IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 15 August 2024, accepted 16 September 2024, date of publication 20 September 2024, date of current version 4 October 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3464644

TOPICAL REVIEW

Machine Learning for Modeling Underwater Vehicle Dynamics: Overview and Insights

XAN MACATANGAY[1], SARGON A. GABRIEL[1], REZA HOSEINNEZHAD[1], ANTHONY FOWLER[2], (Member, IEEE), AND ALIREZA BAB-HADIASHAR[1], (Senior Member, IEEE)

[1]School of Engineering, RMIT University, Melbourne, VIC 3000, Australia
[2]Platforms Division, Defence Science and Technology Group, Fishermans Bend, VIC 3207, Australia

Corresponding author: Xan Macatangay (xan.macatangay@rmit.edu.au)

The work of Xan Macatangay was supported by the Royal Melbourne Institute of Technology (RMIT) University through scholarships, including an Australian Government Research Training Program (RTP) scholarship.

ABSTRACT Accurate modeling of underwater vehicle dynamics is an essential component of various solutions designed to address a range of challenges involved in both the vehicle's design and operation. Such models are usually parametric, including dynamic equations that simulate the vehicle's response to various controls and environment conditions. They can be used to determine the vehicle's capabilities, estimate the vehicle's state in the absence of external communications, or to derive control signals to produce desired state responses. While a range of explicitly derived models have been commonly used in various applications, modeling the complex nonlinear dynamics using machine learning has recently attracted considerable interest. This topical review focuses on the integration of machine learning in underwater vehicle modeling, and covers two categories: artificial neural networks and non-parametric regression models. The first category includes recurrent neural networks and physics-informed neural network. They are trained to estimate model parameters, forces and moments from damping and disturbances, or to completely replace the dynamic model by outputting the expected state responses. The second category of the reviewed models covers support vector machines and Gaussian process models. These are non-parametric dynamic models and their training requirements are generally lower than ANN-based models. An overview of the theory behind each model is presented, along with examples of specific applications. The capabilities of each machine learning method are compared, and the challenges of their implementation for underwater vehicle dynamic modeling are discussed.

INDEX TERMS Underwater vehicle, dynamics, machine learning, artificial neural network, Lagrangian mechanics, support vector machine, Gaussian process.

I. INTRODUCTION

There are many challenges in operating underwater vehicles, which arise from the complexity and uncertainty in these systems and their environment. These include the highly nonlinear system dynamics, unknown external disturbances, limited wireless communication [1], and underwater navigation difficulties [2]. Underwater vehicle dynamic models are used to address many of these challenges, in both the design and operation processes. Accurate dynamic models allow for the capabilities of potential vehicle designs to be evaluated in a timely manner compared to physical trials and computational fluid dynamics (CFD) simulations. Using these models, the real-time estimation of the vehicle's motion also enables localization in the absence of external communications and the compensation of undesired nonlinear dynamics in uncertain ocean environments for adaptive control.

A range of parametric, coefficient-based dynamic models for underwater vehicles have been developed over the years, differentiated by the velocity relationships considered in their dynamic equations [3]. The choice of which model

The associate editor coordinating the review of this manuscript and approving it for publication was Ravinesh C. Deo.

to use depends on the application's required accuracy, maximum processing time, and available computation power. The specific equations used for an application may also be adapted to account for variations such as the available actuators and the relevant degrees of freedom.

Recent advancements in the field of supervised machine learning (ML) have demonstrated promising results in regression and approximation for complex nonlinear dynamic models. Specifically, these ML algorithms have been applied in two ways. The first is a form of black box non-parametric modeling, where the ML algorithm is trained to estimate the system dynamics response directly from the system's present and past states. The second is a form of regression for estimating parametric model coefficients. Both applications have trade-offs regarding accuracy limitations and implementation challenges. Recent implementations of ML algorithms for underwater vehicle modeling span over a range of algorithms including various neural network architectures for modeling hydrodynamics [4], [5] and model uncertainties [6], [7], support vector regression [8], [9], [10], and Gaussian process regression [11], [12], [13]. Documentation of these developments include a recent survey [14] that explores the history and developments of estimating hydrodynamic coefficients using traditional and ML-based methods. Wehbe et al. [15] have also compared four ML algorithms and the traditional least-squares method for estimating the decoupled hydrodynamic damping function.

In this topical review, we elaborate on the implementation of different machine learning algorithms for developing dynamic models of underwater vehicles and similar maritime systems, analyzing their demonstrated and potential applicability for different underwater vehicle modeling tasks. The benefits and limitations of both non-parametric and parametric modeling methods are explored, and a focus on physics-informed neural networks and non-parametric regression methods differentiates this review from existing literature for underwater vehicles. For each modeling method, the theoretical background and mathematical implementation are reviewed, followed by examples of its implementation for underwater vehicle dynamic model approximation. We conclude the review with a final comparison and discussion on the suitability of the reviewed ML algorithms for different underwater vehicle modeling applications.

The rest of this review is structured as follows. Section II provides the background information on the parametric models used for underwater vehicles. Section III describes the general implementation methods of ML algorithms for producing both parametric and non-parametric underwater vehicle models, highlighting their comparative advantages and limitations. Section IV introduces artificial neural networks (ANNs) and its variations. The development of physics-informed neural networks and their properties are explored in Section V. The literature on support vector regression and Gaussian process regression is introduced in Section VI, highlighting works that compare different regression methods. A discussion on the application of machine learning for modeling underwater vehicle dynamics is presented in Section VII, followed by the conclusion in Section VIII.

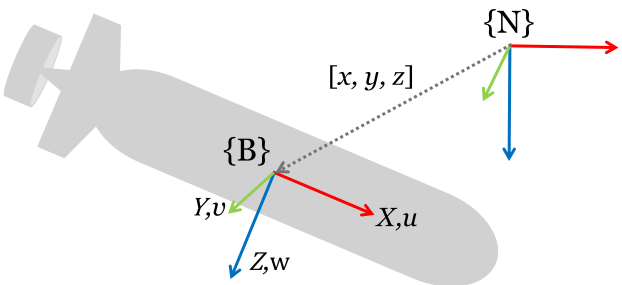


FIGURE 1. An illustration of the body-fixed frame $\{B\}$ relative to the reference frame $\{N\}$, displaying the position and directions for the control forces and linear velocities.

II. CLASSICAL PARAMETRIC MODELS

Various parametric dynamic equations that approximate the complex nonlinear dynamics of an underwater vehicle are available for time efficient simulation and control applications. Considering the vehicle as a rigid body with all six Degrees of Freedom (DoF), these models are based on Newton's second law and express the vehicle's dynamics using forces and moments relative to the body-fixed frame $\{B\}$ attached to the vehicle [16]. The forces and moments considered in these models include the Coriolis forces/moments, hydrodynamic damping, combined gravity and buoyancy, actuator forces/moments, and external disturbances. By expressing the dynamics in $\{B\}$, these forces and moments can be estimated using functions relating to the vehicle's states. For example, Fossen's vectorial model [17], [18] expresses an underwater vehicle's dynamics and kinematics as:

$$M\dot{\nu} + C(\nu)\nu + D(\nu)\nu + g(\eta) = \tau + J(\eta)^{-1}b \quad (1)$$

$$\dot{\eta} = J(\eta)\nu, \quad (2)$$

where:

- $\nu = [u\ v\ w\ p\ q\ r]^\top$ is the vehicle's linear and angular velocity components in the body-fixed frame $\{B\}$.
- $\tau = [X\ Y\ Z\ K\ M\ N]^\top$ denotes the total control forces and moments produced by the vehicle's actuators relative to $\{B\}$, which can be calculated from models of the individual actuators.
- $\eta = [x\ y\ z\ \phi\ \theta\ \psi]^\top$ denotes the global location and pose vector of the vehicle, expressed as the position and orientation of frame $\{B\}$ relative to the north-east-down reference frame $\{N\}$, as shown in Figure. 1

The rotation matrix $J(\eta) \in \mathbb{R}^{6\times6}$ transforms state vectors between the $\{B\}$ and $\{N\}$ frames, and the function $g(\eta) \in \mathbb{R}^6$ outputs the combined gravity and buoyancy forces/moments vector. The accuracy of the model depends on how accurate

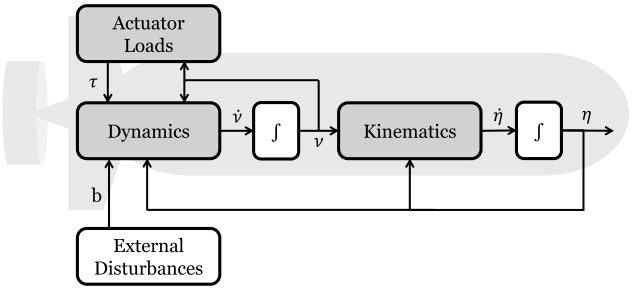


FIGURE 2. Block diagram of a simple 6-DoF dynamic model based on equations (1) and (2), featuring the actuator loads, external disturbances, 6-DoF equations for dynamics and kinematics, and time integrators.

the external disturbance vector $b \in \mathbb{R}^6$ is estimated, and on the accuracy of the parameters for total (rigid body and added) inertia matrix $M \in \mathbb{R}^{6\times6}$, total (rigid body and added) Coriolis matrix $C(\nu) \in \mathbb{R}^{6\times6}$, and the hydrodynamic damping matrix $D(\nu) \in \mathbb{R}^{6\times6}$. The off-diagonal terms in these matrices account for the coupling between different DoFs in the model, and using constant values in C and D will produce a linearized model.

This dynamic model can be conceptualized with the simple block diagram shown in Figure 2. Such models can be used to simulate the vehicle's states in response to control signals, or inverted to derive the control signals required to achieve a desired vehicle state.

Vectorial models such as equation (1) express the 6-DoF dynamics of the vehicle in a compact form, but at the complication of hiding the individual terms (and their coefficients) that affect the individual DoFs. Models such as the ones proposed by Gertler and Hagen [20] and Feldman [21] express the dynamic equations for an underwater vehicle in an expanded form, with equations for each DoF. As an example, in Gertler and Hagen's model, the equation for the surge DoF (along the x-axis of $\{B\}$) is given by:

$$\begin{aligned}
&m(\dot{u} - vr + wq - x_G(q^2 + r^2) + y_G(pq - \dot{r}) + z_G(pr + \dot{q})) \\
&= \frac{1}{2}\rho L^4(X'_{qq}q^2 + X'_{rr}r^2 + X'_{rp}rp) \\
&\quad + \frac{1}{2}\rho L^3(X'_{\dot{u}}\dot{u} + X'_{vr}vr + X'_{wq}wq) \\
&\quad + \frac{1}{2}\rho L^2(X'_{uu}u^2 + X'_{vv}v^2 + X'_{ww}w^2) \\
&\quad + \frac{1}{2}\rho L^2(X'_{\delta R\delta R}u^2\delta_r^2 + X'_{\delta s\delta s}u^2\delta_s^2 + X'_{\delta B\delta B}u^2\delta_b^2) \\
&\quad + \frac{1}{2}\rho L^2(a_i u^2 + b_i u u_c + c_i u_c^2) \\
&\quad - (W - B)sin(\theta) \\
&\quad + \frac{1}{2}\rho L^2(X'_{\delta R\delta R\eta}u^2\delta_r^2 + X'_{\delta s\delta s\eta}u^2\delta_s^2)(\eta_v - 1) \\
&\quad + \frac{1}{2}\rho L^2(X'_{vv\eta}v^2 + X'_{ww\eta}w^2)(\eta_v - 1),
\end{aligned} \tag{3}$$

where the X'_i terms are non-dimensional hydrodynamic coefficients with subscripts $i \in \{qq\ rr\ rq\ \ldots\ vv\eta\ ww\eta\}$ specifying the state variables (velocity vector components and actuator states $\delta_r, \delta_s, \delta_b, \eta_v$) to which the coefficient is relevant. $(W - B)$ is the combined gravity and buoyancy force magnitude, $\delta_r, \delta_s, \delta_b$ are angles of the rudder, stern plane, and bow plane, respectively, and η_v is the self-propulsion to actual velocity ratio ($\eta_v = 1$ at steady state). The fifth and eighth lines of equation (3) are models of the surge force produced by control surfaces, and the sixth line is a model of the force produced by a main thruster (using a_i, b_i, c_i as parameters).

From the velocity terms in the Gertler-Hagen model expressed in equation (3), it is clear that hydrodynamic damping is approximated as a quadratic function of velocity with coupling between the DoFs. In comparison, the notation of Fossen's model provides flexibility in how the hydrodynamic damping is calculated depending on the choice of $D(\nu)$. The velocity-based terms used to approximate the hydrodynamic damping of the underwater vehicle are what differentiate most of the classical parametric dynamic equations from one another. Examples of the velocity terms used by different models are listed in Table 1.

The choice of which velocity terms are used to approximate the hydrodynamic damping is based on the desired balance between accuracy and computation complexity. Factors such as a vehicle's hydrodynamic characteristics and its expected velocity range during operation will affect which specific velocity terms produce the most desirable accuracy-to-complexity ratio. Studies, such as [3] and [19], have conducted comparisons between parametric model simulation states and recorded physical trial results to determine the effects of hydrodynamic term selection on dynamic model accuracy. The eight different 6-DoF underwater vehicle dynamic models used in the comparison were the McFarland-Whitcomb, pitch-yaw, Gertler-Hagen, linear, Coe, Prestero, uncoupled, and Fossen models. Though there was not a clear advantage found between a majority of the models, it was seen that certain models produced less accurate state estimates due to the assumptions made in the velocity term selection being invalid for the tested maneuvers [19].

Although the selection of velocity terms does differentiate models, the accuracy of the coefficients used by a model is critical to its practical utility. Coefficients for the inertia, added mass, gravity, and buoyancy can be obtained from the vehicle's measurable physical characteristics. Captive model tests, or high-fidelity simulations, can be used to obtain a model's remaining coefficients by recording the forces experienced by a vehicle (or a scaled-down model of the vehicle) when traveling at a range of velocities varying in magnitude and direction [16]. The coefficient values are then approximated from the state-force data points. This process of fitting a function to model the relationship between observed inputs and outputs is known as regression. With the model's dynamic equations providing the structure of the function, several analytical semi-empirical methods are available for determining the coefficient values that provide the best correlation between the dynamic equations and the data set of state-force measurements.

TABLE 1. Velocity term choices of the different parametric dynamic equations that are compared in [19], with examples of the velocity terms consisting of the hydrodynamic coefficients X'_i and their relevant velocity components $i \in \{u|u|\ u|v|\ u\ |u|\ uv\ \ldots\ u^2\ v^2\}$ [19].

Model name	Velocity term(s)
McFarland-Whitcomb	Sign-preserved quadratic: $X'_{u\|u\|}u\|u\|, X'_{u\|v\|}u\|v\|, \ldots$
Pitch-yaw	Linear, quadratic variations: $X'_u u, Y'_{\|u\|}\|u\|, X'_{uu}u^2, X'_{u\|v\|}u\|v\|, X'_{uv}uv, \ldots$
Gertler-Hagen	Quadratic: $X'_{uu}u^2, X'_{vr}vr, \ldots$
Linear	Linear: $X'_u u, X'_v v, \ldots$
Coe	Quadratic variations (specific terms): $X'_{uu}u^2, X'_{u\|u\|}u\|u\|, X'_{vv}v^2, \ldots$
Prestero	Quadratic variations (specific terms): $X'_{qq}q^2, X'_{vr}vr, X'_{u\|u\|}u\|u\|, \ldots$
Uncoupled	Linear, quadratic variations (no coupling): $X'_u u, X'_{\|u\|}\|u\|, X'_{uu}u^2, X'_{u\|u\|}u\|u\|, \ldots$
Fossen	Linear, sign-preserved quadratic (no coupling): $X'_u u, X'_{u\|u\|}u\|u\|, \ldots$

The work in both [3] and [19] use the least-squares regression method to obtain their model coefficients from data sets constructed from physical underwater vehicle trial data. Each data set consisted of the vehicle's state (position, orientation, velocities) over the course of the trial. Paired with information regarding the vehicle's physical characteristics, the data set is augmented with estimates of the forces and moments from the inertia, added mass, gravity, buoyancy, and forward propulsion. Using this data with a given parametric model, that model's unknown hydrodynamic terms can be estimated for each data point. The least-squares method is then used to optimize the model's coefficients that correlate the hydrodynamic term estimates to the vehicle's state. The model comparison in [3] was conducted for two similar vehicles 690AUV and 690SAUV, each with a training and validation data set. The data sets each contained variations in depth, yaw, and propeller speed to capture a range of operational dynamics. Although the data sets are not available, it is stated that the 690AUV and 690SAUV training sets consisted of 200 seconds of data for four control conditions, and 280 seconds of data for six control conditions, respectively. The validation sets consisted of 30 seconds for two control conditions, and 70 seconds for three control conditions, respectively.

In [22], the least-squares regression method was used to obtain the sway and yaw coefficients from data of the ''Small Autonomous Underwater Vehicle That Tows a Large Payload'' vehicle, which was collected from virtual planar motion mechanism tests. Three sets of coefficients were estimated from the data - one using the geometric information of the vehicle, and the other two using simplified approximations of the geometry. Specifically, the coefficients estimated using the prolate spheroid and Myring profile geometries were compared against the true geometry. The predicted sway forces and yaw moments produced by dynamic models using the coefficient sets were compared to the recorded test data, demonstrating the accuracy of using least-squares regression and the higher similarity of the prolate spheroid approximation to the actual geometry.

The work by Ahmed et al. [23] proposes an improved analytical semi-empirical methods for 6-DoF Fossen model coefficient estimation from free-running trial data. The resulting model was analyzed by comparing its trajectory estimations with results from equivalent CFD simulations. Further examples of implementing parameterized underwater vehicle models have been surveyed in [14].

III. MACHINE LEARNING ALGORITHMS FOR MODELING DYNAMICS

Machine learning algorithms can be classified in different ways. From a learning strategy perspective, they can be classified into three main categories determined by the data used to condition or train them, namely supervised, unsupervised, and semi-supervised learning [24]. In supervised learning, the models are trained using paired input and output (i.e. labeled) data. The training algorithm usually learns a mapping between the input and desired output. In unsupervised learning, the algorithms are conditioned with unlabeled data. The resulting models are commonly trained to discover patterns or features within the input data. In semi-supervised learning, a combination of labeled and unlabeled data is used. The role of the training algorithm may therefore be to learn mappings from the labeled data and features from the unlabeled data, thereby reducing errors in prediction as compared to learning from the labeled data alone [25]. Models of maritime vehicle dynamics are generally most amenable to supervised learning algorithms, as training data sets are typically comprehensive. This is because the time histories of all control signals and vehicle states can often be readily obtained from free-running trials or captive-model runs.

Due to the broad definition of machine learning, all regression methods that are applied for parametric model coefficient estimation can technically be considered as a form

of supervised ML. For a parametric model, using parametric methods such as linear and polynomial regression is intuitive for estimating specific model coefficients individually [14]. To obtain most of these model coefficients, data sets consisting of a vehicle's correlated states and resulting hydrodynamic forces and moments must be created. Such data sets can be produced through physical system trials in controlled conditions, or through high fidelity CFD simulations. Given that the parametric model structure is predefined, the number of data points needed to estimate a coefficient can be minimized to match a dynamic model term's correlation to vehicle states. To reduce the impact of any disturbances and unmodeled dynamics represented in the data, it is common practice to sample data at regular intervals as seen in works surveyed by Ahmed et al. [14]. Once the main challenge of producing a comprehensive data set is addressed, the computational complexity of traditional regression methods are minimal in comparison to that of most ML algorithms such as neural network training.

The review conducted by Panda et al. [26] highlights the various methods of obtaining data sets from both controlled experiments and high-fidelity simulations. Unlike parametric models which simplify the vehicle to a point, the high-fidelity simulation methods all model the vehicle's geometry. This also allows for the modeling of time-dependent interactions between the vehicle geometry and the turbulence it produces during motion. Listed in order of increasing computational complexity, the methods reviewed are strip methods, panel methods, and CFD. The experimental methods for obtaining dynamic model data as seen in [22] and [23] may not be computationally complex, but instead require expensive specialized equipment, sensors, and a physical vehicle model.

However, machine learning is more commonly used to refer to non-parametric methods such as support vector regression, Gaussian process regression, and ANN training. It should be noted that ANN training is only non-parametric when the network structure is not predefined. Although a predefined network structure has a finite number of connections and is technically parametric, the data-driven nature of training the network for dynamic modeling does not have the limitations of classical parametric models. With a suitably large network structure and diverse data set, all prominent dynamics terms can be learned.

Early applications of ANNs in modeling underwater vehicle dynamics were primarily as replacements for the parametric regression methods used to determine parametric model coefficients, from either online or offline motion data. In online identification, regression is performed on real-time data pertaining to a vehicle's live motion. Derived model constants are continually updated, and reflect the live conditions of the vehicle. In offline identification, regression is performed on experimental data from free-running trials or captive-model runs.

In this manner, Xing and McCue [27] used ANN regression to identify hydrodynamic coefficients pertaining to dynamics in one degree of freedom (roll motion) of a surfaced vehicle, noting that the method can be extended to other degrees of freedom. The ANN itself is trained to estimate the produced rolling moment output from an input of vehicle states and actuator loads. The coefficients can then be extracted by specifying a parametric dynamic equation, and inserting the input states and estimated output moment.

To derive hydrodynamic coefficients for the damping in all six degrees of freedom of a submerged vehicle, Van de Ven et al. [28] used six segregated ANNs in the same manner, one for each degree of freedom. A quantitative comparison of the damping coefficients was conducted to demonstrate the accuracy of the method. As previously mentioned, [14] provides a survey of the developments in machine learning methods focusing on hydrodynamic coefficient estimation for underwater vehicles.

A second approach to applying ML algorithms for underwater vehicle dynamic modeling is to partially/completely replace the parametric model with the data-driven/non-parametric models that are learned from the observed input-output dynamics. It follows that either or all of the blocks in Figure 2 can be replaced or augmented with ML models. Unlike the parametric model equations which are limited in their fidelity by the pre-selected velocity terms, data-driven and non-parametric models effectively have unlimited fidelity. Indeed, they learn prominent input-output correlations, from the very large (practically infinite) number of possible parameters, within the machine learning model's architecture.

From this general understanding of how ML algorithms are implemented for underwater vehicle dynamic modeling, it can be inferred that certain methods are most suitable for different modeling applications. For applications that demand high accuracy over computational simplicity, and where adequate data is available, the use of ML-based non-parametric modeling is most suitable. However, for real time applications such as modeling for control feedback, a parametric model using pre-calculated coefficients provides the necessary response time. For such applications, ML-based regression of dynamic disturbance or modeling uncertainty terms is more suitable. In contrast, there is no obvious benefit for using ML-based regression for static coefficient estimation in predefined parametric models. A growing range of research has formed to explore these hypotheses, investigating the practical feasibility, limitations, and relative advantages of various proposed ML-based modeling methods for underwater vehicles.

IV. ARTIFICIAL NEURAL NETWORK METHODS

The architecture of an artificial neural network, also known as a feedforward neural network or multilayer perceptron, consists of an input layer, one or more hidden layers, and an output layer. Each layer is composed of multiple interconnected nodes, also called neurons, which perform computations on the input data.

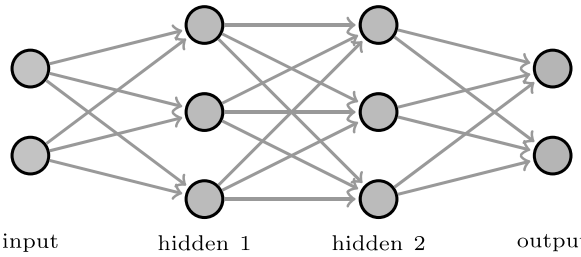


FIGURE 3. Schematic of a generic feedforward neural network with two hidden layers.

In an ANN, the nodes in each layer are fully connected to the nodes in the next layer. Each connection is associated with a weight, which determines the strength of the connection. During training, the weights are adjusted in order to minimize the error between the network's output and the desired output.

A simple ANN with two hidden layers, as shown in Figure 3, has limited ability to represent or approximate functions beyond a narrow and specific category. To address this challenge in ANNs, Hornik et al. [29] established the underlying theory which allows the utilization of "Multilayer Feedforward Networks," as universal estimators. The network typically consists of multiple layers of forward-connected neurons. These networks are commonly used for classification, regression, and data clustering, utilizing a set of weights to connect neurons in each layer and transmitting information forward through the network. Multilayer feedforward networks are typically employed for supervised learning, where input data is labeled, and the desired output is known. Hornik et al. [29] established that multilayer feedforward networks with a single hidden layer, equipped with any "squashing" activation function, can serve as universal approximators that are capable of approximating any Borel measurable function from one finite-dimensional space to another with a desired degree of accuracy. This eliminates the need for prior assumptions, linearization, or local time-stepping and requires an adequate number of hidden units.

Recurrent neural networks (RNNs) build upon the ANN structure by using the neuron states from the previous operation as part of the input, allowing RNNs to learn time-dependent behaviors. To achieve this, the data set for training an RNN consists of ordered pairs of input and observed output values, taken at discrete time intervals. The structures in the network that enable and control the state feedback are known as "Long Short-Term Memory" (LSTM) cells [30]. The LSTM cells control which states and prior time steps are memorized. When "unfolded" along a discrete time series of inputs, the RNN can be represented as a chain of ANNs linked together by the memorized outputs from the LSTM cells, as illustrated in Figure 4. Although the LSTM cells can only memorize a short interval of prior states, the dependence of those prior states on even older states allows the network to learn time-based dynamic relationships through the whole sequence in the data set.

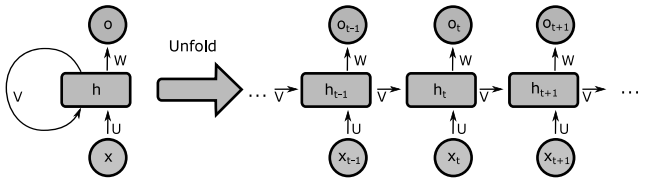


FIGURE 4. Schematic of a generic recurrent neural network with inputs x, outputs o, hidden layers h, memorized outputs V, and the network's corresponding unfolded structure through time.

A. LEARNED MANEUVER MODELS

Learned maneuver models are data-driven machine learning algorithms that learn to model the total vehicle dynamics, modeling the relationship between vehicle states and control signals, with the resulting vehicle velocity. Learning is supervised and is performed on training data from free-running trials. The dynamics are therefore learned from the training data. The form of functions defining loads and motions is implicitly learned, as opposed to being explicitly specified in parametric models. In reference to Figure 2, this approach lumps the actuator loads and dynamics blocks together in an ANN. Should the training data be inconsistent, such as being adversely affected by noise, it cannot be guaranteed that the learned model will conform with physical expectation and may result in nonphysical behavior.

Faller et al. [31], [32], [33] used a RNN with two hidden layers to model the time-varying states of a submerged vehicle. Each node would take the sum of its weighted inputs $\Sigma w_i x_i$, and use a binary sigmoid activation function:

$$f(x) = \frac{1}{1 + e^{-x}}, \tag{4}$$

to calculate the node's output. The network took the current vehicle state and control signals (i.e. propeller rotation rate and control surface deflection) to determine the vehicle state at the successive time step. The model was trained and validated on free-running trials data.

It is noteworthy that a form of dynamic similarity needs to be determined if the RNN is to be generalized across different length scales (i.e. when training on data from model scale to determine behavior at full scale). Faller et al [31] provided a working solution using dimensionless velocity (based on current speed):

$$v'(t' + \Delta t') = \frac{v(t' + \Delta t') \odot [1\ 1\ 1\ L\ L\ L]^\top}{U(t')}, \tag{5}$$

where L is the vehicle hull length, $U(t')$ is the translational speed, and t' is the current dimensionless time, which corresponds with the time required for the flow to travel the length of the hull:

$$t' + \Delta t' = t' + \frac{\Delta t\, U(t')}{L}, \tag{6}$$

where $\Delta t'$ is the chosen constant sampling time interval, and Δt is the varying sampling time interval that depends on the vehicle length and speed. Because of the varying sample time,

the number of data points obtainable from a given maneuver varies with $U(t')$ and L.

Furthermore, Faller et al. [32] noted that training data must be high quality (i.e. processed before it is fed to the RNN), and that the RNN architecture and time step size need to be determined by trial. Further maneuver simulations were conducted to evaluate the RNN model, which reinforced confidence in its accuracy and the observation that dynamic similarity should be made with the time-varying velocity rather than initial velocity [34], [35].

Results from the study indicated that the RNN can accurately predict maneuvers in 6-DoF motion, specifically crashbacks, rise jams, dive jams, rudder jams, turns, vertical overshoots and horizontal overshoots. The model also indicated that it had learned some characteristics of forced unsteady separated flows, which are not readily modeled with conventional parametric dynamic models, as it demonstrated adequate accuracy in predicting similar maneuvers at intensities outside of the training data range.

Following on from the work of Faller et al [31], [32], [33], Moreira and Soares [36], [37] developed an RNN for surfaced vehicles. The study showed that an RNN can be trained to predict maneuvers with sea trials data, however, it was identified that environmental factors can contaminate the training data, resulting in an improperly trained network. This can pose a point of concern, specifically when the RNN predicts nonphysical behavior.

B. LEARNED DYNAMIC MODELS

In later works, actuator load models were used to pre-process the control signals into actuator load estimates $\hat{\tau}$ for the same purpose of estimating the future dimensionless velocity of the vehicle. The premise of these studies is that the motion of a vehicle is not directly related to the desired control signals, but rather to the loads generated as a result of actual actuator outputs. The intermediate determination of loads therefore serves to relate control signals to the dynamics of the vehicle. As such, the RNN is effectively constrained physically, so long as the supplied loads are physical. The burden therefore falls on the loads module to remain physically meaningful. In Hess and Faller [38], [39] Hashem [40], and Hess et al. [41], [42], a parametric model was used for the actuator load model, which provide physically meaningful loads. The predetermined form of the parametric equations limit the fidelity of models, and may not consider conditions such as hydrodynamic stall. The RNN developed in [42] was trained for the faster-than-real-time 6DoF simulation of non-axisymmetric underwater vehicle motion. Training was conducted offline on 17 control conditions for 29000 epochs, though details regarding the data set size were not made available. The data was obtained through free-running trials of the vehicle, and the control conditions varied in turn angle and forward velocity while using different combinations of the vehicle's available actuators. The trained network's modeling accuracy was then validated against a data set of 7 control conditions, with the accuracy of a predicted time series being quantified by the Average Angle Measure $AMM = 0.860$ and correlation coefficient $R = 0.932$, where a value of 1 corresponds to a perfect magnitude and phase correlation to the validation data.

Chiu et al. [43] used the same parametric actuator load model, and demonstrated that their trained RNN was capable of capturing the effects of unsteady fluid dynamics in the vehicle dynamics. To model the propeller thrust X_p that was used as an input for these RNNs, the following parametric equations were used:

$$X_p = (1 - t_p)\rho n^2 D_p^4 K_T \tag{7}$$

$$K_T = a_0 + a_1 J_p + a_2 J_p^2 \tag{8}$$

$$J_p = \frac{u_p}{nD_p} \tag{9}$$

$$u_p = u(1 - \omega_p), \tag{10}$$

where ρ is the water density, D_p is the propeller diameter, K_T is the thrust coefficient estimate with model parameters $[a_0\ a_1\ a_2]$, J_p is the advance ratio, and u_p is the propeller's effective inflow speed estimate related to the wake fraction ω_p. The net thrust acting on the vehicle is given by $X = X_p - R_r$, where R_r is the resistance for the rudder deflection δ_r. The deflected rudder also produces the lateral force Y_r at the aft of the ship, which produces a yawing moment. The equations for these rudder forces are:

$$R_r = (1 - t_r)F_N \sin(|\delta_r|) \tag{11}$$

$$Y_r = -F_N \cos(\delta_r), \tag{12}$$

where t_r is the rudder force deduction factor and F_N is the normal force acting on the rudder. Further details on the parametric component of the models, such as the equation for F_N and the Munk moment, can be found in [43] and similar works by Hess and Faller [39] and Hess et al. [42] that also use a parametric actuator load model connected to an ANN.

The RNN and parametric actuator model combination was also deemed suitable as a plant model for systems not well suited to linearized plant models. The applied model required less training data to achieve similar accuracy as related works. For example, only 20 rise-to-jam experiment results were needed to acceptably predict the responses of all rise-to-jam maneuvers.

The capability to learn underwater vehicle dynamics has also been applied to hull design applications. Faller et al. [44], [45] used the same RNN framework and used the hull geometry as an additional input, demonstrating that the trained RNNs can predict the dynamic response of vehicles following changes to their hull shape. A similar study is conducted in [46] for estimating maneuvering characteristics for surface ships. Instead of training to estimate vehicle dynamics, an ANN is trained to estimate characteristics such as the vehicle's turning circle diameter. This process provided fast feedback on the capabilities of vehicle hull designs while being far less computationally

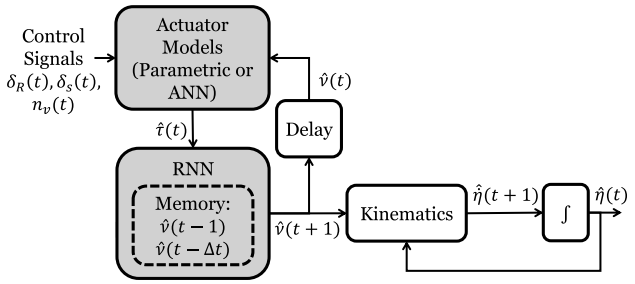


FIGURE 5. Block diagram of how an RNN is used for modeling dynamics with a separate actuator model, based on the works in [42] and [47].

intense than performing multiple simulations to determine these maneuvering characteristics.

In [47], the parametric actuator models (specifically for propellers) were replaced with separate ANNs. In this approach, the actuator load models consist of feedforward neural networks that estimate the output τ as functions of control signals and vehicle states. Separate ANNs were trained, each correlating the input vector to one element of the output τ. The use of an ANN for underwater vehicle actuator modeling is further explored in [48] with quantified analysis on the accuracy of the ANN models.

The resulting implementation of an RNN for dynamic modeling is illustrated in Figure 5, where the actuator load models (parametric or using ANNs) provide the inputs to the RNN. The prior output velocities are memorized and used as further inputs to estimate the future velocity as the RNN's output. The vehicle's velocity in the inertial frame is then obtained using the kinematic equations, and can be integrated to estimate the vehicle's position and orientation.

As noted by Van de Ven et al. [28], when separate networks are employed for each output element, the dynamics are uncoupled in each degree of freedom. This helps reduce under/over training since the rates of learning can differ between degrees of freedom. It is noteworthy that this approach improves training and prediction performance, and is utilized in the RNN implementations already discussed above. What differentiates the work of Van de Ven et al. [28] is the further use of ANNs to estimate individual components of the dynamic equation. Specifically, ANNs are trained to approximate the responses of the inertia term M, gravity term $g(\eta)$, Coriolis term $C(\nu)\nu$, and damping term $D(\nu)\nu$ to form the dynamic model as expressed by (1).

C. CORRECTOR TO EXISTING MODEL

Although the advantages of using ANNs for dynamic modeling are clear, for many underwater vehicles, parametric models may have been already developed from the design process or prior control implementations. Using an ANN together with a parametric model provides dynamic modeling capabilities with a predefined level of baseline performance, while taking advantage of the increased modeling flexibility and fidelity made possible through ANNs. These characteristics enable the modeling of uncertain disturbances and vehicle parameter variations that are unaccounted for in a parameterized model. If performed in real time, ANN-based model correction can be implemented as part of an adaptive model-based controller for underwater vehicles as demonstrated in the recent work by Gong et al. [49].

The study by Marlantes and Maki [6] demonstrated the techniques in predicting effects on the nonlinear vehicle motions from a range of wave conditions using an RNN. Similar work is reported in [7] and [50], where the external disturbance forces and moments (from wind and waves, respectively) that affect a surface ship are estimated by an ANN.

Faller et al. [51] demonstrated a different approach to augmenting a parametric model with an ANN. Here, RNNs are trained to correlate the parametric model's net forces and moments estimate $M_t\dot{\nu}_t$ with the error $M\dot{\nu} - M_t\dot{\nu}_t$ between the estimate and its corresponding simulations/experimental value. Once trained, the output of the RNN is summed with the parametric model estimate such that its errors are compensated. The study demonstrated that it is possible, within limits, to create estimates of vehicle motion, with no significant difference between model and experiment.

D. RECENT ARCHITECTURE ADVANCEMENTS

The versatility of the general ANN architecture has enabled the development of countless architecture variations, which are still being explored by the recent literature. Although these novel implementations are often for other dynamic systems or for applications other than dynamics modeling, they do indicate the feasibility of these ANN architectures for underwater vehicle modeling.

Feng et al. [52] explores the use of a Graph Convolutional Neural Network (GCNN) in classifying an underwater vehicle's motion state as either "straight near surface", "straight at fixed depth", or "divergent at fixed depth due to external disturbance". The proposed implementation for multivariate time series classification was demonstrated to have higher classification accuracy than a classification support vector machine, and two other deep ANN architectures.

Many recent works which attempt to use neural networks for modeling dynamic systems implement a form of physics-informed neural network. A relevant example can be found in [53], where an ANN is trained to approximate the 3-DoF dynamic model of a surface ship. The dynamic model, along with models for steering and forward speed are used as loss functions to train the ANN to output the expected velocities for a given input time. Compared to an ANN that was trained to predict velocities directly, the velocity mean square errors of the physics-informed neural network were an order of magnitude lower at all the investigated training data set sizes (1000 to 5000 samples).

V. PHYSICS-INFORMED NEURAL NETWORKS

Unlike RNNs, physics-informed neural networks (PINNs) as proposed in [54] are not associated with a specific network structure. Rather, the capability of deep ANNs to act as

universal function approximators is utilized [29]. The choice in network inputs and training loss function is what enables PINNs to output physically valid solutions or approximations of nonlinear partial differential equations (PDEs).

PINNs incorporate the PDEs and output prediction errors in their loss functions during training, and are trained using data of the boundary and initial conditions of the PDEs' solutions. This enables PINNs to learn from a relatively small amount of data by incorporating prior knowledge of the underlying physics. It should be noted that the number of layers and neurons per-layer in the network must be chosen to provide sufficient approximation capacity for the expected complexity of the solution function. Procedures such as Bayesian optimization can be used to determine suitable values for these network hyperparameters.

Raissi et al. [54] presented two training methods (for discrete and continuous time models) to produce PINNs that output the solutions to a PDE (acting as an approximation of the solution function), or that output estimates of the parameters/coefficients. The quantified performance of these methods was demonstrated on physics problems from various disciplines, including fluids, quantum mechanics, reaction-diffusion systems, and the propagation of nonlinear shallow-water waves. The networks themselves were simple deep feed-forward networks using hyperbolic tangent activation functions, without any regularization layers.

We can express the general formula for a nonlinear PDE as:

$$\frac{\partial}{\partial t}u + N(\lambda, u, \frac{\partial}{\partial x}u, \frac{\partial^2}{\partial x^2}u, \ldots) = 0, t \in [0, T], \tag{13}$$

where $N(\lambda, u, \frac{\partial}{\partial x}u, \ldots)$ is a nonlinear function of the parameter vector λ, the hidden solution function $u(t, x)$ and its partial derivatives in terms of the space (e.g. state vector) x. By specifying a PDE in this notation, a PINN can be trained to perform one of two functions: estimating the outputs of $u(t, x)$ given the fixed model parameters λ, or estimating the parameters λ given training data that correlates outputs of $u(t, x)$ to the inputs of time t and state vector x. For an underwater vehicle, x will consist of the states η, ν, and τ. Note that the u, x, and y variables used in this section and Section VI do not refer to the scalar state variables described in Section II.

A. SOLVING PDES WITH A PINN

The solution function $u(t, x)$ is defined as a deep ANN, such that the partial derivatives of the function can be obtained using the gradients of the network. We can then define the PINN f using the left-hand side of the general PDE equation, the ANN u, and the partial derivative function N using the defined parameters λ of the PDE:

$$f = \frac{\partial}{\partial t}u + N(\lambda, u, \frac{\partial}{\partial x}u, \frac{\partial^2}{\partial x^2}u, \ldots). \tag{14}$$

Both networks are then trained to minimize the mean square error (MSE) loss function:

$$MSE = MSE_u + MSE_f \tag{15}$$

$$MSE_u = \frac{1}{n_u}\sum_{i=1}^{n_u} \|u(t_u^i, x_u^i) - u^i\|^2 \tag{16}$$

$$MSE_f = \frac{1}{n_f}\sum_{i=1}^{n_f} \|f(t_f^i, x_f^i)\|^2, \tag{17}$$

where MSE_u enforces the initial and boundary conditions of the PDE using the training data set $\{t_u^i, x_u^i, u^i\}_{i=1}^{n_u}$, and MSE_f enforces the structure of the PDE using the set of collocation points $\{t_f^i, x_f^i\}_{i=1}^{n_f}$. In [54], the limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm was used to optimize the loss function.

The trained ANN $u(t, x)$ can then approximate the solution of the PDE for time and space input outside the training data set. It provides computationally efficient inference compared to the traditional analytical/experimental methods utilized to generate the training data.

A relevant example presented in [54] was the use of a PINN trained using the Navier-Stokes equation (in two dimensions). Trained on noisy measurements of the velocity field $\{t^i, x^i, y^i, u^i, v^i\}_{i=1}^N$ around a circular cylinder, the resulting PINN could accurately identify the Navier-Stokes equation parameters and predict the pressure field function. Such a method could be applied to augment results from the CFD simulations that are used in modeling underwater vehicles.

A recent review on PINNs [55] further discusses the potential of future applications in dynamics modeling. The three methods of incorporating physics information into an ANN are described as observational bias (training on data which reflects the physics), inductive bias (modifying the ANN model architecture to adhere to mathematical constraints), and learning bias (selecting the loss function, state constraints, and post-model inference algorithms to express physical relationships). A combination of these methods can be used to develop a PINN, with the review providing example implementations ranging from 3D fluid dynamics estimation to molecular simulation. The current limitations of PINNs are also highlighted, including difficulties learning high-frequency functions due to steep gradient, and learning multiple simultaneous physics concepts due to computational complexity. The more complex loss functions used to develop a learning bias are also likely to be highly non-convex, such that convergence to a global minimum cannot be guaranteed during training.

B. PINN FOR AN UNDERWATER GLIDER

Lei et al. [4] proposed a new method for modeling the dynamics of underwater gliders (UGs) that combines physics-driven and data-driven modeling. Their PINN-based method uses a theoretical model, including subsystem models developed by [56] and [57], as a baseline and compensates

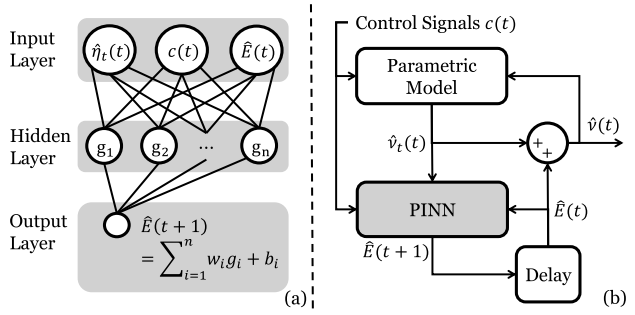


FIGURE 6. (a) The network structure of the RBF PINN using Gaussian activation functions $g_i|i \in [1, n]$ for a hidden layer with n nodes, and (b) the framework for using the PINN to estimate the modeling errors E of the parametric model output v_t as proposed in [4].

for modeling errors using an ANN known as a radial basis function (RBF) neural network. This ANN consists of a single hidden layer and uses an RBF activation function:

$$g_i = \exp(-\frac{\|x - C_i\|^2}{2b_i^2}), \tag{18}$$

where C_i and b_i are the center and width hyperparameters of the Gaussian function g_i for the i^{th} node in the hidden layer that takes an input vector of x. This network's structure is illustrated in Figure 6(a).

The ANN was trained on prerecorded trial data. The control signals $c(t)$, the state estimate produced by a parametric model $\hat{v}_t(t)$, and the error of the parametric model estimate $E(t) = v(t) - \hat{v}_t(t)$ (relative to external measurements) were used as the inputs $x = [c\ \hat{v}_t\ E]$. These inputs were paired with the observed modeling error of the next time step as the output $y = [E(t+1)]$. The MSE loss function for training using gradient decent was defined as:

$$MSE = \frac{1}{n}\sum_{i=1}^{n} \|E_i - \hat{E}_i\|^2, \tag{19}$$

where n is the number of data points in the data set $D = \{(c_i, \hat{v}_t^i, E_i, E_{i+1})|i = 1, \ldots, n\}$. The resulting trained PINN is used to estimate the modeling error in the absence of external measurements by using its estimated error output $\hat{E}(t+1)$ as an input in the next time step. The final state estimate is produced using:

$$\hat{v}(t+1) = \hat{v}_t(t+1) + \hat{E}(t+1). \tag{20}$$

The implementation of the PINN with the parametric model to produce the final velocity estimate, and how the estimate error is fed back to the PINN, are illustrated in Figure 6(b).

A sliding window method was implemented to improve the online learning capabilities of the PINN. The hydrodynamic characteristics of the UG were determined using towing tank tests and CFD. The proposed method is compared with both a CFD-based theoretical model and an RBF-based data-driven method, and is shown to improve accuracy by 41 and 82 percent, respectively. The PINN method was also shown to be applicable to other underwater vehicles. Overall, their proposed method was an effective and efficient approach for modeling the dynamics of UGs and other underwater vehicles.

C. LAGRANGIAN NEURAL NETWORKS

A broad range of methods for incorporating concepts of physics into ANNs have since been developed for similar applications. This includes physics-constrained neural networks [58] for material modeling, variational PINNs [59] for PDE solution approximation, and conservative PINNs [60] for nonlinear conservation laws. The Lagrangian Neural Network (LNN) is another variant of the PINN, which is trained to account for the conservation of energy in its approximations of system models.

The theoretical foundation of LNNs is grounded in the Euler-Lagrange equation for each DoF $j \in [1, 6]$ (see [61]):

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\eta}_j} = \frac{\partial L}{\partial \eta_j} \tag{21}$$

$$L = T(\eta, \dot{\eta}) - V(\eta), \tag{22}$$

where L is the Lagrangian, an equation of the kinetic energy $T(\eta, \dot{\eta})$ and potential energy $V(\eta)$ within a closed system.

A modified application of the Lagrangian for training ANNs is proposed in [62], which accounts for non-conservative forces in systems with actuators using:

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\eta}_j} = \frac{\partial L}{\partial \eta_j} + \tau_j, \tag{23}$$

where τ_j expresses the actuation forces in each DoF, and can be used to account for damping forces that are a function of the generalized system state variables.

By defining the vector $\nabla_\eta L := [\frac{\partial L}{\partial \eta_1} \ \ldots \ \frac{\partial L}{\partial \eta_6}]^\top$, the Euler-Lagrange equation can be expanded in vector form as:

$$\frac{d}{dt}\nabla_{\dot{\eta}} L = \nabla_\eta L + \tau \tag{24}$$

$$(\nabla_{\dot{\eta}}\nabla_{\dot{\eta}}^\top L)\ddot{\eta} + (\nabla_\eta \nabla_{\dot{\eta}}^\top L)\dot{\eta} = \nabla_\eta L + \tau, \tag{25}$$

which can be used to obtain an estimate of $\ddot{\eta}$ using:

$$\ddot{\eta} = (\nabla_{\dot{\eta}}\nabla_{\dot{\eta}}^\top L)^{-1}(\nabla_\eta L + \tau - (\nabla_\eta \nabla_{\dot{\eta}}^\top L)\dot{\eta}). \tag{26}$$

By using an LNN to approximate the Lagrangian $L(\eta, \dot{\eta})$, the partial derivative vectors ∇_x and $\nabla_{\dot{x}}$ are obtained using the Hessian operation on the trained LNN to obtain its gradients [5]. Finally, (26) can be used to obtain the estimated acceleration. This process is summarized in Figure 7.

An ANN is trained into an LNN through back-propagation with the objective of minimizing the MSE loss between the observed and estimated accelerations:

$$MSE = \frac{1}{n}\sum_{i=1}^{n} \|\ddot{\eta}_i - (\nabla_{\dot{\eta}_i}\nabla_{\dot{\eta}_i}^\top L)^{-1}(\nabla_{\eta_i} L + \tau - (\nabla_{\eta_i}\nabla_{\dot{\eta}_i}^\top L)\dot{\eta}_i)\|^2. \tag{27}$$

Mirzai [5] demonstrated the use of an LNN to learn the state-space model [63] of an underwater vehicle, accounting

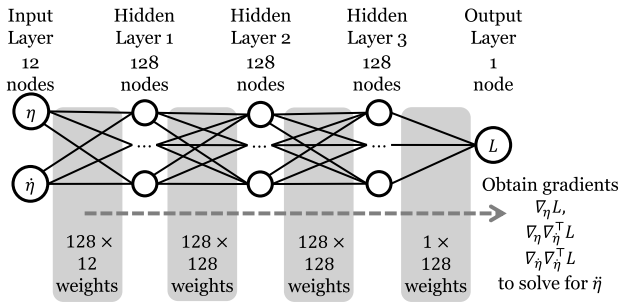


FIGURE 7. Illustration of the Lagrangian Neural Network proposed in [5]. The network weights are used to obtain the Lagrangian gradients $\nabla_\eta L$, $\nabla_\eta \nabla_{\dot{\eta}}^T L$, and $\nabla_{\dot{\eta}} \nabla_{\dot{\eta}}^T L$ from the network weights to calculate the dynamic response $\ddot{\eta}$ using (26).

for non-conservative control forces. The chosen structure for the LNN is shown in Figure 7, containing three hidden layers of 128 nodes, an input of $x = [\eta\ \dot{\eta}]$, and an output of the system's Lagrangian $y = [L]$ as specified in (22).

An additional second ANN was also trained to estimate the damping in the system that was not considered by the LNN. The damping estimate was added to the LNN's output to form the estimated total dynamic response. Both networks were trained to learn from over 65,000 data points obtained through system simulations, consisting of initial states (input) and final states (output) after an integration step of 0.01 seconds. The estimates produced by the LNN were compared to the response of the true dynamic equations over a range of integration steps. The results demonstrated the limitations of training an LNN on a non-conservative system for approximating the dynamics, and that the implemented second ANN did not effectively learn the non-conservative forces to compensate for the LNN's limitation.

VI. ARTIFICIAL NEURAL NETWORK ALTERNATIVES

The use of ANNs is only a subset of the various non-parametric machine learning methods. In the application of underwater vehicle modeling, the use of support vector regression and Gaussian process regression have been found to produce practical results with certain benefits when compared to ANN methods. These non-parametric regression methods are also differentiated from statistical regression methods, as the model structure is learned rather than predefined (as linear, polynomial, etc.) based on assumptions [64], and the use of optimization methods like gradient descent produces estimates rather than the analytical solutions of the correlations produced by least-squares methods. These differences from parametric regression methods allow non-parametric regression methods to learn more complex models with less manual configuration, while needing less computational resources when learning on larger data sets.

A. SUPPORT VECTOR REGRESSION METHODS

Support vector machines (SVMs) have a similar structure to ANNs. These similarities also produce comparable capabilities in classification and estimation tasks. In classification tasks, SVMs learn a high-dimension boundary called a "hyperplane" which separates classes. The input data is transformed into higher dimensions using kernel functions, and a subset of the training data (known as the support vectors) is used to maximize the hyperplane's distance between all classes. The support vectors are selected as data points within a margin from the hyperplane. Since only the support vectors are necessary for the optimization, SVMs do not require large training data sets like ANNs.

Support vector regression (SVR) is a machine learning method derived from SVMs with the objective of function approximation. Instead of maximizing the distance from training data points, SVR maximizes the number of observation points within a certain distance from the high-dimensional estimated function $f(x)$, given by:

$$f(x_*) = \sum_{i=1}^{n} w_i \kappa(x_i, x_*) + b, \tag{28}$$

where w is a linear weight vector to be learned, b is an offset parameter to be learned, and $\kappa(x_i, x_*)$ is a chosen kernel function that maps the input vectors to a higher-dimensional space. The support vectors are a subset of the training data set that is selected as the data points outside of a margin ϵ from the hyperplane.

The function is trained to act as an estimate of the unknown relationship function between the inputs and outputs in the training data set $D = \{(x_i, y_i) | i = 1, \ldots, n\}$ consisting of n points. By ignoring the points within a distance ϵ from the objective function, overfitting during training is prevented. Discrete output values can then be extracted from the trained function for a given set of input states. The use of the boundary distance results in a function that may not coincide with the observed data points.

To learn the function's parameters, the following objective function is minimized:

$$L(\alpha, \alpha^*) = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) \kappa(x_i, x_j) + \epsilon \sum_{i=1}^{n} (\alpha_i + \alpha_i^*) - \sum_{i=1}^{n} y_i (\alpha_i - \alpha_i^*), \tag{29}$$

under the constraints:

$$\sum_{i=1}^{n} (\alpha_i - \alpha_i^*) = 0 \tag{30}$$

$$\forall i : 0 \leq \alpha_i \leq C \tag{31}$$

$$\forall i : 0 \leq \alpha_i^* \leq C, \tag{32}$$

where the weight vector is determined by $w = \alpha - \alpha^*$, and C is the positive box constraint that penalizes data points outside of ϵ to reduce overfitting.

The terms in the objective function minimize the weight terms α and α^*, and the error between $f(x_i)$ and y_i. The optimal solution for minimizing the objective function can

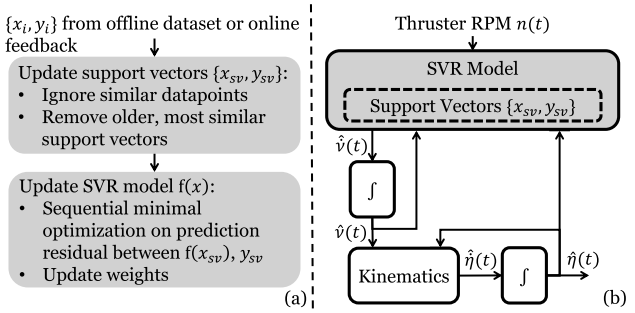


FIGURE 8. (a) The training process of a SVR model, and (b) the implementation of an SVR model for underwater vehicle modeling as presented in [8].

be obtained by satisfying the additional optimization constraints known as the Karush-Kuhn-Tucker complementarity conditions:

$$\forall i : \alpha_i(\epsilon + \xi_i - y_i + f(x_i)) = 0 \tag{33}$$
$$\forall i : \alpha_i(\epsilon + \xi_i^* + y_i - f(x_i)) = 0 \tag{34}$$
$$\forall i : \xi_i(C - \alpha_i) = 0 \tag{35}$$
$$\forall i : \xi_i^*(C - \alpha_i^*) = 0, \tag{36}$$

where ξ_i and ξ_i^* are slack variables that consider data points beyond ϵ in the regression. These conditions also infer that data points within ϵ have a weight $w_i = 0$. All other data points with non-zero weights are considered support vectors [65]. These steps for training an SVR model are summarized in Figure 8(a).

Wang et al. [66], [67] employed ν-"support vector regression" to construct a model for ship maneuvering motion. Introduced in [66], the ν-SVR algorithm is implemented for surface ship model parameter estimation. It uses a cost function to select the disturbance level ϵ, which controls the insensitivity tube size used for the support vector selection. This provides added robustness against different disturbance levels when compared to SVR algorithms that are trained using a fixed/pre-selected ϵ or support vector count. In [67] the work is extended to utilize Kernel-based regularization, a non-parametric system identification approach, to enhance the accuracy of their model by curbing model complexity and overfitting of data. This regularization technique relies on penalty terms that assess the similarity between diverse data points and yield a trade-off between model simplicity and accuracy. However, due to the intrinsic nature of support vector regression, which seeks to describe a highly nonlinear system using a multiple linear regression framework, identification accuracy may be compromised when limited data are available.

Wehbe et al. [8], [9], [10] have done extensive work on applying SVR to model underwater vehicles, with the online model updating during operation. When compared to parametric damping models that were identified using the least-squares method, the SVR models demonstrated superior accuracy in estimating the dynamic responses in a testing data set. To perform the online optimization in real time, the total set of training data points must be limited. A simple solution is to discard the oldest data when new data is received, but this approach can cause the model to "forget" useful dynamics that have not been experienced recently.

In one of their works, Wehbe et al. [9] propose an inclusion and forgetting criteria to control the data set size during online training. The inclusion criteria limits the new data points that are added by ensuring that the observed output y_i does not lie within the ϵ range of $f(x_i)$, and that the new data point's input and output are not within a predefined range of an existing data point. The forgetting criteria is used when the maximum desired data set size is exceeded, and works by segmenting the data set into bins at a set resolution, then removing the oldest data point from the most populated bin. Using this method an SVR model was trained to estimate the damping term of the dynamics for yaw motions, using the input $x_i = [r]$ and the output $y_i = [(D(\nu)\nu)e_6]$, where $e_6 = [0\ 0\ 0\ 0\ 0\ 1]$ is a unit vector that that isolates the yaw component of the damping vector term $D(\nu)\nu$. The online training demonstrated the ability for the estimated damping term to adapt to changes caused by an added physical component to the vehicle. The radial basis function kernel function was chosen for this implementation, defined as:

$$\kappa(x_a, x_b) = \exp(-\gamma \|x_a - x_b\|^2), \tag{37}$$

where γ is a scalar hyperparameter that determines the smoothness of the kernel function.

Another variant of an SVR model was developed in [10], where the input was limited to exclude the position and orientation data to predict vehicle accelerations for planar motion. Specifically, the inputs were defined as $x_i = [u\ v\ r\ n_1\ n_2\ n_3]$ which include the thruster control signals n_1, n_2, n_3, and the output as $y_i = [\dot{u}\ \dot{v}\ \dot{r}]$. This work explored the use of a weight vector in the kernel function, such that each element in the input vector would have an individual hyperparameter to allow the kernel function to learn the relative importance of the inputs to one another. The resulting weighted-distance-squared-exponential kernel had the function:

$$\kappa(x_a, x_b) = \exp(-\sum_{j=1}^{N} \gamma_j \|x_{a_j} - x_{b_j}\|^2), \tag{38}$$

where N is the length of the input vector (in this implementation $N = 6$), and γ is the weight vector.

In a more recent work, Wehbe et al. [8] have proposed an improved forgetting criteria where each data point is assigned a metric:

$$\phi = \frac{d}{\sqrt{t_s} + k}, \tag{39}$$

where d is the density of surrounding data points, t_s is the timestamp at which the data was received, and k is a constant weight parameter that determines the importance of the timestamp relative to the density. This metric ϕ is therefore higher for older and more redundant data points.

When the maximum number of data points is exceeded, the data with the highest ϕ is forgotten. This technique was used to train an SVR model on the dynamic response of an underwater vehicle moving in the horizontal plane. The model had an input consisting of the states and control signals n_1, n_2, n_3 in the surge, sway, and yaw degrees of freedom $x_i = [x\ y\ \psi\ u\ v\ r\ n_1\ n_2\ n_3]$, and an output of the accelerations in the same degrees of freedom $y_i = [\dot{u}\ \dot{v}\ \dot{r}]$. This input and output structure is shown in Figure 8(b), also illustrating how the kinematic equations can be used with the SVR model to estimate the vehicle's position. As the support vector models are multi-input single-output systems, separate SVR models are trained to account for the three output vector element. The squared exponential kernel function was chosen for this implementation, defined as:

$$\kappa(x_a, x_b) = \exp(-(x_a - x_b)^\top S^{-1}(x_a - x_b)), \tag{40}$$

where $S = \Sigma/\gamma$ is a matrix proportional to the covariance matrix Σ on the training data by the constant γ.

B. GAUSSIAN PROCESS REGRESSION METHODS

As explained in [68], "a Gaussian process (GP) model describes a probability distribution over possible functions that fit a set of points". The mean of the GP model provides the most probable estimate of the target function that produced the data points. The variance can be used to quantify the uncertainty of an estimate, which is a unique capability of GP models when compared to other regression and classification methods. Gaussian process regression (GPR) is the machine learning method used to produce a model for function estimation. GPR assumes that the input-output relationship that is being approximated $f(x)$ will contain Gaussian noise ε such that the data set of n points $\{(x_i, y_i)|i = 1, \ldots, n\}$ is given by:

$$y_i = f(x_i) + \varepsilon, \tag{41}$$

where ε has a mean of zero and a variance of σ_ε^2. This makes the GPR model inherently robust to noisy data [11]. Another difference between SVR and GPR is that with optimized hyperparameters, GPR will prioritize the function's coincidence with the input-output pairs used as data points for the regression.

When passing the input x_* to a trained GPR model, it estimates the output y_* and its variance $cov(y_*)$ as:

$$y_*|X, y, x_* = k(x^*, X)(K(X, X) + \sigma_\varepsilon^2 I)^{-1} y \tag{42}$$

$$cov(y_*) = \kappa(x_*, x_*) - k(x_*, X)^\top (K(X, X) + \sigma_\varepsilon^2 I)^{-1} k(x_*, X), \tag{43}$$

where $X = [x_1\ \ldots\ x_n]^\top$ is the matrix of all data set input state vectors, and $\kappa(x_a, x_b)$ is the kernel function between the input vectors x_a and x_b. The vector $k(x_*, X) = [\kappa(x_*, x_1)\ \ldots\ \kappa(x_*, x_n)]^\top$ consists of the kernel values between the test input x_* and the data set inputs x_i, and K is the matrix of kernel values relating the data set inputs with one another such that $K_{ij} = \kappa(x_i, x_j)|i, j \in [1, n]$ [13].

The RBF kernel function used in Gaussian process regression is defined as:

$$\kappa(x_a, x_b) = \sigma^2 \exp(-\frac{\|x_a - x_b\|^2}{2l^2}), \tag{44}$$

where the standard deviation σ and characteristic length scale l are the hyperparameters $\Theta = [\sigma\ l]$ that will be learned to improve the GPR model's estimate of $f(x_*)$. As summarized by Figure 9(a), the training of the GPR's kernel hyperparameters is achieved by maximizing the logarithmic marginal likelihood:

$$L(\Theta) = -\frac{1}{2}\log(\|K(X, X)\|) - \frac{1}{2} y^\top K(X, X)^{-1} y - \frac{n}{2}\log(2\pi), \tag{45}$$

where $y = [y_1\ \ldots\ y_n]$ is the vector of observed outputs that correlate to X in the training data set, and n is the number of data points in the training data set [11]. Methods such as particle swarm optimization and gradient descent are used to approximate the solution for the optimization of Θ. The updated kernel hyperparameters are then used in (42) to approximate the system dynamics, and the variance of the estimate using (43).

Ramirez et al. [11], [12] explored the use of GPs as non-parametric models for surface ships and underwater vehicles. The multiple approaches include the incorporation of an unscented Kalman filter for position estimation, with a GPR model replacing the parametric model used to calculate system states from sensor readings [11]. The model was trained offline using a data set consisting of the input states $x_i = [\eta(t)\ \nu(t)]$ and observed outputs $y_i = [\eta(t+t_\Delta)\ \nu(t+t_\Delta)]$ consisting of the same states after a set time interval t_Δ. As a GPR model approximates the relationship between the input vector and an output value, separate models are trained for each element of the output vector.

A more recent work avoids the use of multiple models by utilizing a single multi-output GPR model for the non-parametric system identification of an underwater vehicle [13]. The vehicle velocity is used with the actuator control signals for the thruster n and control surface angles δ_r and δ_s to form the input states $x_i = [\nu\ n\ \delta_r\ \delta_s]$. The multi-output GPR model is trained to output the resulting system velocity $y_i = [\nu]$, and has the advantage of learning relationships between outputs. This implementation of a GPR model is illustrated in Figure 9 (b) which shows the connection between model inputs and prior outputs in discrete time intervals, and how the position estimate is calculated using the vehicle's kinematics. The performance of the multi-output GPR model is compared to two RNNs with different memorized states, all trained on the same data set. Compared to both RNNs, the GPR model produced a lower average root-mean-square error, predicted residual error sum of squares, and mean absolute error in its estimated output velocity.

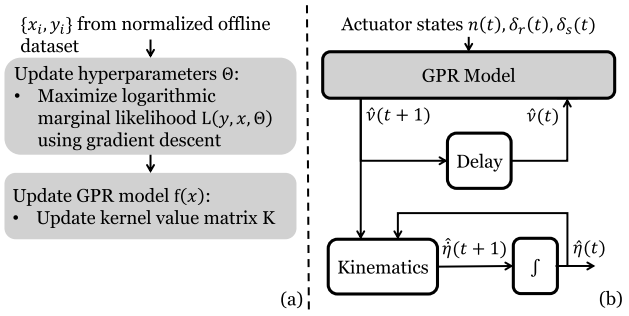


FIGURE 9. (a) The training process of a GPR model, and (b) the implementation of a GPR model for underwater vehicle modeling as presented in [13].

C. SUPPORT VECTOR AND GAUSSIAN PROCESS REGRESSION COMPARISONS

With both Gaussian process and support vector regression having similar applications in underwater vehicle modeling, the two methods have been quantitatively compared in the literature. In terms of computational complexity, the learning of the weight vector in SVR has a complexity of $O(n^2)$ while the matrix inversion for learning the kernel hyperparameters in GPR has a complexity of $O(n^3)$ [8].

Van de Ven et al. [28] also compared ANN regression to least-squares regression, observing that the former yields better estimates of hydrodynamic coefficients from experimental data. This was determined to be due to its ability to generalize, and hence filter out, the effects of noise in data. This result was also determined by Wehbe et al. [15], who examined Gaussian process regression, kernel ridge regression, ANN regression, and support vector regression, observing that these methods yield better estimates of hydrodynamic coefficients from experimental data, in comparison to least-squares regression. The reason for this is twofold. On one hand, these techniques provide a means for robust regression, effectively reducing or removing the effects of outlier data. On the other hand, the form of hydrodynamic loads terms can be learned with these techniques and need not be predetermined, as is the case with least-squares regression. This permits for nonlinearities to be modeled, as would be observed when relations break down due to hydrodynamic stall.

Concluding from the study, Wehbe et al. [15] determined that Gaussian process regression is useful for providing a confidence interval for predictions. However, it was deemed too computationally expensive in comparison to the other methods, and increasingly so for larger data sets. This was also noted by Ouyang et al. [69], who proposed an improved variant of Gaussian process regression for determining hydrodynamic coefficients, but concluded that further work should be directed toward improving its computational efficiency. The other methods trialed by Wehbe et al. [15] were comparatively less computationally expensive, except for ANN regression; although it was expensive to train for small data sets, it was comparatively less expensive with increasing data set size, and considerably less expensive to evaluate.

The relatively low computational cost of kernel ridge regression, ANN regression, and support vector regression, in combination with their ability to generalize, renders these methods favorable to both offline and online identification. Support vector regression, in particular, has gained increased attention in identification. The technique has been applied to identify hydrodynamic coefficients pertaining to motions from free-running trials data [67], [70], [71], captive-model runs data [72], as well as simulated motion data from a coefficient-based dynamic model Xu et al. [73], Chen et al. [74]. The accuracy of the models produced through these non-parametric regression models is demonstrated by comparing the model outputs to the recorded states from physical/simulated trials.

VII. SUMMARY AND DISCUSSION

Machine learning algorithms have been applied to underwater vehicle dynamic modeling to approximate their dynamic response (partially and completely) in the form of data-driven/non-parametric models, as well as to estimate the coefficients in parametric models. This section summarizes the trends found in the reviewed literature and discusses the rationale, challenges, and future directions of these trends. The reviewed literature specifically on ML applications for underwater vehicle modeling is summarized in Table 2, highlighting key aspects of each application.

A. ARTIFICIAL NEURAL NETWORKS

Early applications of ANNs for underwater vehicle modeling involved learning the coefficients for parametric models. Fully connected feed-forward ANN structures were trained offline on prerecorded data, or online using sensor readings during real-time operation. The manually predefined parametric model structure in this form of application did not fully utilize an ANN's approximation capabilities, and required larger training data sets while providing little benefit when compared to the parametric regression methods used for system identification. Comparisons show that machine learning methods for coefficient estimation had improved tolerance to outliers in training data, but otherwise produced comparable values to least-squares regression [14], [15].

The data-driven nature of ANNs demonstrated more utility in applications that replace sections of a parametric model with a trained ANN. These applications range from estimating the hydrodynamic drag function and external disturbances, to replacing the entire parametric model with a trained network to estimate the systems acceleration or velocity from prior states and control signals [15], [28], [47].

The ability of RNNs to approximate time-varying functions contrasts with the time-independent nature of feed-forward ANNs and parametric models. This would enable RNNs to capture the time-varying component of an underwater vehicle's dynamics caused by the wakes and vortices produced from prior motions. A large portion of the reviewed

TABLE 2. The reviewed literature specifically regarding the application of machine learning algorithms for underwater vehicle dynamic modeling.

ML Technique (Training Type)	Application Summary	Input(s)	Output(s)	Approximate Data set Size	Ref
ANN (Offline) SVR (Offline) GPR (Offline)	Estimate damping in sway and yaw	$\nu, \dot{\nu}, \tau$	$D(\nu)\nu$	7×10^3	[15]
ANN (Offline, Online)	Estimate inertia/gravity/damping terms and coefficients	η, ν, τ	$M, g(\eta)$, $D(\nu)\nu, C(\nu)$, extracted coefficients	Unspecified	[14, 27, 28]
RNN (Offline)	Solve dynamic equations	Control signals, past outputs	Dimensionless velocity ν'	Unspecified (184 maneuvers)	[31, 32, 33, 34, 35]
RNN (Offline)	Solve dynamic equations (with parametric actuator model)	τ, past outputs	Dimensionless velocity ν'	Unspecified (250 maneuvers)	[38, 39, 40, 41, 42, 43]
RNN (Offline)	Solve dynamic equations (with parametric actuator model)	Vehicle geometry, past outputs, τ	Dimensionless velocity	Unspecified	[44, 45]
ANN (Offline)	Estimate maneuvering characteristics	Vehicle geometry, ν, τ	Steady turn diameter	Unspecified	[46]
ANN (Offline)	Estimate propeller output loads	Velocity terms, control signals	Propeller τ	Unspecified	[47, 48]
RNN (Offline)	Estimate added resistance from waves	Vehicle geometry, disturbance parameters	Non-dimensional added resistance	Unspecified	[6]
ANN (Offline)	Estimate external disturbance forces/moments from waves/wind	η, ν	External disturbance force/moment b	Unspecified	[50]
RNN (Offline)	Estimate parametric model equations error	Parametric model $M_t\dot{\nu}_t$, past outputs, τ	Parametric model error $M\dot{\nu} - M_t\dot{\nu}_t$	Unspecified	[51]
PINN (Offline, Online)	Estimate parametric model equations error	$c, \hat{\nu}, (\nu - \hat{\nu})$	$(\nu - \hat{\nu})$	1.75×10^3	[4]
PINN (Offline, Online)	Estimate dynamic equations response	η, ν, τ	$\dot{\nu}$	2^{16}	[5]
SVR (Offline, Online)	Estimate horizontal plane dynamic equations response	η, ν, and control signals in the horizontal plane	$\dot{u}, \dot{v}, \dot{r}$	3×10^4	[8]
SVR (Offline, Online)	Estimate damping in yaw	Yaw velocity r	Yaw term of $D(\nu)\nu$	10^4	[9]
SVR (Offline, Online)	Estimate horizontal plane dynamic equations response	ν and control signals in the horizontal plane	$\dot{u}, \dot{v}, \dot{r}$	10^4	[10]
GPR (Offline)	Estimate position and velocity from dynamic response	η, ν	ν, η	4×10^3	[11]
GPR (Offline) RNN (Offline)	Estimate velocity from dynamic response	Control signals, ν	ν	4^3	[13]
SVR (Offline)	Estimate damping coefficients in the vertical plane	η, ν, τ in the vertical plane	$D(\nu)\nu$, extracted coefficients in the vertical plane	Unspecified	[70]
SVR (Offline)	Estimate combined damping and Coriolis terms	ν	$D(\nu)\nu + C(\nu)\nu$	Unspecified	[73]

literature utilizes RNNs rather than feed-forward ANNs [6], [42], [43], [45], [51].

The development of physics-informed neural networks through selecting physics-based optimization functions has

shown desirable performance in producing physically-constrained estimates for other dynamic systems, but current implementations for underwater vehicle modeling have produced estimates with lower accuracy than coefficient-based models [4], [5].

B. NON-PARAMETRIC REGRESSION

Two alternatives to using ANNs for non-parametric underwater vehicle modeling are support vector regression and Gaussian process regression, which have been applied for both estimating the hydrodynamic drag and the entire dynamic model. Compared to the ANN methods and parametric models, the added utility of these non-parametric regression models have a strong appeal, with lower training data requirements and non-parametric flexibility. The benefits of using these alternative have been demonstrated through quantitative accuracy comparisons between the modeling estimates when trained and validated on the same data sets [8], [13], [15], [73].

A clear difference between support vector regression and Gaussian process regression can be seen in the behavior of their learned functions. When using GPR, the learned function prioritizes coinciding with the training data points, while SVR results in a function which balances average error with a reduced non-linearity. GPR also has the unique property of providing the quantified uncertainty of a prediction. This has been utilized in model-based control design to optimize adaptive control terms. The lower training data requirements of both methods also makes the use of online training feasible [8], [9], [10], [14].

C. ML-BASED MODELING COMPARISONS AND RECOMMENDATIONS

As both parametric and ML-based modeling methods are developed using pre-recorded data sets, their modeling accuracy is almost always presented relative to a validation data set rather than in comparison to other modeling methods. Apart from the occasional modeling method comparison conducted to demonstrate iterative method improvements, there is a gap in the literature regarding the relative accuracy between the range of parametric and ML-based underwater vehicle modeling methods.

From the comparisons conducted in [15] and [28], it is observed that non-parametric ML-based models (using neural networks, SVR, or GPR) did produce higher accuracy models than those developed using least-squares regression for a parametric model using the same training data. In comparison, the differences in mean absolute errors for predicted forces and moments between SVR and GPR was negligible, with the neural network model being slightly less accurate.

It should be noted that this increased accuracy does require increased model training time as observed in [15], which suggests that least-squares and SVR are comparatively more computationally expensive for smaller data sets (<100 data samples), and that for larger data set (>10,000 data samples) all the ML-based regression methods are magnitudes more computationally expensive than least-squares. The Inference time for the compared methods was also analyzed, showing both least-squares and neural networks having the fastest inference and being independent of training data size. The GPR and SVR methods had longer inference times that increased with data set size. Although not included in any comparisons, PINNs have demonstrated the possibility for neural network models to train more efficiently and produce physically accurate results. The complex loss functions that would be necessary for their application with underwater vehicles currently limits their practicality.

From these comparisons, the use of SVR for producing non-parametric underwater vehicle models [8], [9], [10] can be recommended for its applicability to small data sets, its balance of accuracy and computational complexity, and its online training capabilities. It is also suggested that the dynamics model can be segmented into multiple modeling terms, such that ML-based terms do not have to approximate overly complex dynamic relationships, and that parametric modeling terms that may already be available can be utilized in addition.

D. CHALLENGES AND FUTURE RESEARCH

- *Data sets*: Details regarding a majority of the data sets used in the reviewed literature are unspecified, and the data sets themselves are rarely published. This raises issues regarding the repeatability of these works, and limits their utility in method performance comparisons between separate works. Both physical trial data and synthetic simulation data have been used to create the data sets found in the literature, each with their own challenges. System uncertainties regarding sensor accuracy and unmeasured external disturbances can degrade the dynamics learned by models that use physical trial data, while models trained on simulation data cannot demonstrate their ability to account for uncertainties outside the simulation scope (such as the time-varying dynamics excluded from parametric models). The development of physical trial data sets for underwater vehicles is costly, and the distribution of this data may be against non-academic interests, but will be necessary for the advancement of machine learning applications for underwater vehicle modeling.
- *Application validation*: The performance of the trained models is measured by error metrics relative to validation data that has the same input-output pairs as the training data. Some of the reviewed works use novel validation data, but many use very similar maneuver data for both training and validation. If the modeled underwater vehicle will be performing maneuvers that were not included in the training data, they should be well represented in the validation data to verify the trained model's suitability. For the models trained

online, there has also not been work conducted to quantify any accuracy improvements from the online learning in comparison to an equivalent model that is trained offline.

- *Hyperparameter selection*: Optimizing model hyperparameters is a difficulty shared by all machine learning applications. Though the hyperparameter values are specified in most of the reviewed literature, the selection process is only briefly mentioned, if mentioned at all. The limited validation tests conducted in these works also brings into question how applicable/effective the chosen hyperparameter values are for learning the dynamics of maneuvers that are outside the training data, and for other vehicles.
- *Online learning safeguards*: The behavior of a model that is trained offline can be validated prior to use by sampling its response to the range of possible inputs. This is not true for models that continue training online during use, which opens the possibility of unknown model responses in critical applications such as vehicle control and state estimation. The use of physics-informed neural networks have the potential to restrict the model responses to a physically-accurate domain, and has not yet been extensively investigated in the underwater vehicle literature.

VIII. CONCLUSION

This paper presents an overview of the current developments in applications of machine learning for the modeling of underwater vehicle dynamics. The classical parametric models used for underwater vehicles are introduced. The use of artificial neural networks for both partially and completely replacing parametric models were reviewed. The utility of recurrent neural networks and physics-informed neural networks for these applications was explored, including the mathematics behind their implementations. The use of non-parametric regression methods in the modeling of underwater vehicles was also reviewed, with the literature containing both offline and online learning methods. A high-level analysis of the literature was presented, discussing the trends and current challenges of applying machine learning to underwater vehicle modeling. The shift from ANNs to RNNs was noted due to their time-varying capability, as well as the recent attempts to adapt PINNs for underwater vehicle modeling. The appeal of using non-parametric regression methods was also highlighted by recent applications. We find that future research will have to address the lack of hyperparameter selection details and publicly available data sets, continue to investigate online learning capabilities and safeguards, and quantify model applicability for more varied maneuvers.

ACKNOWLEDGMENT

(Xan Macatangay and Sargon A. Gabriel are co-first authors.)

REFERENCES

[1] C. I. Sprague, Ö. Özkahraman, A. Munafo, R. Marlow, A. Phillips, and P. Ögren, "Improving the modularity of AUV control systems using behaviour trees," in *Proc. IEEE/OES Auto. Underwater Vehicle Workshop (AUV)*, Nov. 2018, pp. 1–6.

[2] G. Taraldsen, T. A. Reinen, and T. Berg, "The underwater GPS problem," in *Proc. OCEANS IEEE*, Santander, Spain, Jun. 2011, pp. 1–8.

[3] S. B. Gibson and D. J. Stilwell, "Hydrodynamic parameter estimation for autonomous underwater vehicles," *IEEE J. Ocean. Eng.*, vol. 45, no. 2, pp. 385–394, Apr. 2020.

[4] L. Lei, Y. Gang, and G. Jing, "Physics-guided neural network for underwater glider flight modeling," *Appl. Ocean Res.*, vol. 121, Apr. 2022, Art. no. 103082.

[5] B. Mirzai, "Physics-informed deep learning for system identification of autonomous underwater vehicles: A Lagrangian neural network approach," School Elect. Eng. Comput. Sci., KTH Royal Inst. Technol., Stockholm, Sweden, Tech. Rep. 2021:393, 2021. [Online]. Available: https://www.diva-portal.org/smash/record.jsf?pid=diva2%3A1592532&dswid=9184

[6] K. E. Marlantes and K. J. Maki, "A neural-corrector method for prediction of the vertical motions of a high-speed craft," *Ocean Eng.*, vol. 262, Oct. 2022, Art. no. 112300.

[7] K. Yang, W. Duan, L. Huang, P. Zhang, and S. Ma, "A prediction method for ship added resistance based on symbiosis of data-driven and physics-based models," *Ocean Eng.*, vol. 260, Sep. 2022, Art. no. 112012.

[8] B. Wehbe, M. Hildebrandt, and F. Kirchner, "A framework for on-line learning of underwater vehicles dynamic models," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 7969–7975.

[9] B. Wehbe, A. Fabisch, and M. M. Krell, "Online model identification for underwater vehicles through incremental support vector regression," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2017, pp. 4173–4180.

[10] B. Wehbe and M. M. Krell, "Learning coupled dynamic models of underwater vehicles using support vector regression," in *Proc. OCEANS-Aberdeen*, Jun. 2017, pp. 1–7.

[11] W. A. Ramirez, Z. Q. Leong, H. Nguyen, and S. G. Jayasinghe, "Position estimation for underwater vehicles using unscented Kalman filter with Gaussian process prediction," *Underwater Technol.*, vol. 36, no. 2, pp. 29–35, 2019.

[12] W. Ariza Ramirez, Z. Q. Leong, H. Nguyen, and S. G. Jayasinghe, "Non-parametric dynamic system identification of ships using multi-output Gaussian processes," *Ocean Eng.*, vol. 166, pp. 26–36, Oct. 2018.

[13] W. A. Ramirez, J. Kocijan, Z. Q. Leong, H. D. Nguyen, and S. G. Jayasinghe, "Dynamic system identification of underwater vehicles using multi-output Gaussian processes," *Int. J. Autom. Comput.*, vol. 18, no. 5, pp. 681–693, Jul. 2021, doi: 10.1007/s11633-021-1308-x.

[14] F. Ahmed, X. Xiang, C. Jiang, G. Xiang, and S. Yang, "Survey on traditional and AI based estimation techniques for hydrodynamic coefficients of autonomous underwater vehicle," *Ocean Eng.*, vol. 268, Jan. 2023, Art. no. 113300.

[15] B. Wehbe, M. Hildebrandt, and F. Kirchner, "Experimental evaluation of various machine learning regression methods for model identification of autonomous underwater vehicles," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2017, pp. 4885–4890.

[16] M. Renilson, *Submarine Hydrodynamics*. Cham, Switzerland: Springer, 2018.

[17] T. I. Fossen, "Mathematical models of ships and underwater vehicles," in *Encyclopedia of systems and control*. London, U.K.: Springer 2021.

[18] T. I. Fossen, *Handbook of Marine Craft Hydrodynamics and Motion Control*. Chichester, U.K.: Wiley, 2011.

[19] B. R. McCarter, "Experimental evaluation of viscous hydrodynamic force models for autonomous underwater vehicles," Ph.D. dissertation, Dept. Elect. Comput. Eng., Virginia Tech, Blacksburg, VA, USA, 2014.

[20] M. Gertler and G. R. Hagen, "Standard equations of motion for submarine simulation," David W Taylor Naval Ship Res. Develop. Center, Bethesda, Maryland, Tech. Rep., 1967. [Online]. Available: https://api.semanticscholar.org/CorpusID:109295468

[21] J. Feldman, "DTNSRDC revised standard submarine equations of motion," David W Taylor Naval Ship Res. Develop. Center, Bethesda, Maryland, Tech. Rep., 1979. [Online]. Available: https://ntrl.ntis.gov/NTRL/dashboard/searchResults/titleDetail/ADA071804.xhtml

[22] M. E. Kepler, "Dynamics of a small autonomous underwater vehicle that tows a large payload," M.S. thesis, Elect. Eng., Virginia Polytech. Inst. State Univ., Blacksburg, VA, USA, 2018.

[23] F. Ahmed, X. Xiang, H. Wang, J. Zhang, G. Xiang, and S. Yang, "Nonlinear dynamics of novel flight-style autonomous underwater vehicle with bow wings, Part I: ASE and CFD based estimations of hydrodynamic coefficients, Part II: Nonlinear dynamic modeling and experimental validations," *Appl. Ocean Res.*, vol. 141, Dec. 2023, Art. no. 103739.

[24] J. Tobias Springenberg, "Unsupervised and semi-supervised learning with categorical generative adversarial networks," 2015, *arXiv:1511.06390*.

[25] D. J. Miller and H. Uyar, "A mixture of experts classifier with learning based on both labelled and unlabelled data," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 9, 1996, pp. 1–7.

[26] J. P. Panda, A. Mitra, and H. V. Warrior, "A review on the hydrodynamic characteristics of autonomous underwater vehicles," *Proc. Inst. Mech. Eng., M, J. Eng. Maritime Environ.*, vol. 235, no. 1, pp. 15–29, 2021.

[27] Z. Xing and L. Mccue, "Modeling ship equations of roll motion using neural networks," *Nav. Eng. J.*, vol. 122, no. 3, pp. 49–60, Sep. 2010.

[28] P. W. Van de Ven, T. A. Johansen, A. J. Sørensen, C. Flanagan, and D. Toal, "Neural network augmented identification of underwater vehicle models," *Control Eng. Pract.*, vol. 15, no. 6, pp. 715–725, Jun. 2007.

[29] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Netw.*, vol. 2, no. 5, pp. 359–366, Jan. 1989.

[30] M. Kaur and A. Mohta, "A review of deep learning with recurrent neural network," in *Proc. Int. Conf. Smart Syst. Inventive Technol. (ICSSIT)*, Nov. 2019, pp. 460–465.

[31] W. E. Faller, W. E. Smith, R. T. Nigon, and T. T. Huang, "Six degree-of-freedom maneuvering simulation of an experimental model undergoing severe maneuvers using recursive neural networks," in *Proc. 14th Appl. Aerodyn. Conf.*, 1996, pp. 895–905.

[32] W. E. Faller, W. E. Smith, and T. T. Huang, "Applied dynamic system modeling: Six degree-of-freedom simulation of forced unsteady maneuvers using recursive neural networks," in *35th Aerosp. Sci. Meeting Exhibit*, Jan. 1997, p. 336.

[33] W. E. Faller, "Recursive neural networks: Toward advances in simulation and control," in *Fluids Conf. Exhibit*, Jun. 2000, p. 2470.

[34] W. E. Faller, D. E. Hess, W. E. Smith, and T. T. Huang, "Recursive neural networks: Six degree-of-freedom maneuvering simulation capabilities," in *Proc. 16th AIAA Appl. Aerodyn. Conf.*, 1998, pp. 556–564.

[35] W. Faller, D. Hess, W. Smith, and T. Huang, *Applications of Recursive Neural Network Technologies to Hydrodynamics*. Washington, DC, USA: National Academies Press, Feb. 1999.

[36] L. Moreira and C. G. Soares, "Application of neural networks to model catamaran manoeuvres," in *Proc. Int. Conf. High Perform. Mar. Vessels (HSMV 2011)*, 2011, pp. 1–9.

[37] L. Moreira and C. Guedes Soares, "H_2 and H_∞ designs for diving and course control of an autonomous underwater vehicle in presence of waves," *IEEE J. Ocean. Eng.*, vol. 33, no. 2, pp. 69–88, Apr. 2008.

[38] D. Hess and W. Faller, "Simulation of ship maneuvers using recursive neural networks," in *Proc. 23rd Symp. Nav. Hydrodyn.* Val de Reuil, France: National Academies of Sciences, 2001, pp. 223–242.

[39] D. Hess and W. Faller, "Using recursive neural networks for blind predictions of submarine manoeuvres," in *Proc. 24th Symp. Nav. Hydrodyn.*, Fukuoka, Japan, 2002, pp. 719–737.

[40] H. F. Hashem, "Submarine maneuvers prediction using recursive neural networks," in *Proc. 8th Seminar Neural Netw. Appl. Electr. Eng.*, 2006, pp. 73–77.

[41] D. E. Hess, W. E. Faller, L. Minnick, and T. C. Fu, "Maneuvering simulation of Sea Fighter using a fast nonlinear time domain technique," in *Proc. 9th Int. Conf. Numer. Ship Hydrodyn. (NSH)*, 2007.

[42] D. E. Hess, W. E. Faller, J. Lee, and J. W. Broncheau, "Simulation of a non-axisymmetric undersea vehicle using a recursive neural network," in *Proc. 26th Congr. Int. Council Aeronaut. Sci. (ICAS)*, vol. 4, 2008, pp. 1909–1919.

[43] F.-C. Chiu, T.-L. Chang, J. Go, S.-K. Chou, and W.-C. Chen, "A recursive neural networks model for ship maneuverability prediction," in *Proc. MTS/IEEE Techno-Ocean*, vol. 3, Nov. 2004, pp. 1211–1218.

[44] W. Faller, D. Hess, and T. Fu, "Simulation based design: A real-time approach using recursive neural networks," in *Proc. 43rd AIAA Aerosp. Sci. Meeting Exhibit-Meeting Papers*, 2005, pp. 14649–14661.

[45] W. Faller, D. Hess, and T. Fu, "Real-time simulation based design Part II: Changes in hull geometry," *Proc. 44th AIAA Aerosp. Sci. Meeting*, vol. 23, 2006, pp. 17801–17816.

[46] P. T. Martins and V. Lobo, "Estimating maneuvering and seakeeping characteristics with neural networks," in *Proc. OCEANS*, Jun. 2007, pp. 1–5.

[47] D. E. Hess, W. E. Faller, R. F. Roddy, A. M. Pence, and T. C. Fu, "Feedforward neural networks applied to problems in ocean engineering," in *Proc. 25th Int. Conf. Offshore Mech. Arctic Eng.*, 2006, pp. 501–510.

[48] R. F. Roddy, D. E. Hess, and W. Faller, "Utilizing neural networks to predict forces and moments on a submarine propeller," in *Proc. 46th AIAA Aerosp. Sci. Meeting Exhibit*, 2008, pp. 1–17.

[49] H. Gong, M. J. Er, Y. Liu, and C. Ma, "Three-dimensional optimal trajectory tracking control of underactuated AUVs with uncertain dynamics and input saturation," *Ocean Eng.*, vol. 298, Apr. 2024, Art. no. 116757.

[50] D. E. Hess, W. E. Faller, T. C. Fu, and E. S. Ammeen, "Improved simulation of ship maneuvers using recursive neural networks," in *Proc. 44th AIAA Aerosp. Sci. Meeting*, vol. 23, 2006, pp. 17744–17763.

[51] W. Faller, K. Junghans, J. L. Lewis, and D. E. Hess, "Synergistic computing: Combining CFD and neural networks for maneuvering simulation," in *Proc. 48th AIAA Aerosp. Sci. Meeting Including New Horizons Forum Aerosp. Expo.*, 2010, pp. 1–10.

[52] C. Feng, S. Gao, S. Chen, Z. Gao, and C. Grebogi, "Classifying motion states of AUV based on graph representation for multivariate time series," *Ocean Eng.*, vol. 268, Jan. 2023, Art. no. 113539.

[53] P.-F. Xu, C.-B. Han, H.-X. Cheng, C. Cheng, and T. Ge, "A physics-informed neural network for the prediction of unmanned surface vehicle dynamics," *J. Mar. Sci. Eng.*, vol. 10, no. 2, p. 148, Jan. 2022.

[54] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *J. Comput. Phys.*, vol. 378, pp. 686–707, Feb. 2019.

[55] G. Karniadakis, Y. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang, "Physics-informed machine learning," *Nature Rev. Phys.*, vol. 3, no. 6, pp. 422–440, May 2021.

[56] G. Cook and F. Zhang, *Mobile Robots: Navigation, Control and Sensing, Surface Robots and AUVs*. Hoboken, NJ, USA: Wiley, 2020.

[57] S. Zhang, J. Yu, A. Zhang, and F. Zhang, "Spiraling motion of underwater gliders: Modeling, analysis, and experimental results," *Ocean Eng.*, vol. 60, pp. 1–13, Mar. 2013.

[58] D. Liu and Y. Wang, "Multi-fidelity physics-constrained neural network and its application in materials modeling," *J. Mech. Des.*, vol. 141, no. 12, 2019, Art. no. 121403.

[59] E. Kharazmi, Z. Zhang, and G. E. M. Karniadakis, "Hp-VPINNs: Variational physics-informed neural networks with domain decomposition," *Comput. Methods Appl. Mech. Eng.*, vol. 374, Feb. 2021, Art. no. 113547.

[60] A. D. Jagtap, E. Kharazmi, and G. E. Karniadakis, "Conservative physics-informed neural networks on discrete domains for conservation laws: Applications to forward and inverse problems," *Comput. Methods Appl. Mech. Eng.*, vol. 365, Jun. 2020, Art. no. 113028.

[61] M. Cranmer, S. Greydanus, S. Hoyer, P. Battaglia, D. Spergel, and S. Ho, "Lagrangian neural networks," 2020, *arXiv:2003.04630*.

[62] M. Lutter, C. Ritter, and J. Peters, "Deep Lagrangian networks: Using physics as model prior for deep learning," 2019, *arXiv:1907.04490*.

[63] T. Fossen and A. Ross, "Nonlinear modelling, identification and control of UUVs," *IEE Control Eng.*, vol. 69, p. 13, Jan. 2006.

[64] D. Bzdok, N. Altman, and M. Krzywinski, "Statistics versus machine learning," *Nature Methods*, vol. 15, no. 4, pp. 233–234, Apr. 2018.

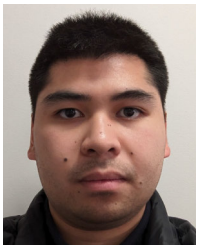
[65] T. M. (2022). *Understanding Support Ector Machine Regression*. Natick, MA, USA. [Online]. Available: https://au.mathworks.com/help/stats/understanding-support-vector-machine-regression.html

[66] Z. Wang, Z. Zou, and C. Guedes Soares, "Identification of ship manoeuvring motion based on nu-support vector machine," *Ocean Eng.*, vol. 183, pp. 270–281, Jul. 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0029801819302112

[67] Z. Wang, H. Xu, L. Xia, Z. Zou, and C. G. Soares, "Kernel-based support vector regression for nonparametric modeling of ship maneuvering motion," *Ocean Eng.*, vol. 216, Nov. 2020, Art. no. 107994, doi: 10.1016/j.oceaneng.2020.107994.

[68] J. Wang, "An intuitive tutorial to Gaussian process regression," *Comput. Sci. Eng.*, vol. 25, no. 4, pp. 4–11, Jul. 2023, doi: 10.1109/mcse.2023.3342149.

[69] Z.-L. Ouyang, Z.-J. Zou, and L. Zou, "Adaptive hybrid-kernel function based Gaussian process regression for nonparametric modeling of ship maneuvering motion," *Ocean Eng.*, vol. 268, Jan. 2023, Art. no. 113373, doi: 10.1016/j.oceaneng.2022.113373.
[70] F. Xu, Z.-J. Zou, J.-C. Yin, and J. Cao, "Parametric identification and sensitivity analysis for autonomous underwater vehicles in diving plane," *J. Hydrodynamics*, vol. 24, no. 5, pp. 744–751, Oct. 2012.
[71] W. Luo, L. Moreira, and C. Guedes Soares, "Manoeuvring simulation of catamaran by using implicit models based on support vector machines," *Ocean Eng.*, vol. 82, pp. 150–159, May 2014.
[72] X.-G. Zhang and Z.-J. Zou, "Estimation of the hydrodynamic coefficients from captive model test results by using support vector machines," *Ocean Eng.*, vol. 73, pp. 25–31, Nov. 2013.
[73] F. Xu, Z.-J. Zou, J.-C. Yin, and J. Cao, "Identification modeling of underwater vehicles' nonlinear dynamics based on support vector machines," *Ocean Eng.*, vol. 67, pp. 68–76, Jul. 2013.
[74] Y. Chen, W. Wang, and G. Xu, "System identification of AUV hydrodynamic model based on support vector machine," in *Proc. IEEE 7th Int. Conf. Underwater Syst. Technol., Theory Appl. (USYS)*, Dec. 2017, pp. 1–7.



XAN MACATANGAY received the B.Eng. degree (Hons.) in advanced manufacturing and mechatronics engineering from RMIT University, Australia, in 2018, where he is currently pursuing the Ph.D. degree in the robust and fault tolerant control of unmanned underwater vehicles. His research interests include the guidance, navigation, and control of autonomous systems, nonlinear system modeling and control, as well as machine learning for sensing and control applications.

SARGON A. GABRIEL received the B.Eng. degree in aerospace engineering, the B.Sc. degree in applied mathematics, and the Ph.D. degree in applied mathematics from RMIT University. He is currently a researcher in the medical devices industry. His current research interests lie in the modeling of coupled multiphysics systems and the design of medical devices.

REZA HOSEINNEZHAD received the B.Sc. degree in electronic engineering, the M.Sc. degree in control engineering, and the Ph.D. degree in robotics from the University of Tehran, in 1994, 1996, and 2021, respectively. He has held various academic positions with the University of Tehran, Swinburne University of Technology, The University of Melbourne, and RMIT University. He is currently a Professor in autonomous systems and the Associate Dean of Mechanical and Automotive Engineering with the School of Engineering, RMIT University. His current research interests include multi-object systems and random finite set filters, with an emphasis on their applications to situational awareness for autonomous systems.

ANTHONY FOWLER (Member, IEEE) received the bachelor's and Ph.D. degrees in electrical engineering from The University of Newcastle, Callaghan, NSW, Australia, in 2010 and 2014, respectively. He is currently with the Defence Science and Technology Group, Melbourne, VIC, Australia. His current research interests include the modeling and control of underwater vehicles.

ALIREZA BAB-HADIASHAR (Senior Member, IEEE) received the B.Sc. degree from the University of Tehran, the M.Eng. degree from The University of Sydney, and the Ph.D. degree from Monash University, Australia. He has held various academic positions with Monash University, Swinburne University of Technology, and Copenhagen University. Currently, he is a Professor with RMIT University, where he leads the Intelligent Automation Research Group. He is an expert in the use of robust statistics for visual motion estimation and has a strong track record in developing robust vision-based industrial automation solutions.