robotics

MDPI

Article

A Novel Methodology for Designing Digital Models for Mobile Robots Based on Model-Following Simulation in Virtual Environments

Brayan Saldarriaga-Mesa [1,2], José Varela-Aldás [3,*], Flavio Roberti [1] and Juan M. Toibero [1]

[1] Institute of Automation (INAUT), National University of San Juan (UNSJ), San Juan 5400, Argentina; bsaldarriaga@inaut.unsj.edu.ar (B.S.-M.); froberti@inaut.unsj.edu.ar (F.R.); mtoibero@inaut.unsj.edu.ar (J.M.T.)
[2] Facultad de Ciencias Básicas e Ingeniería, Universidad de los Llanos, Villavicencio 500017, Colombia
[3] Centro de Investigación MIST, Facultad de Ingenierías, Universidad Tecnológica Indoamérica, Ambato 180103, Ecuador
* Correspondence: josevarela@uti.edu.ec

Abstract

Virtual environment simulations have gained great importance in the field of robotics by enabling the validation and optimization of control algorithms before their implementation on real platforms. However, the construction of accurate digital models is limited not only by the lack of detailed characterization of the components but also by the uncertainty introduced by the physics engine and the plugins used in the simulation. Unlike other works, which attempted to model each element of the robot in detail and rely on the physics engine to reproduce its behavior, this paper proposes a methodology based on model following. The proposed architecture forces the simulated robot to replicate the dynamics of the real robot without requiring exactly the same physical parameters. The experimental validation was carried out on two unmanned surface vehicle (USV) platforms with different dynamic parameters and, therefore, different responses to excitation signals, demonstrating that the proposed approach enables a drastic reduction in error. In particular, RMSE and MAE were reduced by more than 98%, with R^2 values close to 1.0, demonstrating an almost perfect correspondence between the real and simulated dynamics.

Keywords: digital models; modeling and control systems; mobile robots; design methodology

check for updates

Academic Editor: Beno Benhabib

Received: 20 May 2025
Revised: 28 August 2025
Accepted: 30 August 2025
Published: 2 September 2025

Citation: Saldarriaga-Mesa, B.; Varela-Aldás, J.; Roberti, F.; Toibero, J.M. A Novel Methodology for Designing Digital Models for Mobile Robots Based on Model-Following Simulation in Virtual Environments. *Robotics* **2025**, *14*, 124. https://doi.org/10.3390/robotics14090124

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The development of autonomous and teleoperated mobile robots has recently accelerated, driven by advances in artificial intelligence, sensors, and control systems. Ensuring their safe and efficient operation in real environments requires reliable simulation tools that enable validation and optimization prior to deployment.

In this context, Digital Models have emerged as virtual representations that comprehensively capture the geometric, physical, and dynamic characteristics of real systems. Unlike generic simulations, Digital Models integrate parameters from analytical and experimental sources to closely replicate system behavior within physics-based simulation environments.

The concept of a Digital Model is interpreted in various ways, largely depending on the context of its application. Although it is commonly described as a virtual counterpart of a physical entity, the term is often used ambiguously, leading to different understandings and confusions. For example, some researchers do not differentiate between Digital Models

such as three-dimensional representations and Digital Twins, treating these concepts as equivalent. Several works have addressed the categorization of simulation types [1–4]. Table 1 shows a summary of these levels.

Table 1. Simulation levels and their characteristics in robotics simulation environments.

Simulation Level	Description
Numerical representation	Mathematical models implemented in tools like MATLAB/Simulink, describing dynamics through equations without spatial rendering or physics.
Digital Mockup	Geometric visualization using mesh files (e.g., `.STL`) for appearance and layout, without physics-based interaction.
Structural Model	Robot structure with links, joints, and kinematics (URDF/SDF), enabling collision detection and articulation without dynamic simulation.
Software-in-the-Loop (SITL)	Simulation including actual control software running in a virtual environment that emulates robot sensors and actuators, enabling the testing of software logic without hardware.
Hardware-in-the-Loop (HITL)	Integration of physical hardware components with simulation; the controller runs on the real robot hardware (plant), while sensors and actuators are simulated.
Digital Model	Physics-based model with calibrated geometric and dynamic properties, not synchronized in real time with the physical system.
Digital Shadow	Real-time digital representation updated from physical sensor data, without feedback to the real system.
Digital Twin	Real-time bidirectional replica of the physical system, enabling monitoring, control, and predictive optimization.

1.1. Related Works

This subsection reviews digital models developed for different types of robotic platforms, outlining representative examples and then summarizing the main methodologies employed for their development.

In the literature, the term digital twin is often misused to describe only a geometric representation of the robot. In reality, such models correspond to digital mockups without the ability to reproduce dynamic behavior. A true digital twin requires bidirectional synchronization with the real robot, ensuring continuous two-way data exchange. Thus, simulation environments that model a real robot but lack any communication in the loop are, in fact, digital models rather than digital twins.

Several digital models have been developed to represent mobile robots across different domains, including Unmanned Aerial Vehicles (UAVs), Unmanned Ground Vehicles (UGVs), Unmanned Surface Vehicles (USVs), and Unmanned Underwater Vehicles (UUVs). For UAVs, platforms such as the DJI Phantom series and Parrot drones have well-documented dynamic models. DJI's SDK and simulation tools enable detailed flight dynamics and sensor simulation [5–7]. Similarly, Parrot offers simulation environments that replicate vehicle behavior for control development and testing [8,9]. Open source autopilot projects like PX4 provide highly detailed digital models integrated into simulators such as Gazebo and JMAVSim, which incorporate aerodynamic effects, sensor models, and actuator dynamics [10–13].

In the realm of UGVs, Clearpath Robotics' Husky is a widely used platform with accurate kinematic and dynamic digital models designed for ROS-based simulation and development [14,15]. The TurtleBot series is another prominent platform, offering simplified but versatile digital models for indoor navigation, SLAM, and robotics research [10,16].

A review of existing USV simulators highlights these limitations. For instance, PX4's SITL simulator for watercraft provided only a basic environment until it was discontinued in 2023, mainly due to its reliance on a ground vehicle model with Ackermann steering, which is unsuitable for marine dynamics [17]. Webots introduced a Heron USV simulator

in 2025, but it uses a simplistic linear mapping for its thrusters and lacks fluid dynamics modeling, making the system unstable when scaled to faster or heavier vehicles [18]. In contrast, VRX is one of the most advanced simulators, incorporating models of waves, wind, and dynamic thruster response [19–24]. However, it is restricted to the WAM-V catamaran platform developed by the same organization.

Regarding the methodologies for developing Digital Models, several studies employ neural networks or deep learning to identify the robot's dynamics and then integrate these models into virtual environments equipped with plugins that emulate sensors, actuators, and external conditions [25–29]. Nevertheless, the simulated behavior can vary considerably depending on the physics engine. For instance, Gazebo 11 relies on the Open Dynamics Engine (ODE), which numerically solves differential equations to approximate system dynamics, while the Gazebo Garden adopts the PhysX engine, originally designed for video games. Consequently, the accuracy of a Digital Model depends not only on the identification methodology but also on the underlying physics engine, the configuration of its plugins, and the level of detail used in constructing the robot's virtual model.

In addition, such simulation environments require the precise characterization of the devices (e.g., mass, inertia, and density) to reproduce realistic behavior. However, many robotic prototypes are built with components whose physical and dynamic properties are not accurately documented by manufacturers. This lack of reliable characterization prevents the development of truly representative digital models. In practice, researchers often validate their controllers using commercial simulators designed for platforms of a similar type to their own systems. These simulators may share the same dynamic structure; nevertheless, the actual dynamic parameters often differ considerably. Consequently, the behavior of the simulated system may diverge from that of the real robot, leading to unrepresentative simulations.

1.2. Main Contributions

This paper proposes a novel methodology based on a "model following" simulation approach. This technique involves dual system identification: one applied to the real robot and the other to its simulated counterpart in a high-fidelity virtual environment, both using the same structural model. The central idea is not to replicate exact physical parameters, but to ensure that the simulated robot behaviorally matches the real system. This behavioral matching leads to the creation of dynamic digital models that remain valid despite uncertainties, providing a practical and robust alternative for researchers and developers working under resource constraints.

This proposed technique differs from the existing literature in that it does not create a digital model from scratch; instead, it uses an existing one and forces it to behave like the real platform. This approach provides an additional level of realism that is independent of the plugins and the physics engine used.

1.3. Paper Outline

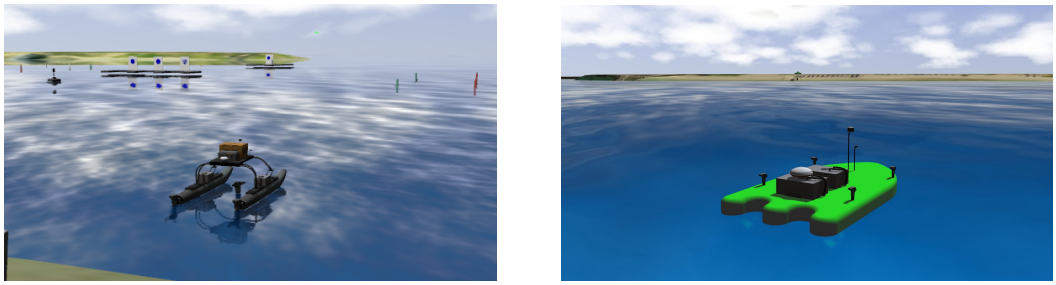
The remainder of this paper is organized as follows. Section 2 introduces the materials and methods, including the platforms and simulation environment. Section 3 presents the system identification process, where Section LQS-Based Parameter Estimation details the LQS-based parameter estimation. Section 4 describes the model-following strategy, with Section Proposed Closed-Loop Model Following Architecture outlining the proposed closed-loop model-following architecture. Section 5 reports the results, with Section 5.1 providing the analysis of results and Section 5.2 presenting future work. Finally, Section 6 concludes the paper, and Appendix A summarizes the excitation signals.

2. Materials and Methods

This study evaluates the proposed simulation methodology through two distinct experimental systems. The first is the widely adopted WAM-V, a catamaran-style system featured in the Virtual RobotX competition environment. The second is the IAcquaBot, a low-cost differential USV developed by the Universidad Nacional de San Juan. These two platforms represent contrasting levels of mechanical complexity and modeling fidelity, and thus provide a meaningful basis for testing the adaptability of the proposed Digital Model strategy.

The IAcquaBot is a compact, single-body USV designed for low-cost autonomous operation in calm or semi-structured aquatic environments. It has a rigid hull with four fixed thrusters in a differential configuration: two on the front and two on the rear. Unlike the WAM-V (Figure 1a), IAcquaBot lacks mechanical suspension or modularity (Figure 1b), making it structurally simpler but more susceptible to environmental disturbances. The main features of the WAM-V and IAcquaBot platforms are summarized in Table 2.

The simulation environment requires the installation of several pre-requisites such as Ubuntu 20.04, ROS Noetic, and Gazebo 11.



(**a**) WAM-V (VRX) simulation environment (**b**) IAcquaBot simulation environment

Figure 1. Simulation environments used for this study: (**a**) WAM-V platform running in the VRX simulation framework; and (**b**) IAcquaBot platform simulated using Gazebo-based simulation framework.

Table 2. Comparison between WAM-V and IAcquaBot platforms.

Feature	**WAM-V**	**IAcquaBot**
Length [m]	4.9	1.5
Width [m]	2.3	0.67
Mass [kg]	180	21
Hull Type	Modular with mechanical suspension	Single rigid body, flat-bottomed
Thruster coordinates (body frame) [m]	Left: $(-0.7, +0.7)$; Right: $(-0.7, -0.7)$	Top Left: $(+0.5, +0.335)$; Top Right: $(+0.5, -0.335)$; Back Left: $(-0.5, +0.335)$; Back Right: $(-0.5, -0.335)$
Max thrust per thruster [N]	$+90$	$+36.4$
Min thrust per thruster [N]	-85	-28.4

Dynamic Model Formulation Based on Fossen's Approach

In this study, we consider the dynamic model in the body frame (see Figure 2). The relationship between the state variables is described by the following equation:

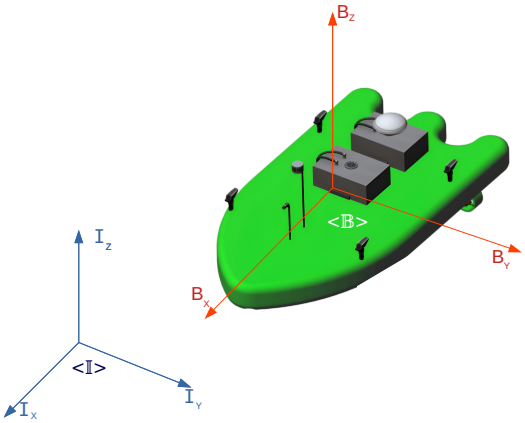


Figure 2. Inertial and body-fixed reference frames for the IAcquaBot.

$$\dot{\eta} = \mathbf{J}(\psi)\nu, \tag{1}$$

$$\mathbf{J}(\psi) = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2}$$

where $\dot{\eta}$ represents the linear and angular velocities of the robot in the inertial frame, and $\nu = \begin{bmatrix} u & vs. & r \end{bmatrix}^T$ denotes the velocity vector in the body-fixed frame, where u and $vs.$ are the surge and sway linear velocities, respectively, and r is the yaw angular velocity.

Finally, we obtain the following dynamic equation:

$$\mathbf{M}\dot{\boldsymbol{\nu}} + \mathbf{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{D}(\boldsymbol{\nu})\boldsymbol{\nu} = \boldsymbol{\tau} \tag{3}$$

From this framework, Fossen [30] proposes a structured representation of the dynamic model, where the matrix $\mathbf{M}$ and its components are given by

$$\mathbf{M} = \begin{bmatrix} m - X_{\dot{u}} & 0 & 0 \\ 0 & m - Y_{\dot{v}} & 0 \\ 0 & 0 & I_z - N_{\dot{r}} \end{bmatrix}, \tag{4}$$

This matrix represents the system's inertia and added mass, where m is the mass of the vehicle, and A_{ij} are the added mass coefficients.

The Coriolis-centripetal matrix $\mathbf{C}(\boldsymbol{\nu})$ is expressed as

$$\mathbf{C}(\boldsymbol{\nu}) = \begin{bmatrix} 0 & 0 & -(m - Y_{\dot{v}})vs. \\ 0 & 0 & (m - X_{\dot{u}})u \\ (m - Y_{\dot{v}})vs. & -(m - X_{\dot{u}})u & 0 \end{bmatrix}, \tag{5}$$

The damping matrix $\mathbf{D}(\boldsymbol{\nu})$ is defined as

$$\mathbf{D}(\boldsymbol{\nu}) = \begin{bmatrix} -X_u - X_{uu}|u| & 0 & 0 \\ 0 & -Y_v - Y_{vv}|v| & 0 \\ 0 & 0 & -N_r - N_{rr}|r| \end{bmatrix}, \tag{6}$$

Finally, the generalized force vector $\boldsymbol{\tau}$ is given by

$$\boldsymbol{\tau} = \begin{bmatrix} T_u \\ T_v \\ T_r \end{bmatrix} \tag{7}$$

where m is the vehicle mass, I_z is the yaw moment of inertia, and $X_{\dot{u}}, Y_{\dot{v}}, N_{\dot{r}}$ are the added mass terms. The linear damping coefficients are X_u, Y_v, N_r, while X_{uu}, Y_{vv}, N_{rr} represent the quadratic damping terms. The generalized input vector is $\boldsymbol{\tau} = [T_u, T_v, T_r]^\top$, where T_u, T_v, and T_r are the surge force, sway force, and yaw moment, respectively.

3. System Identification

The system identification (see Figure 3) begins with the experiment design, generating excitation signals for each thruster to stimulate the USV and recording its response. These signals include sinusoidal, square, mixed, constant, and random functions. For each time instant k, the rise time (T_r) and ultimate time (T_u) were calculated. The signals generated for model identification and validation can be found in Appendix A.

This data must be stored and subsequently filtered and processed to remove noise. A mathematical model of the USV is then calculated using identification techniques with minimum error, in this case, using least squares.

The calculated model is validated through experiments by comparing the model's predictions with actual data. Based on the validation results, the model is calibrated to improve its accuracy, which may involve adjusting the model parameters or modifying the model structure.

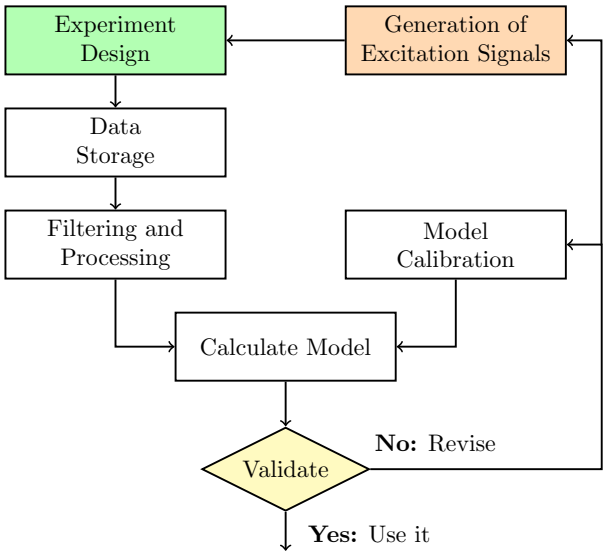


Figure 3. System identification process for USV.

LQS-Based Parameter Estimation

The dynamic equations can be expressed in a regression form as

$$\boldsymbol{\Phi}(\dot{\boldsymbol{\nu}}, \boldsymbol{\nu})\, \boldsymbol{\delta} \;=\; \boldsymbol{\tau}, \tag{8}$$

where $\boldsymbol{\Phi}(\dot{\boldsymbol{\nu}}, \boldsymbol{\nu})$ is the regressor matrix, $\boldsymbol{\delta}$ is the vector of parameters to be identified (see Table 3), and $\boldsymbol{\tau}$ is the generalized force vector. This formulation can be written as

$$\boldsymbol{\Phi}(\dot{\boldsymbol{\nu}}, \boldsymbol{\nu}) = \begin{bmatrix} \boldsymbol{\phi}_1 \\ \boldsymbol{\phi}_2 \\ \vdots \\ \boldsymbol{\phi}_m \end{bmatrix}, \quad \boldsymbol{\tau} = \begin{bmatrix} \tau_1 \\ \tau_2 \\ \vdots \\ \tau_m \end{bmatrix}, \tag{9}$$

where each row block $\boldsymbol{\phi}_i$ corresponds to the i-th measurement and is given by

$$\boldsymbol{\phi}_i = \begin{bmatrix} \dot{u} & -vr & 0 & u & |u|u & 0 & 0 & 0 & 0 \\ ur & \dot{v} & 0 & 0 & 0 & vs. & |v|v & 0 & 0 \\ 0 & 0 & \dot{r} & 0 & 0 & 0 & 0 & r & |r|r \end{bmatrix}_{(i)}. \tag{10}$$

The unknown parameter vector $\boldsymbol{\delta}$ is obtained through the least-squares solution:

$$\boldsymbol{\delta} = \left(\boldsymbol{\Phi}^T\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^T\boldsymbol{\tau} = \boldsymbol{\Phi}^+\boldsymbol{\tau}, \tag{11}$$

where $\boldsymbol{\Phi}^+$ denotes the Moore–Penrose pseudoinverse of the regressor matrix.

Table 3. Estimated parameters.

$\delta_1 = m - X_{\dot{u}}$	$\delta_2 = m - Y_{\dot{v}}$	$\delta_3 = I_z - N_{\dot{r}}$
$\delta_4 = X_u$	$\delta_5 = X_{uu}$	$\delta_6 = Y_v$
$\delta_7 = Y_{vv}$	$\delta_8 = N_r$	$\delta_9 = N_{rr}$

4. Model-Following Strategy

Once both systems have been identified, the objective is to minimize the velocity response error between the reference robot and its Digital Model. The guiding principle is that, if both systems exhibit the same velocity response to a given excitation signal, then the simulation is dynamically equivalent.

In conventional simulation workflows, control inputs, such as generalized torques (T_u, T_r), are directly fed into the plant, and the resulting linear and angular velocities (u, v, r) are observed. This process assumes that the physical model accurately represents the real system. The flow can be expressed as shown in Figure 4.

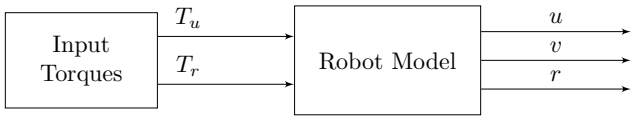


Figure 4. Conventional simulation workflow.

Proposed Closed-Loop Model Following Architecture

Let the identified dynamic models be

$$M_r\dot{\boldsymbol{\nu}}_r + C_r(\boldsymbol{\nu}_r)\boldsymbol{\nu}_r + D_r(\boldsymbol{\nu}_r)\boldsymbol{\nu}_r = \boldsymbol{\tau}_r \quad \text{and} \quad M_s\dot{\boldsymbol{\nu}}_s + C_s(\boldsymbol{\nu}_s)\boldsymbol{\nu}_s + D_s(\boldsymbol{\nu}_s)\boldsymbol{\nu}_s = \boldsymbol{\tau}_c$$

where the subscript r refers to the components describing the reference robot and the subscript s refers to the simulated robot. $\boldsymbol{\tau}_r$ are the excitation torques applied to the system to be emulated, $\boldsymbol{\nu}_r$ are the velocities predicted by the reference model, $\boldsymbol{\tau}_c$ are the torques resulting from the dynamic velocity controller, and $\boldsymbol{\nu}_s$ are the velocities obtained by the simulated robot.

The proposed methodology is a closed-loop velocity control scheme in which the reference system provides target velocities to be followed by the simulated model. The complete flow is represented as shown in Figure 5.

The error between the reference and simulated robot velocities is calculated as

$$e = \boldsymbol{\nu}_r - \boldsymbol{\nu}_s \tag{12}$$

Next, the function Gamma, which includes the velocities error and its derivatives, is defined as

$$\gamma \;=\; \dot{\nu}_r + k_d \dot{e} \tag{13}$$

where k_d is a gain factor. Finally, the control torque $\boldsymbol{\tau}_c$ is calculated using the inverse dynamics model of the simulated robot [31]:

$$\boldsymbol{\tau}_c \;=\; M_s \gamma + C_s \boldsymbol{\nu}_s + D_s \boldsymbol{\nu}_s \tag{14}$$

In this closed-loop system, the dynamics of the error are given by

$$\dot{\boldsymbol{\nu}}_s \;=\; M_s^{-1}(M_s \gamma + C_s(\boldsymbol{\nu}_s)\boldsymbol{\nu}_s + D_s(\boldsymbol{\nu}_s)\boldsymbol{\nu}_s - C_s(\boldsymbol{\nu}_s)\boldsymbol{\nu}_s - D_s(\boldsymbol{\nu}_s)\boldsymbol{\nu}_s) \tag{15}$$

In the limit as $t \to \infty$, the velocity error tends to zero, implying that $\dot{\nu}_s = \dot{\nu}_r$, meaning that the simulated robot accurately tracks the reference robot's velocities despite the differences in their physical parameters.

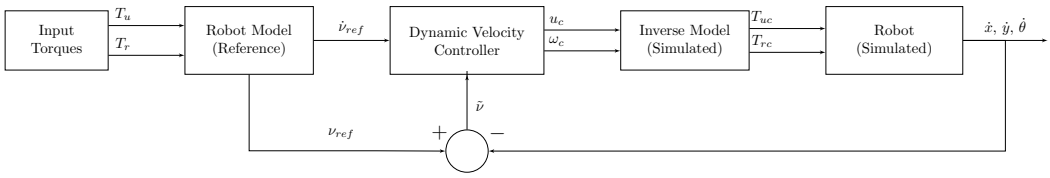


Figure 5. The proposed methodology: closed-loop velocity control where the reference system provides target velocities for the simulated model.

5. Results

In this section, we present the results of experiments conducted in a virtual environment for the identification and validation of the USV dynamic models, as well as the evaluation of the proposed model-following control strategy.

Figure 6 shows the velocity responses of IAcquaBot and WAM-V when subjected to the excitation signals described in Figure A1b. The results demonstrate that each USV exhibits different dynamic behaviors, leading to significantly different velocity responses. The corresponding velocity differences between both platforms, expressed in terms of RMSE, MAE, and R^2, are summarized in Table 4. This confirms the need to identify separate dynamic models for each platform.

Based on these observations, the identification process was carried out using the excitation signals from Figure A1a, which were designed to explore the dynamic range of each vehicle. The LQS method was applied to estimate the δ_i parameters of the dynamic model for each platform. Table 5 presents the identified parameters for both the IAcquaBot and WAM-V platforms, along with their 95% confidence intervals (CIs) and standard deviations (STDs).

Subsequently, two additional experiments were conducted to validate the identified dynamic models. Figure 7a,b present the comparison between the measured data and the simulated responses of the IAcquaBot model, while Figure 8a,b show the corresponding results for the WAM-V. In both cases, the models were excited using the input signals depicted in Figure A1b. The corresponding tracking error metrics for each state variable are summarized in Table 6. The close agreement between the measured states and the model outputs confirms that the identified parameters provide a sufficiently accurate representation of each USV for control design and simulation purposes.

An experiment was conducted to evaluate the performance of the proposed model-following controller. In this case, the excitation signals were manually generated using a joystick. The objective was to assess whether the WAM-V, equipped with the model-

following controller, could emulate the velocity response of the IAcquaBot to the same inputs. The velocity controller gains were empirically tuned.

Figure 9 compares the velocity responses for the joystick-generated inputs in three cases: (i) the IAcquaBot response to joystick inputs; (ii) the WAM-V response to the same joystick inputs, computed using its own identified model; and (iii) the WAM-V response obtained via the velocity controller using the IAcquaBot velocities as reference (WAM-V$_c$).

To quantitatively assess the similarity between the responses, Table 7 presents the performance metrics (RMSE, MAE, and R^2) for both WAM-V and WAM-Vc with respect to IAcquaBot, across all translational and rotational velocity components, as well as the position and orientation states.

Furthermore, four additional experiments were conducted to assess robustness under exogenous disturbances with different magnitudes and headings, combining steady currents and winds. The disturbance configurations for each trial are summarized in Table 8, and the resulting tracking errors (RMSE) are reported in Figure 10. Across all cases, both velocity and position errors remained in the $0.0x$ range.

The results highlight that, while the WAM-V exhibits significant deviations from the IAcquaBot in both kinematic and dynamic states, the WAM-Vc achieves errors that are orders of magnitude lower and R^2 values approaching 1.0, indicating a near-perfect tracking performance. These findings demonstrate that the proposed simulation strategy enables the WAM-V to closely replicate the velocity dynamics of the IAcquaBot.

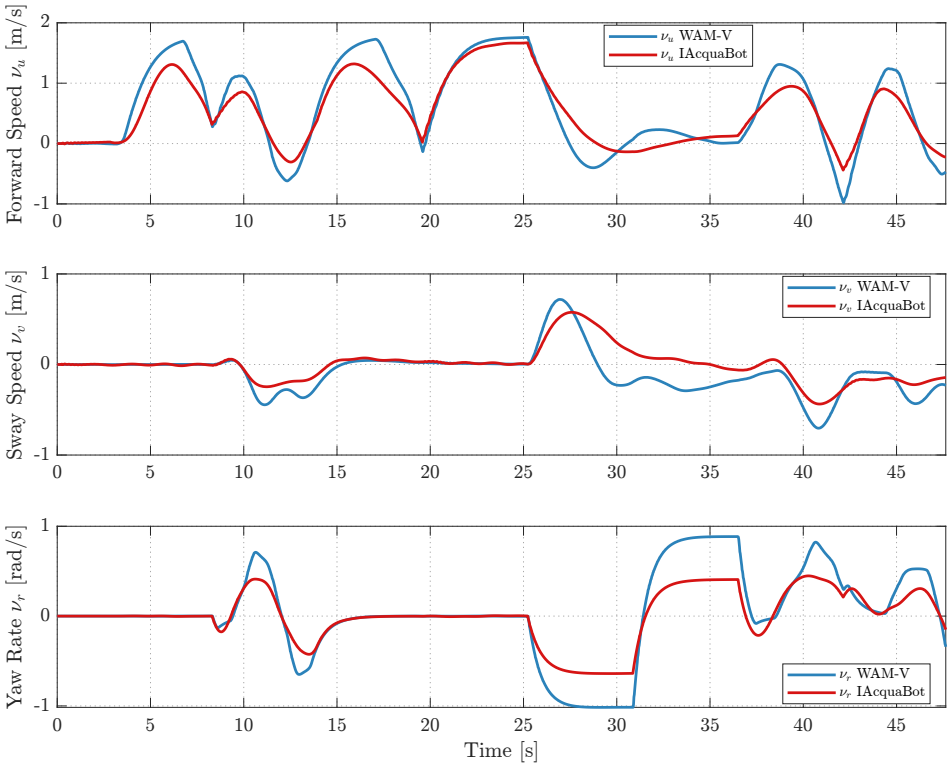


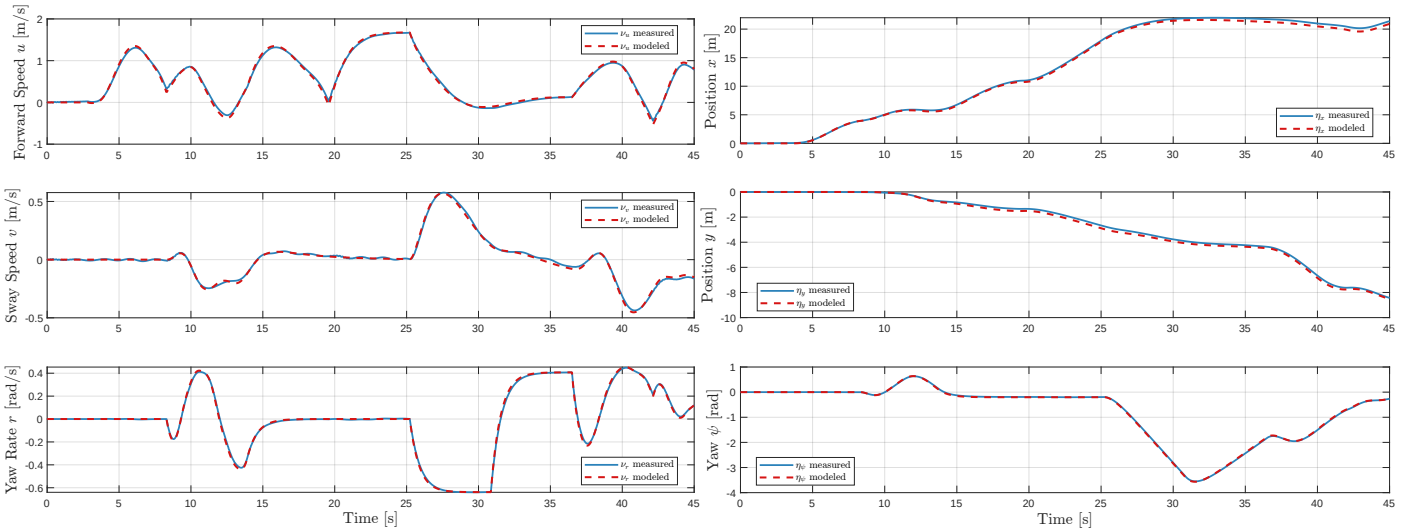
Figure 6. Measured body-fixed velocity responses ν_u, ν_v, and ν_r of IAcquaBot and WAM-V under the same excitation signals.

Table 4. Velocity differences between IAcquaBot and WAM-V.

Velocity Component	RMSE	MAE	R^2
Forward speed u [m/s]	0.25840	0.20668	0.86794
Sway speed v [m/s]	0.15799	0.10405	0.51773
Yaw rate r [rad/s]	0.21584	0.13807	0.79921

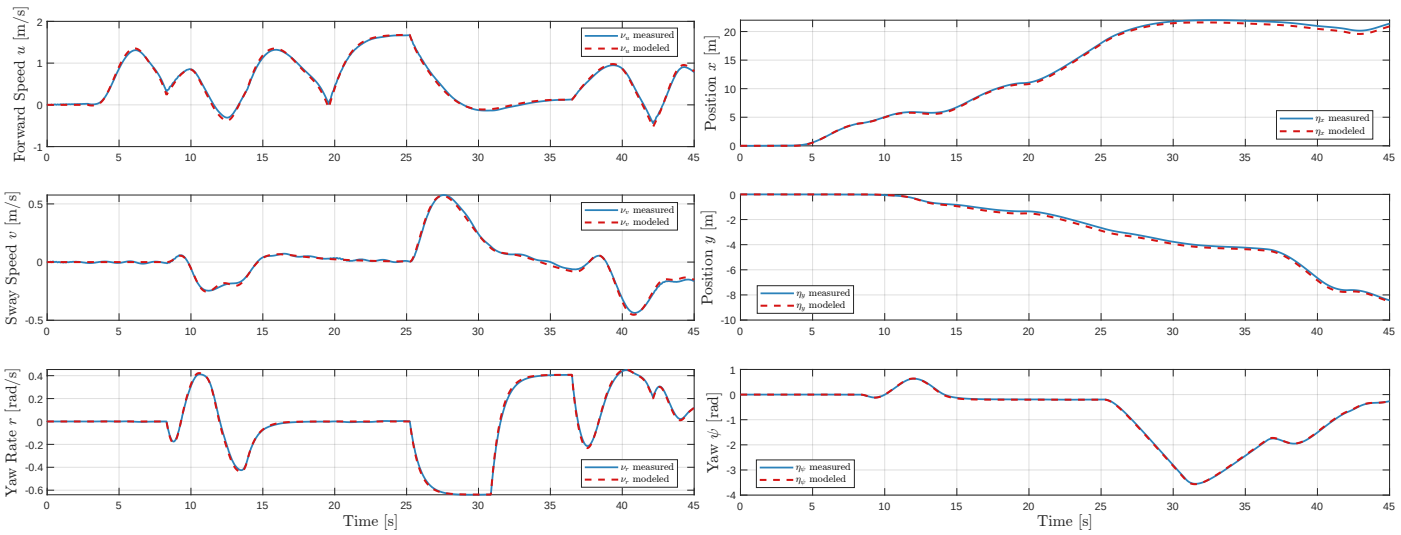
Table 5. Identified parameters δ_n for IAcquaBot and WAM-V with 95% confidence intervals (CIs) and standard deviations (STDs).

IAcquaBot			
Symbol	**Estimate**	**95% CI**	**STD**
δ_1	124.5923	[123.9456, 125.2389]	0.3299
δ_2	120.0275	[118.7015, 121.3535]	0.6765
δ_3	141.5164	[140.9026, 142.1302]	0.3132
δ_4	25.9335	[25.0368, 26.8303]	0.4575
δ_5	35.9833	[35.2669, 36.6996]	0.3655
δ_6	37.0247	[34.1923, 39.8570]	1.4451
δ_7	25.7968	[21.0590, 30.5346]	2.4172
δ_8	200.2944	[198.1114, 202.4775]	1.1138
δ_9	18.5319	[15.3741, 21.6898]	1.6111
WAM-V			
Symbol	**Estimate**	**95% CI**	**STD**
δ_1	125.9712	[125.4830, 126.4593]	0.2490
δ_2	121.4476	[120.2338, 122.6614]	0.6193
δ_3	70.5019	[69.9934, 71.0104]	0.2594
δ_4	26.5692	[25.5330, 27.6054]	0.5287
δ_5	35.8778	[35.2117, 36.5440]	0.3399
δ_6	36.4337	[33.3414, 39.5260]	1.5777
δ_7	27.4138	[22.6374, 32.1903]	2.4369
δ_8	99.1686	[97.1265, 101.2107]	1.0419
δ_9	10.6007	[8.2782, 12.9233]	1.1849



(**a**) (**b**)

Figure 7. Validation of the identified dynamic model for the IAcquaBot platform. (**a**) Velocities in the body-fixed frame: ν_u, ν_v, and ν_r. (**b**) Positions and orientation in the inertial frame: η_x, η_y, and η_ψ.



(**a**) (**b**)

Figure 8. Validation of the identified dynamic model for the WAM-V platform. (**a**) Velocities in the body-fixed frame: ν_u, ν_v, and ν_r. (**b**) Positions and orientation in the inertial frame: η_x, η_y, and η_ψ.

Table 6. Validation metrics (RMSE, MAE, and R^2) for the identified dynamic models of IAcquaBot and WAM-V.

	IAcquaBot			WAM-V		
State	**RMSE**	**MAE**	**R^2**	**RMSE**	**MAE**	**R^2**
ν_u [m/s]	0.03714	0.03111	0.99548	0.04508	0.03519	0.99598
ν_v [m/s]	0.01254	0.00984	0.99541	0.01314	0.00880	0.99666
ν_r [rad/s]	0.00956	0.00644	0.99887	0.01714	0.01025	0.99873
η_x [m]	0.32232	0.26361	0.99847	0.11266	0.09312	0.99988
η_y [m]	0.13300	0.11067	0.99774	0.33700	0.28346	0.97451
η_ψ [rad]	0.01699	0.01276	0.99976	0.02663	0.01918	0.99974

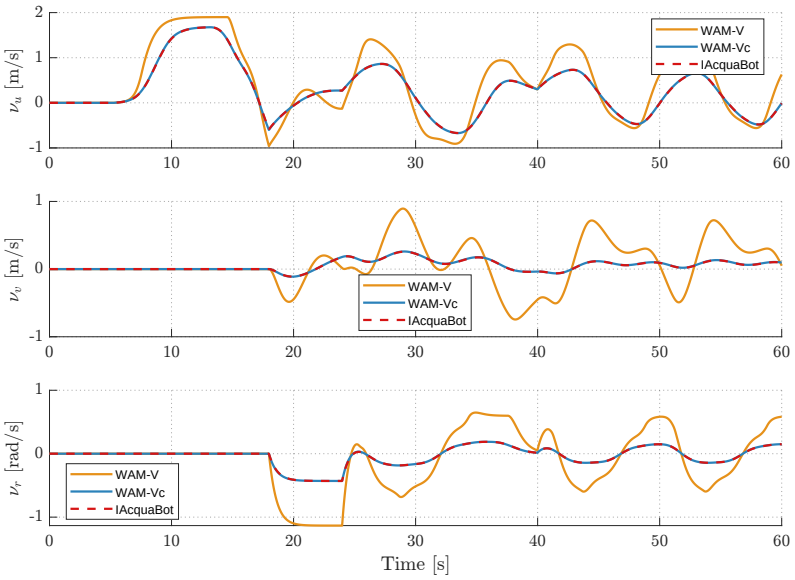


Figure 9. Comparison of the velocity responses for the three simulated vessels. Red dashed lines represent IAcquaBot's response to joystick inputs, burnt yellow lines correspond to WAM-V's response to the same inputs, and blue lines correspond to WAM-V's response obtained via a velocity controller using IAcquaBot's velocities as a reference.

Table 7. Performance metrics (RMSE, MAE, R^2) for WAM-V and WAM-Vc with respect to IAcquaBot.

State	RMSE		MAE		R^2	
	WAM-V	**WAM-Vc**	**WAM-V**	**WAM-Vc**	**WAM-V**	**WAM-Vc**
ν_u [m/s]	0.3596	0.0048	0.2830	0.0038	0.6120	0.99993
ν_v [m/s]	0.2850	0.0017	0.1995	0.0012	−12.435	0.99953
ν_r [rad/s]	0.3152	0.0015	0.2278	0.0009	−3.3631	0.99990
η_x [m]	14.0200	0.0162	10.6480	0.0119	−14.295	0.99998
η_y [m]	7.3910	0.0377	4.6920	0.0275	−3.3243	0.99989
η_ψ [rad]	3.3198	0.0031	2.6537	0.0020	−6.5001	0.99999

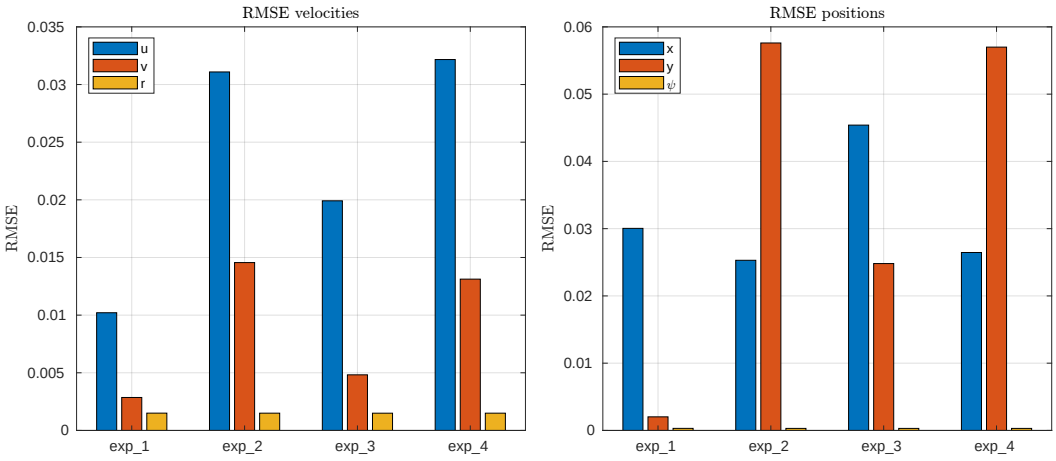


Figure 10. RMSE for WAM-V_c with respect to IAcquaBot across up to four experiments. **Left**: RMSE for velocities; **Right**: RMSE for positions.

Table 8. Disturbance profiles used in the four robustness trials. Notation (left to right): F_c = current force [N]; ψ_c = current direction [deg]; F_w = wind force [N]; ψ_w = wind direction [deg]; F_Σ = resultant force [N]; ψ_Σ = resultant direction [deg]. All directions are inertial-frame angles measured counter-clockwise from the $+X$ axis.

Trial	F_c [N]	ψ_c [deg]	F_w [N]	ψ_w [deg]	F_Σ [N]	ψ_Σ [deg]
Exp. 1	1.20	10.0	0.85	−5.0	2.03	3.8
Exp. 2	6.80	110.0	6.20	70.0	12.22	91.0
Exp. 3	2.70	80.0	2.40	40.0	4.79	61.2
Exp. 4	6.20	−70.0	4.90	−120.0	10.08	−91.9

5.1. Analysis of Results

This work proposes a methodology to build Digital Models of mobile robots in contexts where a complete characterization of their components is not available. This situation is common in research prototypes, where many devices lack detailed specifications from manufacturers or are directly developed by researchers, without a formal documentation of their physical or dynamic properties. This lack of information complicates the construction of simulation environments based on physics engines such as Gazebo or Webots, which require specific parameters to reproduce realistic behavior. As an alternative, we propose a strategy based on the parallel identification of the real vehicle's dynamic model, and the identification of a reliable simulated model with a similar structure. This way, the Digital Model does not need to replicate all physical details but rather emulate the observable dynamic behavior in the degrees of freedom relevant to the control.

We conducted our experiments using USVs to validate the approach. Across all state variables, the RMSE and MAE were reduced by more than two orders of magnitude; for instance, in ν_u, the RMSE decreased from 0.3596 m/s to 0.0048 m/s (approximately 98.7%

reduction) and the MAE from 0.2830 m/s to 0.0038 m/s (also 98.7% reduction). Similar improvements were achieved for ν_v and ν_r, with reductions exceeding 99%, while in η_x and η_ψ the RMSE decreased by 99.88% and 99.91%, respectively. These results indicate that the virtual USV exhibited behavior significantly different from the reference platform. With the controller implemented, however, R^2 values exceeded 0.9995 for all states, demonstrating an almost perfect match between the controlled USV and the reference dynamics. Although demonstrated with USVs, this methodology can be applied to any other mobile platform, provided that a reliable simulator with a similar and fully characterized model structure is available in the literature.

A key observation from the experiments is that, despite the real vehicle exhibiting a non-holonomic constraint, the tracking of the forward velocity and the yaw rate enable the indirect compensation of lateral motion. This was possible because of the natural coupling of the dynamics, showing that, even when only some degrees of freedom are explicitly controlled, it is possible to achieve dynamic behavior similar to that of the reference system. In this sense, a hybrid Digital Model was obtained to faithfully reproduce the three identified degrees of freedom, while the remaining three (roll, pitch, and heave) were inherited from the base simulated model.

An inherent limitation of including a controller in the simulation loop is that the simulated vehicle's behavior is constrained by the imposed references. This prevents the observation of free responses to unknown disturbances. However, if such disturbances can be identified or modeled (e.g., currents, winds, or external forces), they can be incorporated into the simulation loop to approximate the resulting behavior.

Although in this work the dynamic identification of the real platform was carried out using an LQS-based approach, the proposed strategy is not limited to this method. Alternative identification techniques, such as recursive estimation, subspace methods, or machine-learning-based approaches, could be employed to obtain comparable results provided that they produce a dynamic model suitable for reference tracking. Similarly, the velocity controller used in our experiments followed a model feedback structure, but this is not a strict requirement. Any control scheme capable of accurately tracking the reference velocities generated by the model can be integrated into the proposed methodology. In essence, the core idea is to perform model following based on velocity references and a dedicated velocity controller, regardless of the specific identification or control algorithms chosen.

Moreover, the proposed structure allows the testing of various control strategies, such as trajectory tracking, path following, or obstacle avoidance. The general scheme—signals input $\rightarrow$ reference model $\rightarrow$ tracking controller $\rightarrow$ inverse model of the simulator $\rightarrow$ robot simulator $\rightarrow$ output—can be interpreted as a single block, i.e., an input–output system. Consequently, the Digital Model not only represents the vehicle's dynamics but can also serve as a comprehensive simulation unit to validate navigation or control strategies.

5.2. Future Work

Future work will explore the integration of non-invertible models, such as neural networks, into the proposed strategy by adopting alternative control approaches. One promising option is Model Predictive Control (MPC), which can operate without an explicit inverse model. With suitable adaptations, MPC could enable the use of black box dynamics while maintaining tracking accuracy and robustness, although at the cost of increased computational complexity.

Another promising research direction is extending the proposed architecture towards the implementation of a Digital Shadow. This could be achieved by replacing the model-based reference velocities with real-time velocity measurements from the physical robot

(HiTL). In such a configuration, the simulator would continuously mirror the actual state and motion of the vehicle, enabling a synchronized virtual replica.

6. Conclusions

This study presented a novel methodology to design the Digital Models of mobile robots when a complete characterization of their components is not available. The technique is based on the dual identification of the dynamic models of the real robot and a virtual simulator with a similar structure, generating a hybrid twin that uses the identified degrees of freedom and inherits the remaining ones from the base simulator.

Two platforms with completely different characteristics in terms of dimensions, inertias, and hydrodynamic variables were employed to illustrate the applicability of the methodology under markedly different conditions. This selection aimed to illustrate the approach under clearly contrasting conditions. In real applications, however, the researcher should calibrate the mechanical design of the simulated model (URDF) to match the actual robot as closely as possible, and then use the model-following approach to fine-tune the simulated response, reducing residual errors and achieving a closer match to the real vehicle's behavior.

The experimental validation was conducted with two USVs, although the methodology is applicable to other mobile platforms as long as there is a reliable simulator with a comparable structure. This strategy overcomes the limitations of physics-based simulators when faced with incomplete data and facilitates the validation of control strategies for navigation, such as trajectory tracking or obstacle avoidance.

Although, in this case, the modeling relied on identification via LQS, alternative approaches, including models based on artificial intelligence such as neural networks or deep learning, could also be adopted. Since the methodology involves dual model analysis, it is advisable to employ computationally lightweight structures to maintain real-time performance. Additionally, the proposed strategy requires an inverse dynamics model. Black-box models or neural networks are typically not invertible, which limits their direct application in this framework.

The current scheme considers imposed references but can incorporate known external disturbances to improve realism. In summary, it offers a practical way to build Digital Models that are useful for the design and validation of control systems for mobile robots.

Finally, the proposed architecture constitutes a preliminary step toward the development of a Digital Shadow. Specifically, if the reference velocities were obtained directly from the real-time measurements of the physical robot instead of an identified model, the simulator could accurately replicate the actual behavior of the vehicle. This would allow for a synchronized virtual representation of the robot's state and motion, laying the foundation for real-time monitoring and advanced decision-making systems.

Author Contributions: Conceptualization, B.S.-M. and J.M.T.; methodology, F.R. and J.M.T.; software, B.S.-M. and J.V.-A.; validation, B.S.-M., J.V.-A. and J.M.T.; formal analysis, B.S.-M. and J.V.-A.; investigation, B.S.-M. and J.V.-A.; resources, B.S.-M. and J.V.-A.; data curation, B.S.-M. and J.V.-A.; writing—original draft preparation, B.S.-M. and J.V.-A.; writing—review and editing, F.R. and J.M.T.; visualization, B.S.-M. and J.V.-A.; supervision, F.R. and J.M.T.; project administration, J.M.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Instituto de Automática of the Universidad Nacional deSan Juan in collaboration with CONICET (Consejo Nacional de Investigaciones Científicas y Técnicas) and Universidad Indoamérica.

Data Availability Statement: If anyone wants to obtain the original data of this paper, please contact with Brayan Saldarriaga-Mesa.

Acknowledgments: The authors would like to express their gratitude to Universidad Indoamérica for its support of this research through the "Tecnologías de la Industria 4.0 en Educación, Salud, Empresa e Industria" program. The authors also acknowledge the Instituto de Automática of Universidad Nacional de San Juan and CONICET for providing the facilities and workspace necessary to conduct this research.

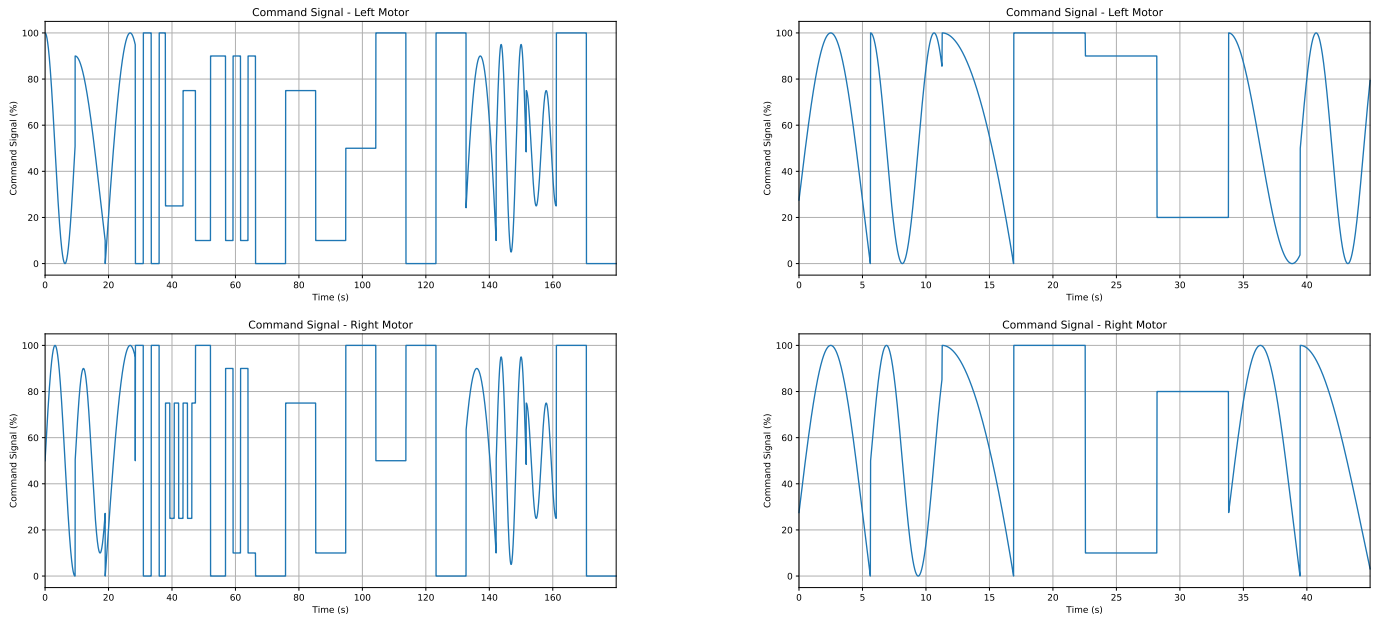
Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

USV	Unmanned Surface Vehicle
VRX	Virtual RobotX (simulation for autonomous vehicle competitions)
WAM-V	Wave Adaptive Modular Vessel
DTs	Digital Twins
DMs	Digital Models
UAV	Unmanned Aerial Vehicle
UGV	Unmanned Ground Vehicle
LQS	Least Quadratic Squares
DOF	Degrees of Freedom
CI	Confidence Intervals
STD	Standard Deviations
RMSE	Root Mean Square Error
MAE	Mean Absolute Error
R2	Coefficient of Determination
SiTL	Software-in-the-Loop
HiTL	Hardware-in-the-Loop

Appendix A



(**a**) (**b**)

Figure A1. Command (CMD) signals sent to the left and right thrusters during the system identification experiments. (**a**) Identification signals, designed to explore the vessel's full dynamic range. (**b**) Independent validation signals, used to test how well the identified model can predict the vessel's behavior.

References

1. Kritzinger, W.; Karner, M.; Traar, G.; Henjes, J.; Sihn, W. Digital Twin in Manufacturing: A Categorical Literature Review and Classification. *IFAC-PapersOnLine* **2018**, *51*, 1016–1022. [CrossRef]
2. Sepasgozar, S.M.E. Differentiating Digital Twin from Digital Shadow: Elucidating a Paradigm Shift to Expedite a Smart, Sustainable Built Environment. *Buildings* **2021**, *11*, 151. [CrossRef]
3. Barbie, A.; Hasselbring, W. From Digital Twins to Digital Twin Prototypes: Concepts, Formalization, and Applications. *IEEE Access* **2024**, *12*, 75337–75365. [CrossRef]
4. MathWorks. What Is Hardware-in-the-Loop (HIL)? 2025. Available online: https://www.mathworks.com/discovery/hardware-in-the-loop-hil.html (accessed on 12 August 2025).
5. DJI. DJI Software Development Kit (SDK). 2025. Available online: https://developer.dji.com (accessed on 12 August 2025).
6. Sagitov, A.; Gerasimov, Y. Towards DJI Phantom 4 Realistic Simulation with Gimbal and RC Controller in ROS/Gazebo Environment. In Proceedings of the 2017 10th International Conference on Developments in eSystems Engineering (DeSE), Paris, France, 14–16 June 2017; pp. 262–266. [CrossRef]
7. Speirs, P.J.; Renker, M.; Aulenbacher, U.; Wellig, P.; Murk, A. High-Detail X-Band RCS Simulations of a DJI S900 Hexacopter, and Comparisons Against Measurements. In Proceedings of the 2021 21st International Radar Symposium (IRS), Berlin, Germany, 21–22 June 2021; pp. 1–10. [CrossRef]
8. Parrot. Parrot SDK and Simulation Environment. 2025. Available online: https://developer.parrot.com (accessed on 12 August 2025).
9. Giernacki, W. Iterative Learning Method for In-Flight Auto-Tuning of UAV Controllers Based on Basic Sensory Information. *Appl. Sci.* **2019**, *9*, 648. [CrossRef]
10. Nandkumar, C.; Shukla, P.; Varma, V. Simulation of Indoor Localization and Navigation of Turtlebot 3 Using Real Time Object Detection. In Proceedings of the 2021 International Conference on Disruptive Technologies for Multi-Disciplinary Research and Applications (CENTCON), Bengaluru, India, 19–21 November 2021; pp. 222–227. [CrossRef]
11. Ma, C.; Zhou, Y.; Li, Z. A New Simulation Environment Based on AirSim, ROS, and PX4 for Quadcopter Aircrafts. In Proceedings of the 2020 6th International Conference on Control, Automation and Robotics (ICCAR), Singapore, 20–23 April 2020; pp. 486–490. [CrossRef]
12. Niit, E.A.; Smit, W.J. Integration of Model Reference Adaptive Control (MRAC) with PX4 Firmware for Quadcopters. In Proceedings of the 2017 24th International Conference on Mechatronics and Machine Vision in Practice (M2VIP), Auckland, New Zealand, 21–23 November 2017; pp. 1–6. [CrossRef]
13. Jung, S.; Kim, Y.J. MILS and HILS Analysis of Power Management System for UAVs. *IEEE Access* **2023**, *11*, 79240–79255. [CrossRef]
14. Clearpath Robotics. Clearpath Simulator. 2025. Available online: https://docs.clearpathrobotics.com/docs/ros/tutorials/simulator/overview/ (accessed on 12 August 2025).
15. Morales, J.; Castelo, I.; Serra, R.; Lima, P.U.; Basiri, M. Vision-Based Autonomous Following of a Moving Platform and Landing for an Unmanned Aerial Vehicle. *Sensors* **2023**, *23*, 829. [CrossRef] [PubMed]
16. Stan, A.C. A Decentralised Control Method for Unknown Environment Exploration Using Turtlebot 3 Multi-Robot System. In Proceedings of the 2022 14th International Conference on Electronics, Computers and Artificial Intelligence (ECAI), Ploiesti, Romania, 30 June–1 July 2022; pp. 1–6. [CrossRef]
17. PX4 Development Team. PX4 USV Simulator—Archived Project Note. 2023. Available online: https://docs.px4.io (accessed on 12 August 2025).
18. Cyberbotics. Heron USV Simulation Model in Webots. 2025. Available online: https://cyberbotics.com (accessed on 12 August 2025).
19. Bingham, B.; Agüero, C.; McCarrin, M.; Klamo, J.; Malia, J.; Allen, K.; Lum, T.; Rawson, M.; Waqar, R. Toward Maritime Robotic Simulation in Gazebo. In Proceedings of the OCEANS 2019 MTS/IEEE Seattle, Seattle, WA, USA, 27–31 October 2019; pp. 1–10. [CrossRef]
20. Wu, X.; Wei, C. DRL-Based Motion Control for Unmanned Surface Vehicles with Environmental Disturbances. In Proceedings of the 2023 IEEE International Conference on Unmanned Systems (ICUS), Hefei, China, 13–15 October 2023; pp. 1696–1700. [CrossRef]
21. Li, J.; Chavez-Galaviz, J.; Azizzadenesheli, K.; Mahmoudian, N. Dynamic Obstacle Avoidance for USVs Using Cross-Domain Deep Reinforcement Learning and Neural Network Model Predictive Controller. *Sensors* **2023**, *23*, 3572. [CrossRef] [PubMed]
22. Chen, L.; Liu, C.; Guo, S.; Qian, H. Vision-Guided UAV Landing on a Swaying Ocean Platform in Simulation. In Proceedings of the 2023 IEEE International Conference on Real-time Computing and Robotics (RCAR), Datong, China, 17–20 July 2023; pp. 456–462. [CrossRef]
23. Bayrak, M.; Bayram, H. COLREG-Compliant Simulation Environment for Verifying USV Motion Planning Algorithms. In Proceedings of the Oceans 2023—Limerick, Limerick, Ireland, 5–8 June 2023; pp. 1–10. [CrossRef]
24. Campos, D.F.; Matos, A.; Pinto, A.M. Multi-domain Inspection of Offshore Wind Farms Using an Autonomous Surface Vehicle. *SN Appl. Sci.* **2021**, *3*, 455. [CrossRef]

25. Li, C.; Yao, L.; Lu, Y.; Zhang, S.; Zhang, T. DTL-GNN: Digital Twin Lightweight Method Based on Graph Neural Network. *Future Internet* **2025**, *17*, 65. [CrossRef]
26. Stączek, P.; Pizoń, J.; Danilczuk, W.; Gola, A. A Digital Twin Approach for the Improvement of an Autonomous Mobile Robots (AMR's) Operating Environment—A Case Study. *Sensors* **2021**, *21*, 7830. [CrossRef] [PubMed]
27. Li, R.; Shang, X.; Wang, Y.; Liu, C.; Song, L.; Zhang, Y.; Gu, L.; Zhang, X. Research on Parameter Compensation Method and Control Strategy of Mobile Robot Dynamics Model Based on Digital Twin. *Sensors* **2024**, *24*, 8101. [CrossRef] [PubMed]
28. Betzer, J.S.; Boudjadar, J.; Frasheri, M.; Talasila, P. Digital Twin Enabled Runtime Verification for Autonomous Mobile Robots under Uncertainty. *arXiv* **2024**, arXiv:2412.09913. [CrossRef]
29. Cheng, C.; Zhang, H.; Sun, Y.; Tao, H.; Chen, Y. A Cross-Platform Deep Reinforcement Learning Model for Autonomous Navigation without Global Information in Different Scenes. *Control Eng. Pract.* **2024**, *150*, 105991. [CrossRef]
30. Fossen, T.I. *Handbook of Marine Craft Hydrodynamics and Motion Control*; Wiley: Oxford, UK, 2021.
31. Carelli, R.; Secchi, H.; Mut, V. Algorithms for Stable Control of Mobile Robots with Obstacle Avoidance. *Lat. Am. Appl. Res.* **1999**, *29*, 191–196.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.