22, 23], enables more focus on the dominant dynamics of the system, which can be written as follows:

$$\dot{x} = \cos(\psi)u - \sin(\psi)v \tag{1}$$
$$\dot{y} = \sin(\psi)u + \cos(\psi)v \tag{2}$$
$$\dot{z} = w \tag{3}$$

$$\begin{aligned}(m - X_{\dot{u}})\dot{u} &= X + (mv + Y_{\dot{v}}v)r \\ &\quad + (X_u + X_{uc}|u|)u + \Delta_x \end{aligned} \tag{4}$$
$$\begin{aligned}(m - Y_{\dot{v}})\dot{v} &= Y - (mu + X_{\dot{u}}u)r \\ &\quad + (Y_v + Y_{vc}|v|)v + \Delta_y \end{aligned} \tag{5}$$
$$\begin{aligned}(m - Z_{\dot{w}})\dot{w} &= Z + (Z_w + Z_{wc}|w|)w \\ &\quad + mg - V_{sub}\rho_{water} + \Delta_z \end{aligned} \tag{6}$$
$$\dot{\psi} = r \tag{7}$$
$$\begin{aligned}(I_{zz} - N_{\dot{r}})\dot{r} &= M_z - (mv - Y_{\dot{v}}v)u \\ &\quad - (X_{\dot{u}}u - mu)v \\ &\quad + (N_r + N_{rc}|r|)r + \Delta_{M_z} \end{aligned} \tag{8}$$

In the equations above, the translational positions x, y, and z are specified within the earth-fixed frame $\mathcal{F}_E$. The translational velocities u, v, and w are represented in the body frame $\mathcal{F}_B$, whereas r denotes the rotational velocity about the z-direction. Translational velocities in $\mathcal{F}_E$ can be obtained by rotating the body velocities with an angle ψ about the heave direction. The vehicle's mass is denoted by m, the angular moment of inertia about the z-axis is denoted as I_{zz}, g describes the gravitational constant, ρ_{water} is the water density, and V_{sub} denotes the total submerged volume. X, Y, and Z denote applied wrenches by actuators acting on the AUV body in $\mathcal{F}_B$, while M_z represents

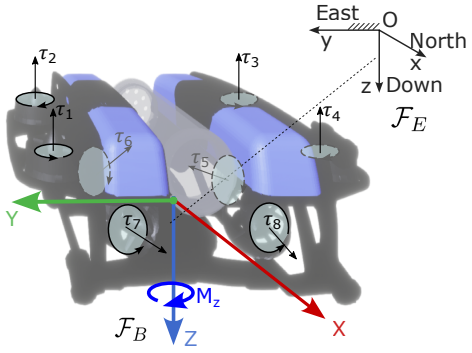

Fig. 1: AUV free body diagram and reference frames [24].

the externally applied moment about the z-axis. Here, $\mathbf{\Delta} \in \mathbb{R}^4$ given by $[\Delta_x,\ \Delta_y,\ \Delta_z\ \Delta_{M_z}]$ are the external disturbances affecting the AUV body in the body frame. This term accounts for the lumped unmodeled disturbances arising from currents, tether, and other effects.

The parameters $X_{\dot{u}}$, $Y_{\dot{v}}$, $Z_{\dot{w}}$, and $N_{\dot{r}}$ represent added mass parameters. X_u, Y_v, Z_w, and N_r, as well as X_{uc}, Y_{vc}, Z_{wc}, and N_{rc}, denotes linear and quadratic drag coefficients about the surge, sway, heave, and yaw directions, respectively. The parameters for the BlueROV2 heavy model, used for both simulation and controller model, are provided in Table I. The equations of motion can also be rewritten in state-space form:

$$\dot{\mathbf{x}} = \mathtt{f}(\mathbf{x}_k, \mathbf{u}_k, \mathbf{\Delta}), \tag{9}$$

The vectors $\mathbf{x} \in \mathbb{R}^8$ and $\mathbf{u} \in \mathbb{R}^4$ denote the state and control vectors, respectively

$$\mathbf{x} = [x, y, z, u, v, w, \psi, r]^T, \tag{10}$$

$$\mathbf{u} = [X, Y, Z, M_z]^T. \tag{11}$$

The state transition function is represented by $\mathtt{f}(\cdot,\cdot,\cdot)$: $\mathbb{R}^8 \times \mathbb{R}^4 \times \mathbb{R}^4 \to \mathbb{R}^8$.

IV. Model Predictive Control

MPC is an optimization-based control strategy aimed at determining an optimal sequence of actions over a predefined time horizon, denoted by N_c. This optimization process is conducted at each time step in a receding horizon manner after the system's initial state is updated. The optimal control problem is given in discrete time as follows:

$$\min_{\mathbf{x}_k, \mathbf{u}_k} \quad \sum_{k=0}^{N_c-1} \left(\|\mathbf{x}_k - \mathbf{x}_k^{ref}\|^2_{\mathbf{Q}} + \|\mathbf{u}_k\|^2_{\mathbf{R}} \right) + \|\mathbf{x}_{N_c} - \mathbf{x}_{N_c}^{ref}\|^2_{\mathbf{Q}_T} \tag{12a}$$

$$\text{s.t.} \quad \mathbf{x}_{k+1} = \mathtt{f}_d(\mathbf{x}_k, \mathbf{u}_k, \mathbf{\Delta}), \quad k \in [0, N_c - 1], \tag{12b}$$

$$\mathbf{u}_{k,\mathrm{l}} \leq \mathbf{u}_k \leq \mathbf{u}_{k,\mathrm{u}}, \quad k \in [0, N_c - 1]. \tag{12c}$$

The discretized nonlinear model, denoted as $\mathtt{f}_d(\mathbf{x}_k, \mathbf{u}_k, \mathbf{\Delta})$, is obtained by applying a 4th-order Runge-Kutta method to the continuous dynamics in (9). The equations are discretized using a chosen sampling time T_s. Additionally, a zero-order hold assumption is adopted for the control inputs, whereby the control inputs $\mathbf{u}$ remain constant throughout each sampling interval T_s.

Herein, $\mathbf{u}_{k,\mathrm{l}}$ and $\mathbf{u}_{k,\mathrm{u}}$ denote the respective minimum and maximum bounds on the controls, where each is a vector in $\mathbb{R}^4$. The components inside the summation in (12a) represent costs associated with the full state and control vectors (10) and (11). While the state cost penalizes the deviations between the predicted $\mathbf{x}_k$ and reference $\mathbf{x}^{ref}$ states, the control cost minimizes energy consumption along the trajectory. The second term in (12a) is the terminal cost, which caters to the finite nature of the prediction horizon, thereby ensuring convergence. In this context, $\mathbf{Q} \in \mathbb{R}^{8\times8}$, $\mathbf{Q}_T \in \mathbb{R}^{8\times8}$, and $\mathbf{R} \in \mathbb{R}^{4\times4}$ are positive-(semi) definite weight matrices. In this work, we obtain the coefficients of the weight matrices using the meta-heuristic strategy described in [25].

V. Dynamic Forgetting Gaussian Process

In this section, we elaborate on the overall DF-GP methodology proposed in this work. We begin with summarizing the nominal dense GP expressions, followed by the adaptive sparse gaussian process (ASGP) method proposed in [19]. Then, we illustrate the implementation details of the online ASGP method along with the adopted GP architecture. Finally, we deduce the optimization problem of the DF-GP methodology and establish its global convergence.

A. Gaussian process preliminaries

In essence, a GP is a set of infinitely-dimensional random variables characterized by a joint Gaussian probability distribution [26]. Let there be data set $\mathbf{D}$ of size n given by

$$D_i = \{\mathbf{a}_i, y_i\}, \qquad i \in [1, n], \tag{13}$$

where $\mathbf{a}_i \in \mathbb{R}^d$ and y_i are the input-target of observations, with d representing the input dimensionality. The goal is to estimate an unknown function $f(\mathbf{a}_i)$ that maps the inputs $\mathbf{a}_i$ to the target $\mathbf{y}_i$:

$$y_i = f(\mathbf{a}_i) + \epsilon_i, \qquad \epsilon_i \sim \mathcal{N}(0,\, \sigma^2_{\varepsilon_i}), \tag{14}$$

where ϵ_i is an additive white noise, with a variance $\sigma^2_{\varepsilon_i}$. For simplicity, we will henceforth omit the index i. A prior on the underlying function can be deduced, such that $f(\mathbf{a})$ forms a GP defined by a mean function $m(\mathbf{a})$ and covariance kernel $k(\mathbf{a}, \mathbf{a}')$.

A prediction $f_* = f(\mathbf{a}_*)$ at a test point $\mathbf{a}_*$ assumes that the vector of target values $\mathbf{y} = [y_1, \ldots, y_n]^\top$ and the prediction f_* have a joint Gaussian distribution given by:

$$\begin{bmatrix} \mathbf{y} \\ f_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K}_{aa} + \sigma^2_\varepsilon \mathbf{I} & \mathbf{k}_{a*} \\ \mathbf{k}_{*a} & k_{**} \end{bmatrix}\right), \tag{15}$$

with the covariance calculated using a squared exponential kernel, as follows:

$$k(\mathbf{a}, \mathbf{a}') = \sigma_f^2 \exp\left(-\frac{1}{2}(\mathbf{a} - \mathbf{a}')^T \mathbf{L}_a^{-2} (\mathbf{a} - \mathbf{a}')\right), \tag{16}$$

wherein $\sigma_f^2 \in \mathbb{R}^+$ denote the prior variance, and $\mathbf{L}_a \in \mathbb{R}_{>0}$ is a diagonal matrix that represents the input length scales. Together, these parameters form the kernel hyperparameters $\theta \in \mathbb{R}^{d+1}$, which determine the characteristics of the functions that the GP samples from. Typically, the optimal hyperparameters, θ_{opt}, are obtained by minimizing the negative log marginal likelihood represented as follows:

$$\theta_{\text{opt}} = \arg\min_\theta \left(\frac{1}{2}\mathbf{y}^T \mathbf{K}_{aa}^{-1} \mathbf{y} + \frac{1}{2}\log|\mathbf{K}_{aa}| + \frac{n}{2}\log 2\pi\right). \tag{17}$$

Finally, the joint probability distribution (15) is conditioned on $\mathbf{y}$ to calculate the posterior distribution of f_*, resulting in the following posterior prediction equations for the test point $\mathbf{a}_*$:

$$\mu_* = \mathbf{k}_{*a}\left(\mathbf{K}_{aa} + \sigma^2_\varepsilon \mathbf{I}\right)^{-1} \mathbf{y}, \tag{18}$$

$$\Sigma_* = k_{**} - \mathbf{k}_{*a}\left(\mathbf{K}_{aa} + \sigma^2_\varepsilon \mathbf{I}\right)^{-1} \mathbf{k}_{a*}. \tag{19}$$

TABLE I: BlueROV2 heavy parameters

$X_{\dot{u}}$	$Y_{\dot{v}}$	$Z_{\dot{w}}$	$N_{\dot{r}}$	X_u	Y_v	Z_w	N_r	X_{uc}	Y_{vc}	Z_{wc}	N_{rc}	m	V_{sub}	I_{zz}
-2.6	-18.5	-13.3	-0.28	-0.09	-0.26	-0.19	-4.64	-34.9	-103.25	-74.2	-0.43	11.4	115	0.24

B. Adaptive sparse Gaussian process

A recent work in [19] introduces an adaptive version of the variational sparse GP method, referred to as the ASGP, which utilizes the following mean and variance equations:

$$\mu_*(\lambda) = \sigma_\varepsilon^{-2}\mathbf{k}_{*s}\mathbf{B}_\lambda\mathbf{K}_{sa}\mathbf{\Lambda}\mathbf{y}, \tag{20}$$

$$\Sigma_*(\lambda) = k_{**} + \mathbf{k}_{*s}\left(\mathbf{B}_\lambda - \mathbf{K}_{ss}^{-1}\right)\mathbf{k}_{s*}, \tag{21}$$

where $\mathbf{\Lambda}$ (forgetting factor matrix) and $\mathbf{B}_\lambda$ are expressed as:

$$\mathbf{\Lambda} = \operatorname{diag}\begin{pmatrix}\lambda^{n-1} & \lambda^{n-2} & \cdots & \lambda^0\end{pmatrix}, \tag{22}$$

$$\mathbf{B}_\lambda = \left(\mathbf{K}_{ss} + \sigma_\varepsilon^{-2}\mathbf{K}_{sa}\mathbf{\Lambda}\mathbf{K}_{as}\right)^{-1}. \tag{23}$$

The equations above utilize a vector of inducing points $\mathbf{s}_m$, where $m \in \{1, \ldots, M\}$. There are M such inducing points, which together provide a sparse approximation of the dense input $\mathbf{a}$, enabling computationally efficient and real-time prediction capabilities. Furthermore, introducing a forgetting factor enables prediction equations tailored for time-series data. This factor prioritizes recent data points with exponentially increasing weights, emphasizing recent data in the prediction.

C. Online ASGP for disturbance prediction

Next, we describe the procedure for generating the data set $D = \{\mathbf{a}, y\}$, which is used by the ASGP for predicting the disturbance at the following time step ($\hat{\Delta}_{t+1} = \mu_*(\lambda)$). At an arbitrary measurement time τ, the input vector and target scalar are defined in the following manner:

$$\mathbf{a} = [\Delta_{\tau-1-H}, \mathbf{x}_{\tau-1-H}, \mathbf{u}_{\tau-1-H}, \cdots, \Delta_{\tau-1}, \mathbf{x}_{\tau-1}, \mathbf{u}_{\tau-1}], \tag{24}$$

$$y = \Delta_\tau. \tag{25}$$

The input vector $\mathbf{a}$ is constructed utilizing the historical data, wherein H represents the length of the data history. The disturbance value at time instance τ (Δ_τ) is computed from the model equations (9), by incorporating the corresponding acceleration and velocity feedback along with the control input.

It is important to note that there is a one-time step difference between the input and output to facilitate learning. This input-output choice renders the GP model more effective in learning disturbances. Also, the feature space is augmented with states and control inputs to embed the associated dynamics information. Moreover, a separate GP model is employed for disturbance estimation along each axis to streamline the input space and reduce computational complexity. Furthermore, the dataset is partitioned into two sets as proposed in [14], namely D^1 and D^2. The static dataset D^1, collected at the beginning of the operation, is utilized for hyperparameter optimization. On the other hand, the dynamic dataset D^2 is updated online, wherein the latest acquired data point replaces the first element (oldest data point). This design allows the forgetting factor to balance between the offline collected dataset D^1, representing nominal model errors, and the online collected dataset D^2, representing sudden disturbances, such as currents.

D. Dynamic weight optimization

As mentioned earlier, the forgetting factor in the ASGP formulation can be crucial in time-series data prediction, making it suitable for predicting rapidly changing disturbances. However, a critical challenge lies in determining the optimal value for this factor. One potential approach is to choose the forgetting factor that maximizes the log-likelihood. However, this method can be computationally demanding and may struggle to adapt effectively to dynamic datasets.

To address the limitation above, a dynamic forgetting approach is proposed to effectively weigh the predictions from multiple GP models with varying forgetting factors. The formulation of the proposed optimization problem is as follows:

$$\min_{\eta_j} \quad \sum_{j=1}^{K}\sum_{i=1}^{N}\eta_j\|\mu_{*t-i-1}^{j}(\lambda_j) - \Delta_{t-i}\|_{\alpha_i}^2, \tag{26a}$$

$$\text{s.t.} \quad \sum_{j=1}^{K}\eta_j = 1, \quad j \in [1, K], \tag{26b}$$

$$\eta_j \geq 0, \qquad j \in [1, K]. \tag{26c}$$

Here, K denotes the number of GP models used in the prediction, where K is a user-defined number. In essence, the objective is to obtain the optimal weights (η_j) that minimize the Euclidian error between the individual GP prediction ($\mu_{*t-i-1}^{j}(\lambda_j)$) and the corresponding measured disturbance values (Δ_{t-i}). Note that the optimization problem in (26a) is solved over a historical measurement horizon comprising N data points. This measurement horizon ensures that the weighted combination results in higher prediction accuracy over consecutive time points, rather than just the current time, rendering a stable optimal solution. As such, N can be inferred as a critical parameter that substantially influences the optimization process. Opting for a small N might hinder the ability of the GP model to capture past effects, i.e., making it near-sighted, thereby rendering overemphasis on the recent data. Conversely, selecting a large N introduces computational burdens and reduces the responsiveness of the GP model to recent disturbances.

Additionally, α_i represents the relative weightage of GP prediction errors within the N-point horizon. Intuitively, an exponential expression $\alpha_i = e^{0.05(N-i)}$ is defined. Exponential weighting via α_i allows the model to prioritize recent data, improving its ability to track fast-changing disturbances. For instance, if α_i is set as unity, the model will treat all past errors equally, reducing its adaptability to recent disturbances. To maintain consistency among GP model predictions, the constraints in (26a) and (26c) are introduced.

Remark 1. *It is important to distinguish between α_i and the the forgetting factor λ: while λ determines the contribution of historical data points in the GP mean computation, α_i weighs the different prediction-error terms over the measurement horizon N in the optimization process for the weights η.*

The overall learning-based MPC framework, as depicted in Fig. 2. Furthermore, A pseudo-code for the overall frame is provided in algorithm 1. Note that the algorithm is presented for a single dimension of the GP. The same procedure is applied independently to the x, y, and z directions to learn disturbances along each respective axis.

Algorithm 1 Dynamic Forgetting Gaussian Process (DF-GP)

Input:
Measurement horizon H
Number of GP models K
Forgetting factors λ_j
Generate datasets D^1 and D^2, incorporating (24),(25)
Optimize GP hyperparameters, using (17)
while learning is active **do**
  Get current state and control
  Update D^2, append new data and remove old
  for each GP model with λ_j **do**
    predictions μ_{gpj}, using (20)
  end for
  Optimize weights η^{opt} over horizon N, using (26)
  Predict disturbance $\hat{\Delta}_{t+1}$ with η^{opt} using (28).
  Update MPC model with $\hat{\Delta}_{t+1}$, using (12a)
end while

Feasibility and optimality: To analyze the feasibility and (global) optimality, the optimization problem in (26) can be represented in the following standard form:

$$\min_{\boldsymbol{\eta}} \quad \boldsymbol{c}^T\boldsymbol{\eta}, \tag{27a}$$
$$\text{s.t.} \quad \boldsymbol{a}^T\boldsymbol{\eta} = b \tag{27b}$$
$$\boldsymbol{\eta} \geq \mathbf{0}, \tag{27c}$$

where (27a) defines the cost function with the coefficient vector $\mathbf{c} \in \mathbb{R}^{K\times 1}$ written as follows:

$$\mathbf{c} = \begin{bmatrix} \sum_{i=1}^{N} \alpha_i (\mu^1_{t-i-1} - \Delta_{t-i})^2 \\ \sum_{i=1}^{N} \alpha_i (\mu^2_{t-i-1} - \Delta_{t-i})^2 \\ \vdots \\ \sum_{i=1}^{N} \alpha_i (\mu^K_{t-i-1} - \Delta_{t-i})^2 \end{bmatrix} \in \mathbb{R}^{K\times 1} \neq \mathbf{0},$$

and (27b) and (27c) represent the underlying constraints with the following variable definitions:

$$\boldsymbol{a} = \begin{bmatrix} \mathbb{I} \end{bmatrix} \in \mathbb{R}^{K\times 1}, \quad b = 1,$$

wherein $\mathbb{I}$ infers an identity vector. This formulation is clearly a linear program (LP) due to its affine objective and constraint functions, demonstrating the convex nature of the problem. Given the inherent convexity of linear programs, global convergence to the optimal solution is ensured, assuming feasibility. Efficient LP solvers can thus reliably solve this problem.

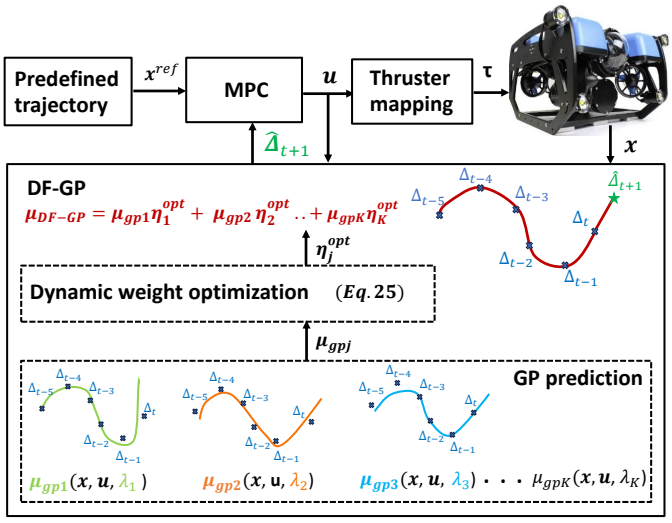

Fig. 2: The proposed learning-based MPC framework.

Remark 2. *The optimization problem in* (27) *is a linear program that guarantees the existence of a solution. Non-uniqueness may occur if the vector* $\mathbf{c}$ *contains identical or zero elements. However, in most practical scenarios where K is a small integer, a unique solution is typically achieved while ensuring feasibility.*

The obtained optimal weights η_j^{opt} are then utilized to obtain a new prediction equation for the DF-GP as follows:

$$\mu_{*DF} = \sum_{j=1}^{K} \eta_j^{opt} \mu_{*t}(\lambda_j), \tag{28}$$

$$\Sigma_{*DF} = \sum_{j=1}^{K} \eta_j^{opt} \Sigma_{*t}(\lambda_j). \tag{29}$$

VI. Experimental Results

The algorithm's predictive accuracy and trajectory tracking capabilities are assessed against state-of-the-art benchmarks through simulations and indoor pool testing.

A. Implementation details

The scenario is simulated using the underwater robotic simulator, Mobula [1]. The inducing points are chosen to be identical to the inputs for consistency between the different methods. The first ten seconds of data are recorded for training the GP, where the conjugate gradient method is employed to optimize the hyperparameters. Regarding implementing the proposed DF-GP approach, we set K to be three, which implies solving the optimization problem (26) for three GPs with forgetting factors namely, 1, 0.8, and 0.6. As such, the number of selected GP models is a user-defined configuration parameter. While adding more GP models is expected to enhance performance, it also increases the computational burden. Additionally, the measurement horizon N is set to 20 samples

[1]https://www.eiva.com/products/navisuite/navisuite-bundles/navisuite-mobula-pro-videoray

which is determined via the trial-and-error method. Moreover, the DF-GP method is implemented using the CasADi [27]. For the MPC, which is used as the tracking controller with all the GP models, a direct multiple shooting method with a 0.05-second grid size is used. The solution is obtained via solving a sequential quadratic program with a combination of the generalized Gauss-Newton method and the real-time iteration scheme. ACADO toolkit [28] and qpOASES solver [29] handle the solution for (12), with $N_c = 100$ prediction horizon and 0.05 second sampling time. A long horizon was selected which ensures accurate tracking, improved safety, and reliable constraint satisfaction.

B. Simulation experiments

A lemniscate trajectory, comprising two cycles, is provided as a reference to the MPC, while the scenario includes the introduction of a disturbance profile consisting of a sine wave, a combined sine wave (combination of different sine waves as included in [14]), and a square wave. We first motivate the inclusion of a dynamic forgetting factor by implementing our method with fixed forgetting values, namely $\lambda = 1.0$, $\lambda = 0.8$, and $\lambda = 0.6$. Additionally, four different baselines are implemented: an MPC without a GP, the GP-MPC method proposed in [14], the fast adaptive sparse GP (Fast-AGP) proposed in [19], the MHE disturbance observer augmented with MPC proposed in [30], and finally, the proposed method, DF-GP. Moreover, we also test the MPC without disturbances to illustrate the best-case tracking performance.

The overall performance enhancement of the DF-GP compared to other methods is illustrated in Table II. Particularly, regarding the motivation for dynamic forgetting compared to a static forgetting factor, while a fixed forgetting factor provides a prediction error of about 0.66 m/s^2, the DF-GP outperforms it by reducing the prediction error to less than half, i.e., to 0.289 m/s^2. The reason is that a fixed forgetting factor is not suitable for different disturbance scenarios. Initially, when the disturbance is repeated, a higher λ value renders a large dataset which is crucial for accurate prediction. Conversely, when the disturbance profile abruptly changes, such as in combined disturbances or square wave cases, a lower λ is needed as older data becomes less relevant and recent data gains importance. The DF-GP method strikes a balance as it weights different GP models and quickly adapts predictions based on the error of each model within a short horizon.

TABLE II: Simulation results summary: The DF-GP method demonstrates significant improvements in prediction compared to baselines. The improvement percentage is calculated relative to the GP-MPC method in disturbance prediction and relative to the MPC with no GP in tracking performance.

Method	Mean prediction error $[m/s^2]$	Improve. %	Mean traj. error [m]	Improve. %	Comp. [ms]
Ideal	-	-	*0.0186*	*82.3*	-
No GP	-	-	0.105	0.0	-
MHE-MPC [30]	0.445	37.8	0.0889	15.3	2.41
GP-MPC [14]	0.716	0.0	0.0597	43.1	1.25
Fast-AGP [19]	0.381	46.8	0.0378	64.0	0.748
$\lambda = 1.0$	0.644	10.1	0.0519	50.6	0.788
$\lambda = 0.8$	0.674	5.97	0.0505	51.9	**0.720**
$\lambda = 0.6$	0.656	8.38	0.0514	51.0	0.756
DF-GP	**0.289**	**59.6**	**0.0285**	**72.9**	2.05

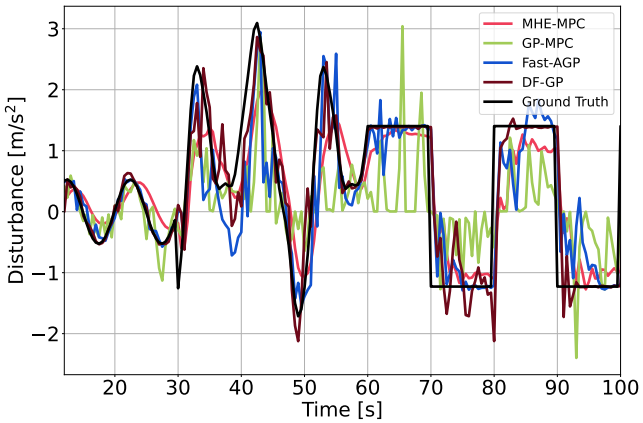

Fig. 3: GP predictions during the simulation run. DF-GP effectively predicts the disturbance profile by optimally combining contributions from different GPs.

Figure 3 illustrates the disturbance profile alongside the predictions generated by GP model from different methods. Up until the first 30 seconds, all GPs are able to predict the disturbance profile relatively well. Starting at time $t = 30$, the disturbance profile shifts to a combined sine wave, where the GP-MPC [14] is failing to follow the profile, as it has not encountered it before in the training data. On the other hand, the Fast-AGP and DF-GP are able to accurately follow the disturbance profile, even as it switches. This can be attributed to the forgetting factor and the dynamic selection of inducing points in [19]. Notably, our DF-GP method demonstrates a superior ability to quickly adapt to the changing disturbance profile and maintain accuracy, as evidenced by the error plot in Fig. 4. This capability is due to its innovative approach of balancing multiple predictions by optimizing the weights of these predictions, enabling it to dynamically adjust and

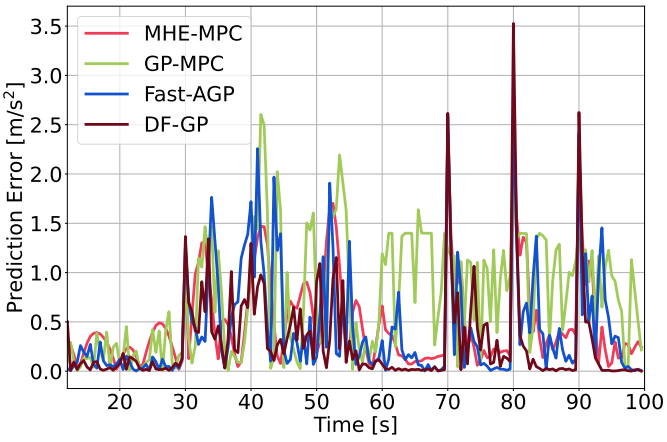

Fig. 4: GP predictions error in simulation for the GP with different forgetting factors. The DF-GP method maintains low prediction errors.

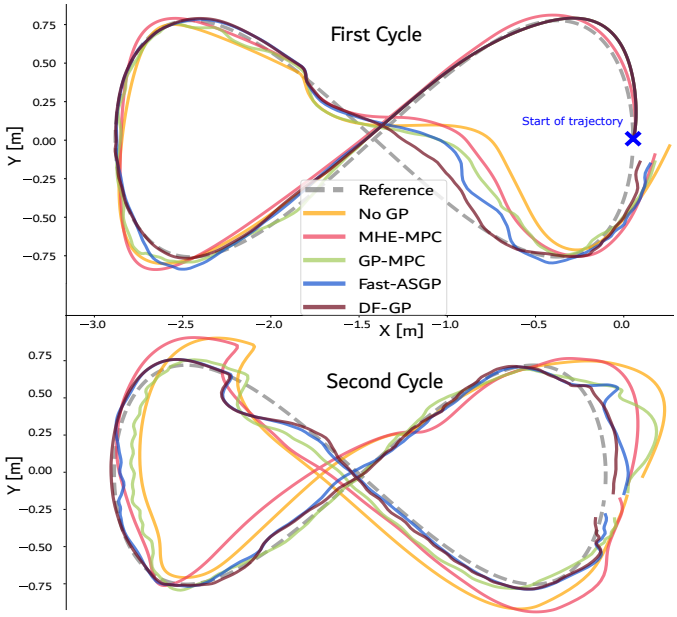

Fig. 5: Obtained trajectory in simulation for the proposed DF-GP method against baselines.

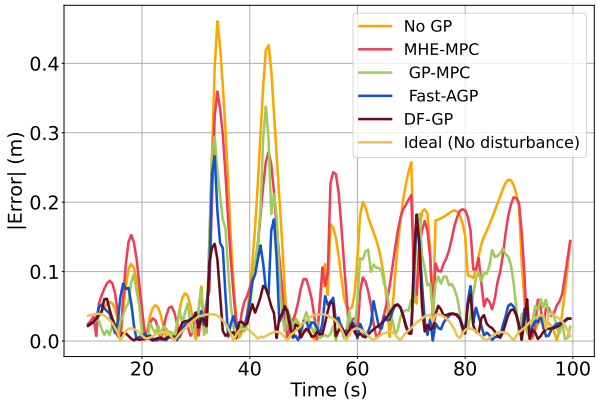

Fig. 6: Trajectory tracking error results. The peaks in the error correspond to when the disturbance profile changes and the applied disturbance reaches a maximum magnitude.

respond to the changing disturbance profile.

Figure 5 presents the tracking accuracy of the DF-GP method compared to the baselines MPC. It is evident that DF-GP outperforms baselines in terms of tracking precision. Figure 6 showcases the trajectory tracking errors, with mean errors depicted for each forgetting factor. The enhancement in disturbance prediction translates into improved tracking accuracy for DF-GP, demonstrating superior performance compared to other methodologies. Furthermore, the MHE-MPC demonstrated worse tracking performance than DF-GP method, primarily due to the lack of predictive capability. Additionally, the computational performance of the methods is shown in II. As expected, the methods based on ASGP are the fastest, as they rely on quick online updates of new data points and dynamic inducing point selection. They are followed by the GP-MPC method, which uses a dense GP implementation. The proposed DF-GP method is not far behind; it only requires additional computation for (26), which is an LP with $K = 3$, adding just 0.8 ms to the GP predictions. Thanks to the efficient C++ implementation, which utilizes multithreading and parallelization, adding multiple GP predictions did not noticeably increase the computation time. On the other hand, the MHE-MPC method had a higher computational cost, as it relies on a more expensive nonlinear optimization problem over a past horizon.

C. Real-world experiments

To enhance the validation of the simulation results, a real-world experiment is carried out at the EIVA experimental test facility, utilizing a $3 \times 5 \times 4$ meter pool [2]. The motivation to conduct real-world tests is twofold:

1) To demonstrate real-time feasibility, including computational feasibility and communication delays.
2) To showcase the effectiveness of the framework against unmodeled real-time effects, wall effects, self-induced currents by the AUV, and tether.

The experiment is conducted with a circular reference trajectory of 1-meter diameter due to space limitations. A BlueROV2 Heavy equipped with a BlueRobotics Navigator board [3], which incorporates an inertial measurement unit (IMU), compass and barometer is utilized. These sensors measure acceleration, heading, depth, and attitude estimates. Furthermore, a Wayfinder Doppler velocity log[4] is used to obtain the velocity estimate. The pose estimates are then derived through dead-reckoning based on the obtained velocities. To introduce periodic disturbances, the AUV tether is tied to one side of the pool, creating a challenging environment for the AUV. Figures 7 and 8 demonstrate that the DF-GP strategy can learn the unmodelled effects of the inaccuracies and achieve a better trajectory tracking accuracy than nominal MPC by improving the tracking error by 34.5%. This allows for learning unmodeled disturbances, such as tether effects, AUV-generated currents, thrust degradation, and other factors.

VII. Conclusion

This study introduces a learning-based MPC framework that utilizes DF-GP to effectively predict disturbances of varying magnitude and frequencies encountered in underwater robotics. This framework facilitates the integration of various GP contributions, ensuring its applicability across different scenarios without the necessity for hyperparameter retuning during operation. Furthermore, the global convergence of the proposed method is analyzed. The conducted experiments demonstrate a 59.6% improvement in disturbance estimation accuracy and a 72.9% enhancement in tracking performance across various disturbance profiles. Overall, the results validate that the proposed framework outperforms its direct competitors, achieving a 25% improvement in both disturbance estimation and tracking performance. Future studies will explore the impact of dynamic input and inducing point selection.

[2]The authors would like to thank Adis Hodzic, Ernest Wilk, Mathias Friis Spaniel, and Thomas Frank Klein from EIVA for their help with the experimental setup.

[3]https://bluerobotics.com/store/comm-control-power/control/navigator/

[4]https://www.teledynemarine.com/en-us/products/Pages/Wayfinder.aspx

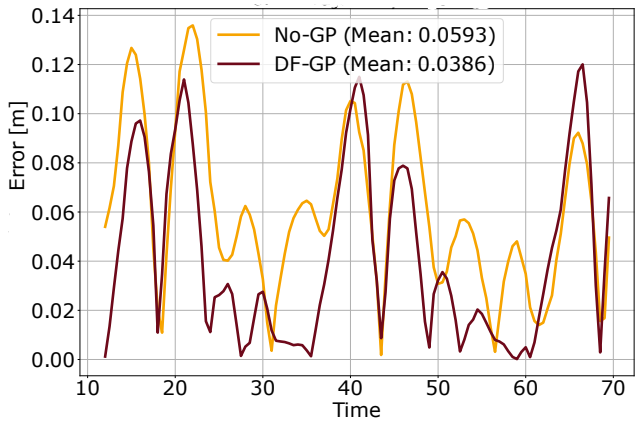

Fig. 7: Trajectory error obtained in real experiments for DF-GP and nominal MPC. Dynamic disturbance learning through DF-GP enhances tracking performance significantly.

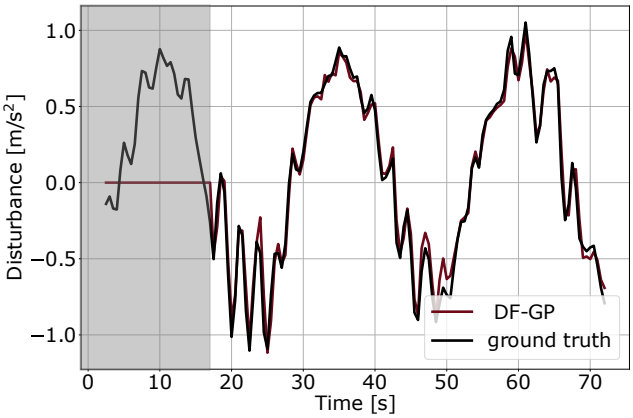

Fig. 8: Comparison of real-world test predictions against disturbance calculations (pseudo-ground truth).

References

[1] U. Eren, A. Prach, B. B. Koçer, S. V. Raković, E. Kayacan, and B. Açıkmeşe, "Model predictive control in aerospace systems: Current state and opportunities," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 7, pp. 1541–1566, 2017.

[2] A. Amer, M. Mehndiratta, J. le Fevre Sejersen, H. X. Pham, and E. Kayacan, "Visual tracking nonlinear model predictive control method for autonomous wind turbine inspection," in *2023 21st International Conference on Advanced Robotics (ICAR)*. IEEE, 2023, pp. 431–438.

[3] M. Mehndiratta and E. Kayacan, "Receding horizon control of a 3 dof helicopter using online estimation of aerodynamic parameters," *Proceedings of the institution of mechanical engineers, part G: Journal of aerospace engineering*, vol. 232, no. 8, pp. 1442–1453, 2018.

[4] M. M. Madebo, C. M. Abdissa, L. N. Lemma, and D. S. Negash, "Robust tracking control for quadrotor uav with external disturbances and uncertainties using neural network based mrac," *IEEE Access*, 2024.

[5] M. Mehndiratta and E. Kayacan, "Online learning-based receding horizon control of tilt-rotor tricopter: A cascade implementation," in *2018 Annual American Control Conference (ACC)*. IEEE, 2018, pp. 6378–6383.

[6] T. Salzmann, E. Kaufmann, J. Arrizabalaga, M. Pavone, D. Scaramuzza, and M. Ryll, "Real-time neural mpc: Deep learning model predictive control for quadrotors and agile robotic platforms," *IEEE Robotics and Automation Letters*, vol. 8, no. 4, pp. 2397–2404, 2023.

[7] M. O'Connell, G. Shi, X. Shi, K. Azizzadenesheli, A. Anandkumar, Y. Yue, and S.-J. Chung, "Neural-fly enables rapid learning for agile flight in strong winds," *Science Robotics*, vol. 7, no. 66, p. eabm6597, 2022.

[8] C. Long, X. Qin, Y. Bian, and M. Hu, "Trajectory tracking control of rovs considering external disturbances and measurement noises using eskf-based mpc," *Ocean Engineering*, vol. 241, p. 109991, 2021.

[9] K. Wang, W. Zou, R. Ma, J. Lv, H. Su, Y. Wang, and H. Ma, "Bionic underwater vehicle: A data-driven disturbance rejection control framework," *IEEE Robotics & Automation Magazine*, 2023.

[10] L. Hewing, J. Kabzan, and M. N. Zeilinger, "Cautious model predictive control using gaussian process regression," *IEEE Transactions on Control Systems Technology*, vol. 28, no. 6, pp. 2736–2743, 2019.

[11] M. Prajapat, J. Köhler, M. Turchetta, A. Krause, and M. N. Zeilinger, "Safe guaranteed exploration for non-linear systems," *arXiv preprint arXiv:2402.06562*, 2024.

[12] N. Schmid, J. Gruner, H. S. Abbas, and P. Rostalski, "A real-time gp based mpc for quadcopters with unknown disturbances," in *2022 American Control Conference (ACC)*. IEEE, 2022, pp. 2051–2056.

[13] W. Choo and E. Kayacan, "Computationally efficient data-driven mpc for agile quadrotor flight," in *2023 American Control Conference (ACC)*, 2023, pp. 2627–2632.

[14] M. Mehndiratta and E. Kayacan, "Gaussian process-based learning control of aerial robots for precise visualization of geological outcrops," in *2020 European Control Conference (ECC)*. IEEE, 2020, pp. 10–16.

[15] G. Cao, E. M.-K. Lai, and F. Alam, "Gaussian process model predictive control of an unmanned quadrotor," *Journal of Intelligent & Robotic Systems*, vol. 88, pp. 147–162, 2017.

[16] M. Maiworm, D. Limon, and R. Findeisen, "Online learning-based model predictive control with gaussian process models and stability guarantees," *International Journal of Robust and Nonlinear Control*, vol. 31, no. 18, pp. 8785–8812, 2021.

[17] G. Torrente, E. Kaufmann, P. Föhn, and D. Scaramuzza, "Data-driven mpc for quadrotors," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3769–3776, 2021.

[18] W. Choo and E. Kayacan, "Data-based mhe for agile quadrotor flight," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 4307–4314.

[19] V. Gómez-Verdejo, E. Parrado-Hernández, and M. Martínez-Ramón, "Adaptive sparse gaussian process," *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[20] D. Nguyen-Tuong, M. Seeger, and J. Peters, "Model learning with local gaussian process regression," *Advanced Robotics*, vol. 23, no. 15, pp. 2015–2034, 2009.

[21] T. I. Fossen, *Handbook of marine craft hydrodynamics and motion control*. John Wiley & Sons, 2011.

[22] C. Shen, Y. Shi, and B. Buckham, "Path-following control of an auv: A multiobjective model predictive control approach," *IEEE Transactions on Control Systems Technology*, vol. 27, no. 3, pp. 1334–1342, 2018.

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

[24] A. Amer, O. Álvarez-Tuñón, H. İ. Uğurlu, J. L. F. Sejersen, Y. Brodskiy, and E. Kayacan, "Unav-sim: A visually realistic underwater robotics simulator and synthetic data-generation framework," in *2023 21st International Conference on Advanced Robotics (ICAR)*. IEEE, 2023, pp. 570–576.

[25] M. Mehndiratta, E. Camci, and E. Kayacan, "Automated tuning of nonlinear model predictive controller by reinforcement learning," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 3016–3021.

[26] C. K. Williams and C. E. Rasmussen, *Gaussian processes for machine learning*. MIT press Cambridge, MA, 2006, vol. 2, no. 3.

[27] J. A. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "Casadi: a software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, pp. 1–36, 2019.

[28] B. Houska, H. J. Ferreau, and M. Diehl, "Acado toolkit—an open-source framework for automatic control and dynamic optimization," *Optimal Control Applications and Methods*, vol. 32, no. 3, pp. 298–312, 2011.

[29] H. J. Ferreau, C. Kirches, A. Potschka, H. G. Bock, and M. Diehl, "qpoases: A parametric active-set algorithm for quadratic programming," *Mathematical Programming Computation*, vol. 6, no. 4, pp. 327–363, 2014.

[30] A. Papadimitriou, H. Jafari, S. S. Mansouri, and G. Nikolakopoulos, "External force estimation and disturbance rejection for micro aerial vehicles," *Expert Systems with Applications*, vol. 200, p. 116883, 2022.