Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnLine 59-22 (2025) 681–686

Gaussian Process Regression for System Identification of Autonomous Surface Vessels

Sönke Bartels [*] Thomas Meurer [*]

[*] *Karlsruhe Institute of Technology, Digital Process Engineering Group, Institute for Mechanical Process Engineering and Mechanics, 76131 Karlsruhe, Germany (e-mail: {soenke.bartels, thomas.meurer}@kit.edu).*

Abstract: Autonomous surface vehicles (ASVs) have gained significant attention across a range of applications, yet a primary challenge lies in developing accurate mathematical models to describe their complex dynamical behavior. Given the partial submersion of surface vessels in water, deriving a first-principles description proves difficult. In general, data-driven approaches, particularly black-box and gray-box models, are increasingly employed to avoid the need of structural first-principle models. Among the range of supervised learning approaches, Gaussian Process Regression (GPR) models stand out due to their simplistic and nonparametric nature. This paper presents an approach to modeling the dynamics of surface vessels using GPRs. The work outlines the process of generating synthetic data, training the GPR model, and applying it to the vessel maneuvers of path-following and dynamic positioning.

Keywords: Gaussian processes, autonomous surface vehicle, autonomous surface vessel, optimal control, data based modeling

1. INTRODUCTION

Autonomous surface vehicles (ASVs) have seen growing popularity in recent years, with a wide range of applications and use cases emerging across different domains. A common challenge is the need to derive an abstract description of the dynamical behavior of the vessel. Typically, this involves the development of a mathematical description. However, finding structural mathematical models and identifying the necessary parameters and degrees of freedom is a complex and challenging task. Surface vessels are rigid bodies that are partially submerged, and as such, comprise rigid-body kinetics and the multiphase interaction of the submerged body with air and water. Achieving such a description typically relies on making assumptions and exploiting sophisticated approximations of reality. Due to these complexities, black-box and gray-box models, which minimize the need for explicitly defined structures have become of considerable interest. These models often rely on data-driven approaches. Ultimately, the goal is the development of an in-place substitution of the nominal systems model in the control loop, particularly with respect to its numerical robustness and useability.
Supervised learning and data-driven approaches, such as machine learning, offer significant advantages in model development. Gaussian Process (GP) models, in particular, eliminate the need for optimizing complex model structures due to their nonparametric nature. A key feature of GP regression models (GPR) is the use of a stationary covariance function, which differentiates them from other black-box models. One of the main advantages of this approach is the ability to assess prediction variance, which can be leveraged in model validation, guarantees or chance constraints.
GPRs have been extensively researched since decades, but are increasingly adopted as data-based modeling approaches in the context of control of autonomous systems. Basics of GPRs are presented in the work of Rasmussen and Williams (2008). Applications to dynamical systems can be found in Kocijan et al. (2004), Kocijan (2016), Hewing et al. (2020), Umlauft et al. (2018). In the domain of ASVs, GPs are employed in the context of trajectory generation for longer passages. Finding usage in the regression of the dynamics is sparse but can be found e.g. in Chen et al. (2021), Ouyang et al. (2023) and Liu et al. (2023).
This contribution describes a comprehensive blueprint of using GPRs in the domain of modeling the maneuvering dynamics of surface vessels. The focus is here on the inclusion of a priori knowledge of the vessel in the form of input dynamics and therefore the hybrid modeling. A way of generating data, training the GPR and performing different maneuvers is shown and explained.
First, in Section 2, a brief introduction into GPRs is given. It is shown how to learn a discrete-time dynamical system with previous knowledge. Afterwards, in Section 3, the model consisting of the known nominal dynamics and GRPs is used to perform path-following and dynamic positioning maneuvers. Finally, Section 4 concludes this contribution.

[⋆] Funding by the German Federal Ministry of Economic Affairs and Climate Action (BMWK), grant 03SX558B (OCUMAR), is gratefully acknowledged.

10.1016/j.ifacol.2025.11.713

2. GAUSSIAN PROCESS REGRESSION FOR SYSTEM IDENTIFICATION

A GP can be understood as defining a distribution over functions, with inference occurring directly within the space of functions, a perspective referred to as the function space view, see Rasmussen and Williams (2008). GPRs are closely connected to supervised learning, which is the problem of learning input-output mappings from empirical data. As a generalization of the Gaussian distribution, GPs are a collection of random variables characterized by a joint multivariate Gaussian distribution, (Kocijan et al., 2004). The GPR model is typically expressed as

$$f(\boldsymbol{z}) \sim \mathcal{GP}(m(\boldsymbol{z}), \mathrm{Cov}(\boldsymbol{z}, \boldsymbol{z}')), \tag{1}$$

where $\boldsymbol{z}$ is the input vector to the GP, meaning it can be completely specified by the mean and covariance

$$m(\boldsymbol{z}) = \mathbb{E}[f(\boldsymbol{z})] \tag{2a}$$
$$\mathrm{Cov}(\boldsymbol{z}, \boldsymbol{z}') = \mathbb{E}[(f(\boldsymbol{z}) - m(\boldsymbol{z}))[f(\boldsymbol{z}') - m(\boldsymbol{z}'))]. \tag{2b}$$

In practical applications, measurements usually include noise. The joint distribution over joint values with noisy observations is

$$\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{f}_* \end{bmatrix} \sim \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix} K(Z,Z) + \sigma_n^2 I & K(Z, Z_*) \\ K(Z_*, Z) & K(Z_*, Z_*) \end{bmatrix}\right) \tag{3a}$$
$$\sim \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix} K + \sigma_n^2 I & K_* \\ K_* & K_{**} \end{bmatrix}\right), \tag{3b}$$

where K denotes the covariance matrix computed from the training inputs Z, and K_* represents the covariance between training inputs Z and test inputs Z_* and K_{**} between the test inputs respectively. From (3) and the introduction of $K' = K + \sigma_n^2 I$ the predictive equations for the GPR are

$$\mu(\boldsymbol{z}) = K_*(K')^{-1}y, \tag{4a}$$
$$\Sigma(\boldsymbol{z}) = K_{**} - K_*(K')^{-1}K_{**}, \tag{4b}$$

where the vector $\alpha = (K')^{-1}y$ is based on the training data and can be computed ahead of simulation time. For training, a dataset $\mathcal{D}$ with inputs to the GP and observations of size M are necessary, which is defined as

$$\mathcal{D} = \{\boldsymbol{y} = [\boldsymbol{y}_0, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_M]^\top \in \mathbb{R}^{M \times n_y}, \tag{5a}$$
$$\boldsymbol{z} = [\boldsymbol{z}_0, \boldsymbol{z}_1, \ldots, \boldsymbol{z}_M]^\top \in \mathbb{R}^{M \times n_z}\}, \tag{5b}$$

where $\boldsymbol{y}$ are the measurements of the system and $\boldsymbol{z} = [\boldsymbol{x}^\top, \boldsymbol{u}^\top]^\top$ is the extended vector of regressors, hinting at state and input values. In this contribution the squared exponential kernel function

$$k_{i,j}(\boldsymbol{z}_i, \boldsymbol{z}_j) = \sigma_{f,a}^2 e^{\left(-\frac{1}{2}(\boldsymbol{z}_i - \boldsymbol{z}_j)^\top L_a^{-1}(\boldsymbol{z}_i - \boldsymbol{z}_j)\right)}, \tag{6}$$

is used for calculating the covariance (2), where $\sigma_{f,a}$ is the signal variance, determining the overall variance of the function outputs, and L_a is the length-scale, representing the influences of how rapidly the covariance decays with distance between two input vectors. These values describe degrees of freedom in the training of the model. This kernel is chosen based on the smoothness assumption and no further knowledge about the process is introduced. To determine values for these parameters, the *log marginal likelihood* is optimized, which quantifies how well the model fits the observed data.

2.1 Discrete-time dynamical system

The expression in (4) enables us to predict the expectation value of an unknown function using only measured data. Taking advantage of this, the aim is to model the a discrete time system in the form of

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}_\Delta(\boldsymbol{x}_k, \boldsymbol{u}_k), \quad \boldsymbol{x}_0 = \boldsymbol{x}^0, \tag{7a}$$
$$\boldsymbol{y}_k = \boldsymbol{c}^\top \boldsymbol{x}_k, \tag{7b}$$

where $\boldsymbol{f}_\Delta$ is the mapping from the current state $\boldsymbol{x}_k \in \mathcal{X} \subseteq \mathbb{R}^{n_x}$ and input $\boldsymbol{u}_k \in \mathcal{U} \subseteq \mathbb{R}^{n_u}$ to the next state. Oftentimes GPRs are exploited to model some affine model error acting on the system, but it can also be used to model part of the desired system. Since the use of GPRs in itself describe a black-box approach to regression, or system identification, no knowledge of the to be identified system is necessary. This does not mean, that it is not beneficial to introduce some knowledge. The hybrid model consisting of known dynamics, which can be first-principle, and unknown dynamics reads

$$\boldsymbol{x}_{k+1} = \tilde{\boldsymbol{f}}_\Delta(\boldsymbol{x}_k, \boldsymbol{\tau}_k) + \mathcal{G}(\boldsymbol{x}_k, \boldsymbol{\tau}_k) \tag{8a}$$
$$= \tilde{\boldsymbol{f}}_\Delta(\boldsymbol{x}_k, \boldsymbol{\tau}_k) + B_d(\boldsymbol{g}(\boldsymbol{x}_k, \boldsymbol{\tau}_k) + \boldsymbol{w}_k), \tag{8b}$$

where $\tilde{\boldsymbol{f}}_\Delta$ is the discretized known part part of our system dynamics, $B_d \in \mathbb{R}^{n_x \times n_g}$ is the mapping between the subspaces, $\boldsymbol{g} \in \mathbb{R}^{n_g}$ are the unknown dynamics and $\boldsymbol{w}_k$ is the process noise. Typically, only a subset of states is modeled by GPs, meaning $n_g < n_x$. The expression for the observations for training the GPs, utilized to form the dataset $\mathcal{D}$ in (5), necessary for training in the hybrid approach reads

$$\boldsymbol{y}_k = B_d^\dagger(\boldsymbol{x}_{k+1} - \tilde{\boldsymbol{f}}_\Delta(\boldsymbol{x}_k, \boldsymbol{\tau}_k)) = \boldsymbol{g}(\boldsymbol{x}_k, \boldsymbol{\tau}_k) + \boldsymbol{w}_k. \tag{9}$$

It is important to note that, if the unknown dynamics are not scalar, every dimension is learned independently and concatenate them together afterwards, so that $\boldsymbol{g}(\boldsymbol{x}_k, \boldsymbol{u}_k) = [g_0(\boldsymbol{x}_k, \boldsymbol{u}_k), g_1(\boldsymbol{x}_k, \boldsymbol{u}_k), \ \ldots \ , g_{n_g}(\boldsymbol{x}_k, \boldsymbol{u}_k)]^\top \in \mathbb{R}^{n_g}$.

GPR Training Training the GPR with measured data is a straightforward process. However, the dimensions of the matrix $K \in \mathbb{R}^{M \times M}$ can pose a challenge when the number of data points and observations becomes high. As the dataset grows larger, calculating the inverse of K becomes increasingly computationally expensive. Addressing this issue, sparse matrix approximations can be utilized. In this context, the *fully-independent training conditional* (FITC) is assumed, which posts that the training set observations are independent of each other. This assumption allows us to introduce *inducing points*, see Snelson and Ghahramani (2005), meaning to approximate the GPR on a state-space grid. For now a uniformly spaced grid on all regressors with M_i data points is assumed. This way the number of data points is reduced from M to $M_r = \sum_{i=0}^{n_g} M_i$.

Uncertainty Propagation Modeling the dynamical system using GPRs, future predicted states are stochastic distributions. Following the common approach, i.e.,

$$\boldsymbol{\mu}_i = \boldsymbol{f}(\mu_x, \mu_u) + B_d \mu_d(\mu_x, \mu_i) \tag{10a}$$
$$\Sigma_i = \tilde{A}_i \Sigma \tilde{A}_i^\top + B_d(\Sigma(\mu_x, \mu_i) + \Sigma_w)B_d^\top, \tag{10b}$$

where $\tilde{A}_i = \nabla(\boldsymbol{f}(x, \mu_u + K_i x) + B_d \mu_d(x, \mu_u + K_i x))|_{\mu_x}$, while $\Sigma_u = K_i \Sigma K_i$, allows for predicting the mean and covariance values. This enables us to propagate not only the mean value of the GP, but also the covariance in time.

2.2 Derivation of the nominal system dynamics

The modeling of the system dynamics of an ASV is typically guided by requirements, while the main objective is always to model the system as simple as possible, but as complex as necessary. Task dependent, employing a simplified yaw dynamics model, such as a Nomoto model, or a more comprehensive model, such as a full three degrees of freedom (3-DOF) model is possible. The 3-DOF model of a surface vessel can be determined using either Newtonian or Lagrangian mechanics and reads

$$\boldsymbol{\eta} = R(\psi)\boldsymbol{\nu} \tag{11a}$$
$$M\dot{\boldsymbol{\nu}} = \boldsymbol{\tau}(\boldsymbol{u}) - C(\boldsymbol{\nu}) - D(\boldsymbol{\nu})\boldsymbol{\nu}, \tag{11b}$$

where $C(\boldsymbol{\nu})$ is the Coriolis matrix, $D(\boldsymbol{\nu})$ the damping matrix of linear and non-linear damping terms and $\boldsymbol{\tau}$ the control input. The state-space can be separated into pose $\boldsymbol{\eta} = [x, y, \psi]^\top$ and the body-frame velocities $\boldsymbol{\nu} = [u, v, r]^\top$, which results in $\boldsymbol{x} = [\boldsymbol{\eta}^\top, \boldsymbol{\nu}^\top]^\top$. Here, $\boldsymbol{\tau} = [X, Y, N]^\top$ describes the generalized control input. The configuration of the 3-DOF vessel is illustrated in Fig. 1. For a more detailed derivation refer to, e.g., Fossen (2002). The coordinate transformation matrix is given by

$$R(\psi) = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{12}$$

whereas the parameterization of the inertia matrix can be described by

$$M = \begin{bmatrix} m_{11} & 0 & 0 \\ 0 & m_{22} & m_{23} \\ 0 & m_{32} & m_{33} \end{bmatrix} = \begin{bmatrix} m - X_{\dot{u}} & 0 & 0 \\ 0 & m - Y_{\dot{v}} & -Y_{\dot{r}} \\ 0 & -N_{\dot{v}} & I_{zz} - N_{\dot{r}} \end{bmatrix}. \tag{13}$$

Coriolis and damping matrices are given by

$$C(\boldsymbol{\nu}) = C(\boldsymbol{\nu})^\top = \begin{bmatrix} 0 & 0 & c_{13} \\ 0 & 0 & c_{23} \\ -c_{13} & -c_{23} & 0 \end{bmatrix}, \tag{14}$$

where

$$c_{13} = -m_{22}v - \frac{m_{23} + m_{32}}{2}r, \quad c_{23} = m_{11}u,$$

and

$$D(\boldsymbol{\nu}) = \begin{bmatrix} X_u + X_{|u|u}|u| & 0 & 0 \\ 0 & Y_v + Y_{|v|v}|v| & Y_r \\ 0 & N_v & N_r + N_{|r|r}|r| \end{bmatrix}. \tag{15}$$

The structural model needs to be parameterized. For simulation tasks in this paper the parameters for Cybership 2 are used, see, e.g., Skjetne et al. (2004a).

Based on the structured and parameterized formulation in (11), the components of the vessel model that can be reasonably assumed to be known can be systematically identified. First, the separation of kinematics and kinetics, i.e. $\boldsymbol{\eta}$ and $\boldsymbol{\nu}$ is exploited. These subsystems are connected by an integration, which would make it redundant to learn the states of $\boldsymbol{\eta}$. Instead, it appears reasonable to further decompose the model into the rigid-body and the affine input system, as in

$$\dot{\boldsymbol{\nu}} = \underbrace{M^{-1}\boldsymbol{\tau}}_{\text{input-part}} - \underbrace{M^{-1}(C(\boldsymbol{\nu}) + D(\boldsymbol{\nu})\boldsymbol{\nu})}_{\text{to be learned}}. \tag{16}$$

The modeling of the affine input system is part of the design of the propulsion system. The actual values of the generalized input $\boldsymbol{\tau}$ depend on the actual actor configuration.

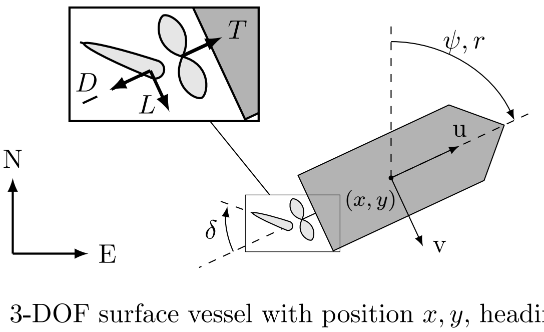


Fig. 1. 3-DOF surface vessel with position x, y, heading ψ and velocities in the body-frame u, v and r.

Actuator dynamics Given the use of a simulation environment, a simplified actor configuration is adopted, consisting of a propeller-rudder system, as depicted in Fig. 1. The derivation is taken from Skjetne et al. (2004b). To model the thrust of the propeller, it is assumed to be fixed in place and aligned with the vessel. The thrust T of the propeller is given by

$$T(\omega, \boldsymbol{x}) = \begin{cases} c^+_{\mathrm{ww}}|\omega|\omega - c^+_{\mathrm{wu}}|\omega|u_{\mathrm{p}}, & \text{if } \omega \geq 0 \\ c^-_{\mathrm{ww}}|\omega|\omega - c^-_{\mathrm{wu}}|\omega|u_{\mathrm{p}}, & \text{if } \omega < 0, \end{cases} \tag{17}$$

where $c^+_{\mathrm{ww}}, c^+_{\mathrm{wu}}, c^-_{\mathrm{ww}}, c^-_{\mathrm{wu}}$ are positive coefficients, ω is the propeller revolutions and u_{p} is the relative velocity at the propeller, which can be assumed close to the surge speed of the vessel, i.e. $u_p \approx u$. The resulting force vector results is given by $\boldsymbol{f}_{\mathrm{prop}}(\omega) = [T, 0]^\top$.

Modeling the rudder, it is necessary to derive expressions for the drag and lift forces, such that the rudder force vector is given by $\boldsymbol{f}_{\mathrm{rud}}(\delta) = [-D, L]^\top$. Elementary for the lift forces of the rudder is the relative fluid velocity at the hydrodynamic foil. If the rudder is standalone, the velocity at the rudder can be assumed to be the general surge velocity. Whereas, if the rudder is mounted sufficiently close to a propeller the fluid velocity at the rudder is increased by the propeller slipstream so that

$$u_r = u + k_u \left(\sqrt{\frac{8}{\pi \rho d^2} T + u^2} - u \right), \tag{18}$$

where k_u is a positive gain coefficient. The relative velocity u_r can be used to calculate the lift force L

$$L = \begin{cases} \left(L^+_\delta \delta - L^+_{|\delta|\delta} |\delta| \delta \right) |u_r| u_r, & u_r \geq 0 \\ \left(L^-_\delta \delta - L^-_{|\delta|\delta} |\delta| \delta \right) |u_r| u_r, & u_r < 0, \end{cases} \tag{19}$$

where δ is the rudder angle as depicted in Fig. 1. Combining the thrust of the propeller with the lift of the rudder, the resulting force vector for the rudder is

$$\boldsymbol{\tau}(\boldsymbol{u}) = B_T \left(\boldsymbol{f}_{\mathrm{prop}}(\omega) + \boldsymbol{f}_{\mathrm{rud}}(\delta) \right) = B_T \begin{bmatrix} T - D \\ L \end{bmatrix}, \tag{20}$$

where $\boldsymbol{u} = [\omega, \delta]^\top$ and B_T is the thrust configuration matrix, describing the mounting position of the actors, which are in this case assumed to centered at the same point. It is also possible to model the propulsion dynamics or the static mapping of actual control values via a GP.

In contrast to (7), which is a discrete-time model, the model in (11) is continuous time. To connect these models, as in (8), a discretization the model according to the Δt of the GPR model

$$\tilde{\boldsymbol{x}}_{k+1} = \tilde{\boldsymbol{f}}_\Delta(\boldsymbol{x}_k, \boldsymbol{u}_k), \tag{21}$$

is necessary, which can be achieved by any discretization technique.

3. SYNTHETIC DATA GENERATION, TRAINING AND SIMULATION

Now, the focus is on modeling the system dynamics using GPRs, integrated in a discrete-time model the discrete-time model (7). For the necessary training data, time-series of the states are needed. In practice, this would involve performing identification maneuvers. To simulate this process, a parameterized 3-DOF model is taken to generate time-series data. The identification maneuver is desired to cover as much of the vessel's state space as possible. For this reason, the vessel's objective is to perform sharp turns in all directions. Since no information about the model is available, despite the vessel's propulsion system, a *model-free* path-following method needs to be derived.

For this purpose, a guidance system with PI control, as presented in Fig. 2, is utilized. The guidance is based on combined speed and Line-Of-Sight (LOS) reference generation. This means, that the PI controller receives a reference speed over ground (SOG), as well as a heading reference. The SOG is set constant, but the heading reference depends on a sequence of waypoints and the current position of the vessel. The LOS heading reference computes as

$$\psi_{i,\mathrm{ref}}(\boldsymbol{p}) = \psi_i + \psi_{\mathrm{LOS}}(\boldsymbol{p}) \tag{22a}$$

$$= \psi_i + \arctan\left(-\frac{e_{xte}(\mathrm{WP}_i, \mathrm{WP}_{i+1}, \boldsymbol{p})}{\Delta_{\mathrm{LOS}}}\right) \tag{22b}$$

where e_{xte} is the cross-track-error and Δ_{LOS} is the lookahead distance, which as seen in Fig. 3, is the distance the vessel looks ahead along the track. The scenario can be executed and the data now represents the desired time series. The resulting time series is 300 s long and sampled with a discretization time step of $dt = 0.1$ s, which results in $N = 3000$ data points and observations. As presented in Fig. 2, the control loop does not in include an observer so that the simulated state vectors is considered as our observations. State space coverage is described in Tab. 1, where also the hyper-parameters of the GPR are given.

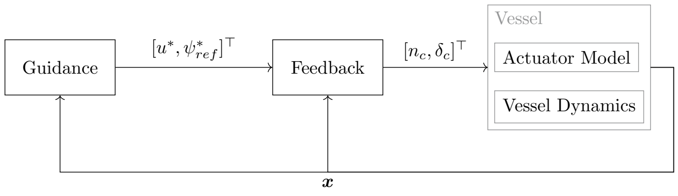


Fig. 2. Closed-loop data generation with LOS-guidance and PI speed and heading control.

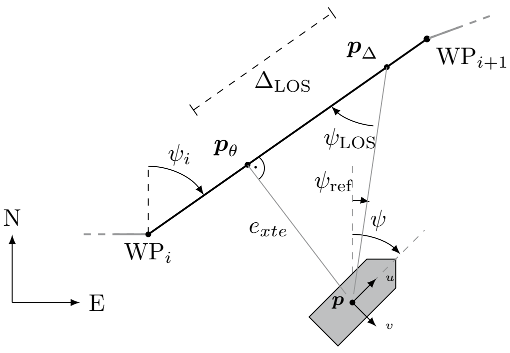


Fig. 3. Heading reference generation setting with a linear waypoint segment.

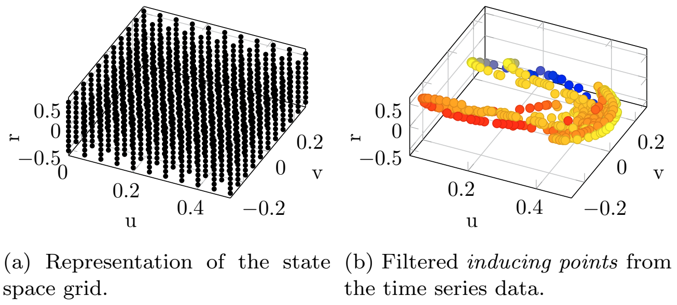
Figure 4 shows the applied state space grid, as well as the chose indcung points as mentionend in Section 2.1.1. Training and hyper-parameter optimization is performed by MATLAB's *fitgpr* as summarized in (3).

Path-following Performing the same maneuver with the hybrid model, i.e., following the same waypoints with the same guidance system and controller as in the data generation phase results in the data shown in Figure 5, 6 and 7. It directly becomes clear that the hybrid model is performing the task qualitatively similar to the nominal model. After half the scenario, there appears to be a small shift in time, as the hybrid model can perform the task slightly faster. As the model is integrated over time, the slightest difference in the dynamics accumulates and leads to a significant deviations at the end. This result is anticipated, since the in the path following task, the most dominant dynamics is the yaw of the vessel. This is one reason why most of the times, for path following, simplified model structures (Nomoto models) can be utilized. These are easier to identify since they only depend on two parameters describing the yaw-dynamics.

Dynamic Positioning In another simulation case the hybrid model is used for a dynamic positioning task. The

Table 1. State space coverage of the data.

state	min	max	σ_f	L_a
u in m s^{-1}	-0.0888	0.3223	0.0117	0.5977
v in m s^{-1}	-0.2125	0.2041	0.0144	0.6047
r in rad s^{-1}	-0.4624	0.4699	0.0459	0.6004



(a) Representation of the state space grid. (b) Filtered *inducing points* from the time series data.

Fig. 4. Data samples in the vessel's state space.

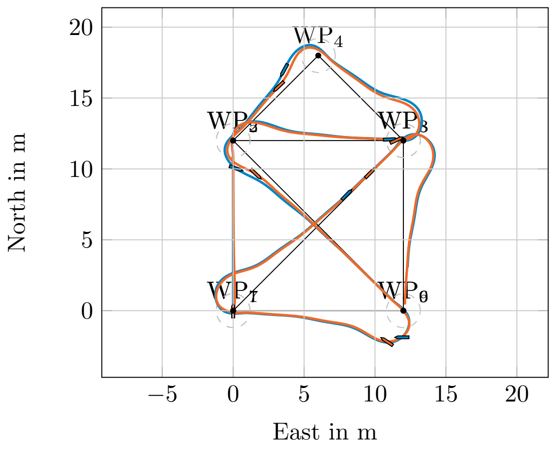


Fig. 5. Top-down view of the path-following maneuver, blue refers to the time series used for training, orange refers to the path taken by the GPR model.

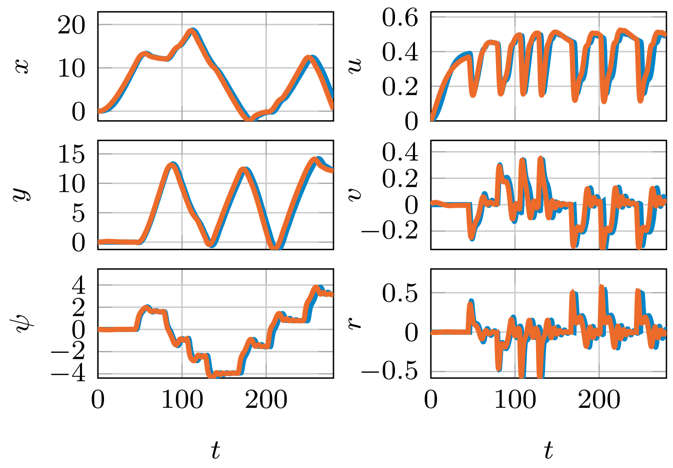
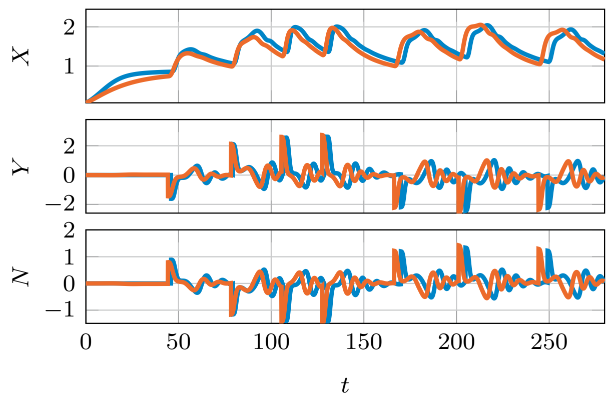
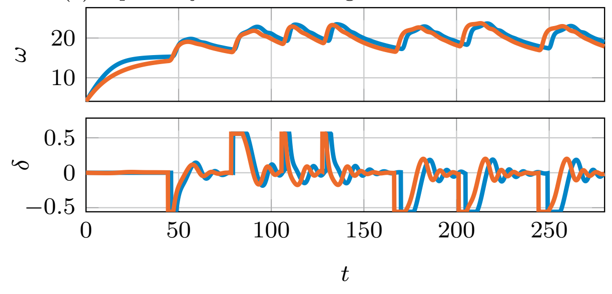


Fig. 6. State trajectories for the path-following task, blue refers to the time series used for training, orange refers to the path taken by the GPR model.



(a) Input trajectories of the generalized forces $\boldsymbol{\tau}$.



(b) Input trajectories of the actual control values $\boldsymbol{u}$.

Fig. 7. Input trajectories for the path-following task, blue refers to the time series used for training, orange refers to the path taken by the GPR model.

goal is to translate between two two defined poses $\boldsymbol{\eta}_0, \boldsymbol{\eta}_f$ in a given time T. The path directly connecting the positions is blocked by two obstacles. To perform this maneuver an optimal control problem (OCP) is formulated.

$$\min_{\boldsymbol{w}} l = \int_0^T ||\boldsymbol{\eta}_k - \boldsymbol{\eta}_f||_2 \, dt \tag{23a}$$

$$\text{s.t.} \tag{23b}$$

$$\boldsymbol{x}_{k+1} = \tilde{\boldsymbol{f}}_\Delta(\boldsymbol{x}_k, \boldsymbol{\tau}_k) + B_d(\boldsymbol{g}(\boldsymbol{x}_k, \boldsymbol{\tau}_k) + \boldsymbol{w}_k) \tag{23c}$$

$$\boldsymbol{u}_k \in \mathcal{U} \tag{23d}$$

$$\emptyset = \mathcal{V}(\boldsymbol{x}_k) \cap \{\mathcal{O}_1, \mathcal{O}_2\} \tag{23e}$$

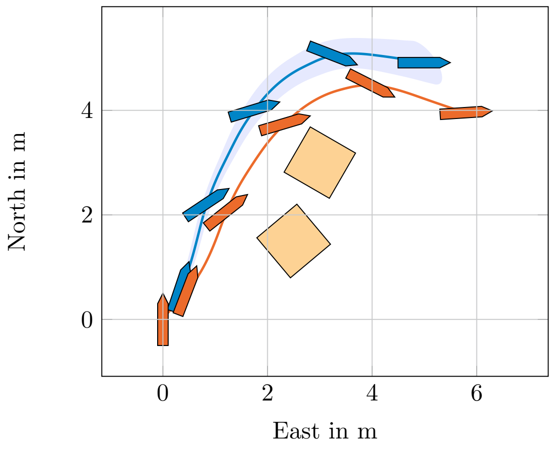


Fig. 8. Top-down view of the environment for the dynamic positioning task, blue refers to the time series used for training, orange refers to the path taken by the GPR model.

where $\boldsymbol{w} = [\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\mu}, \boldsymbol{\lambda}]$. The obstacle avoidance constraints are given by

$$d_{\text{safe}} + \frac{(\boldsymbol{\lambda}_m)^\top \boldsymbol{d}(\boldsymbol{x}) + (\boldsymbol{\mu}_m)^\top \boldsymbol{b}_m}{\|(A_m)^\top \boldsymbol{\mu}_m\|_2} \leq \epsilon_m, \tag{24a}$$

$$C^\top(\boldsymbol{x})\boldsymbol{\lambda}_m + (A_m)^\top \boldsymbol{\mu}_m = \boldsymbol{0}, \tag{24b}$$

$$\boldsymbol{\mu}_m, \boldsymbol{\lambda}_m \geq \boldsymbol{0}, \tag{24c}$$

where $C(\boldsymbol{x}) = \tilde{C}R^\top(\psi)$ and $\boldsymbol{d}(\boldsymbol{x}) = \tilde{\boldsymbol{d}} + \tilde{C}R^\top(\psi)\boldsymbol{p}_n^b$ and $\tilde{C}, \tilde{\boldsymbol{d}}$ depend on the shape of the controlled vessel. The (dual) decision variables $\boldsymbol{\lambda}_m$ and $\boldsymbol{\mu}_m$ therefore need to be included in the OCP for all $m = 1, \ldots, K$. See Helling et al. (2021) for more details on the dual approach. The results are shown in Figure 8, 9 and 10. The OCP is solved using Ipopt (Wächter and Biegler, 2006) and Casadi (Andersson et al., 2019) is used for effectively generating gradients. It is important to note that in this case the actor configuration is left out and the the generalized forces are seen as the system's input. After solving the OCP, the comparison of trajectories can be seen in Fig. 8, 10 and 9. In these figures, the solution of the OCP is compared to the feed-forward simulation of the ground truth model. This way a statement can be made over the suitability of using the hybrid model in the planning process. In this case a larger error as in the path-following case is visible. This is due to the fact the this is a high dynamic maneuver. Since the rather simple time series used for training is not representing all possible states of the vessel, the model is not as accurate in these situations.

4. CONCLUSIONS

This contribution shows that a hybrid sparse GPR is an effective tool for the modeling of the dynamics of surface vessel. The applications reveal that a compact dataset, such as one obtained from identification maneuvers or trial runs, is enough to provide a reasonable approximation of the underlying dynamics. The proposed hybrid system successfully performs path following tasks based on a 3-DOF model. Moreover, satisfactory results were achieved even for a dynamic positioning maneuver, without extending the dataset. In the presented approach of hybrid modeling, the input mapping is assumed to be known. This requires

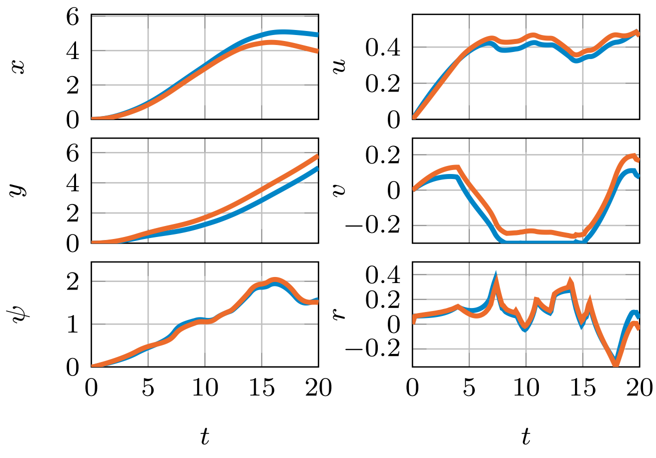


Fig. 9. State trajectories for the dynamic positioning task, blue refers to the time series used for training, orange refers to the path taken by the GPR model.

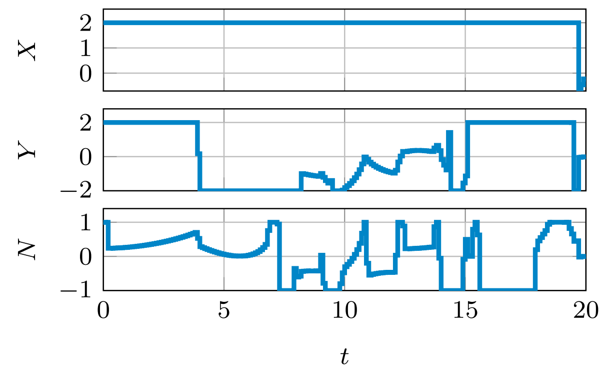


Fig. 10. Input trajectories for the dynamic positioning task, blue refers to the time series used for training, orange refers to the path taken by the GPR model.

the inverse of the mass matrix to be parameterized, which, in turn, requires the identification of the added mass terms. Theoretically, these can be guessed based on the shape of the vessel and literature values. This would also raise the question to why not assume the Coriolis matrix to be known, since it just depends on values of the mass matrix. It is emphasized, that the approach was carried out in simulation only. The next logical step is to compare the simulation results with real world data. But even before that, more optimizations can be done. This includes a more detailed data selection, to reduce the size of the data set, and a more detailed like at hyperparameter tuning and underlying kernel functions.

ACKNOWLEDGEMENTS

The author gratefully acknowledges Erik Widmer for his excellent work on his Master's thesis, which was carried out alongside in collaboration.

REFERENCES

Andersson, J.A.E., Gillis, J., Horn, G., Rawlings, J.B., and Diehl, M. (2019). CasADi: A software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 11(1), 1–36. doi:10.1007/s12532-018-0139-4.

Chen, G., Wang, W., and Xue, Y. (2021). Identification of Ship Dynamics Model Based on Sparse Gaussian Process Regression with Similarity. *Symmetry*, 13(10), 1956. doi:10.3390/sym13101956.

Fossen, T.I. (2002). *Marine Control Systems: Guidance, Navigation and Control of Ships, Rigs and Underwater Vehicles*. Marine Cybernetics, Trondheim, 3. printing edition.

Helling, S., Roduner, C., and Meurer, T. (2021). On the Dual Implementation of Collision-Avoidance Constraints in Path-Following MPC for Underactuated Surface Vessels. In *2021 American Control Conference (ACC)*, 3366–3371. IEEE, New Orleans, LA, USA. doi:10.23919/acc50511.2021.9482890.

Hewing, L., Kabzan, J., and Zeilinger, M.N. (2020). Cautious Model Predictive Control Using Gaussian Process Regression. *IEEE Transactions on Control Systems Technology*, 28(6), 2736–2743. doi:10.1109/TCST.2019.2949757.

Kocijan, J., Murray-Smith, R., Rasmussen, C., and Girard, A. (2004). Gaussian process model based predictive control. In *Proceedings of the 2004 American Control Conference*, 2214–2219 vol.3. IEEE, Boston, MA, USA. doi:10.23919/ACC.2004.1383790.

Kocijan, J. (2016). *Modelling and Control of Dynamic Systems Using Gaussian Process Models*. Advances in Industrial Control. Springer International Publishing, Cham. doi:10.1007/978-3-319-21021-6.

Liu, S.Y., Ouyang, Z.L., Chen, G., Zhou, X., and Zou, Z.J. (2023). Black-box modeling of ship maneuvering motion based on Gaussian process regression with wavelet threshold denoising. *Ocean Engineering*, 271, 113765. doi:10.1016/j.oceaneng.2023.113765.

Ouyang, Z.L., Zou, Z.J., and Zou, L. (2023). Nonparametric Modeling and Control of Ship Steering Motion Based on Local Gaussian Process Regression. *Journal of Marine Science and Engineering*, 11(11), 2161. doi:10.3390/jmse11112161.

Rasmussen, C.E. and Williams, C.K.I. (2008). *Gaussian Processes for Machine Learning*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, Mass., 3. print edition. doi:10.7551/mitpress/3206.001.0001.

Skjetne, R., Smogeli, Ø., and Fossen, T.I. (2004a). Modeling, identification, and adaptive maneuvering of CyberShip II: A complete design with experiments. *IFAC Proceedings Volumes*, 37(10), 203–208. doi:10.1016/S1474-6670(17)31732-9.

Skjetne, R., Smogeli, Ø.N., and Fossen, T.I. (2004b). A Nonlinear Ship Manoeuvering Model: Identification and adaptive control with experiments for a model ship. *Modeling, Identification and Control: A Norwegian Research Bulletin*, 25(1). doi:10.4173/mic.2004.1.1.

Snelson, E. and Ghahramani, Z. (2005). Sparse gaussian processes using pseudo-inputs. 18.

Umlauft, J., Beckers, T., and Hirche, S. (2018). Scenario-based Optimal Control for Gaussian Process State Space Models. In *2018 European Control Conference (ECC)*, 1386–1392. IEEE, Limassol. doi:10.23919/ECC.2018.8550458.

Wächter, A. and Biegler, L. (2006). On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming.