Journal of *Marine Science and Engineering*

MDPI

Article

An Efficient Underwater Navigation Method Using MPC with Unknown Kinematics and Non-Linear Disturbances

Pablo Barreno, Juan Parras and Santiago Zazo *

Information Processing and Telecommunications Center, Universidad Politécnica de Madrid, 28040 Madrid, Spain
* Correspondence: santiago.zazo@upm.es

Abstract: Many Autonomous Underwater Vehicles (AUVs) need to cope with hazardous underwater medium using a limited computational capacity while facing unknown kinematics and disturbances. However, most algorithms proposed for navigation in such conditions fail to fulfil all conditions at the same time. In this work, we propose an optimal control method, based on a receding horizon approach, namely MPC (Model Predictive Control). Our model also estimates the kinematics of the medium and its disturbances, using efficient tools that rely on the use of linear algebra and first-order optimization methods. We also test our ideas using an extensive set of simulations, which show that the proposed ideas are very competitive in terms of cost and computational efficiency in cases of total and partial observability.

Keywords: optimal control; model predictive control; least squares; AUV; disturbances

check for updates

Citation: Barreno, P.; Parras, J.; Zazo, S. An Efficient Underwater Navigation Method Using MPC with Unknown Kinematics and Non-Linear Disturbances. *J. Mar. Sci. Eng.* **2023**, *11*, 710. https://doi.org/10.3390/jmse11040710

Academic Editor: Weicheng Cui and Rafael Morales

Received: 13 February 2023
Revised: 17 March 2023
Accepted: 24 March 2023
Published: 25 March 2023

Copyright: © 2023 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

In recent years, the development of autonomous navigation methods has gathered significant attention, yielding an ever-increasing field of research. Special attention is paid to aerial drones, which are becoming inexpensive, and hence, available for a myriad of applications such as remote sensing, real-time monitoring, search and rescue missions, delivery or agriculture, to mention a few [1]. In addition, a lot of interest is placed on the development of autonomous land vehicles, such as cars, which may suppose a revolution in the way that transportation is conceived with deep impacts on our society [2]. Vehicles that receive comparatively less attention are those designed to work underwater, which find applications as diverse as surveillance, warfare, inspection of wreckage, search and rescue missions, ocean exploration, scientific research, and repair and maintenance of structures such as oil platforms or underwater cables [3].

To operate in the underwater medium, Remotely Operated Vehicles (ROVs) are frequently used, where a ROV is controlled by a human using a wired connection to a vessel. However, ROVs have a limited range due to their need to be connected [3]. To overcome this limitation, Autonomous Underwater Vehicles (AUVs) can be used, which, however, face multiple challenges derived from the hazardous underwater medium.

A first challenge is related to the rapid attenuation that radio signals experience in the water, which means that satellite positioning systems cannot be used in the underwater medium [4]. To ease this problem, many studies have developed methods for locating AUVs using inertial sensors, acoustic techniques or geophysical navigation and beacons [3–5].

Another challenge that AUVs face is the complexity of the control of the vehicle, where many methods have been proposed, ranging from simple PID (Proportional–Integral–Derivative) controllers [6] to more complex non-linear control methods [7,8], fuzzy logic based controllers [9,10] and neural networks [11,12]. Note that not all these methods are valid for all AUVs, as the methods that can be included strongly depend on the computational capabilities of the AUV (e.g., an AUV may not have enough capabilities to train a deep neural network, as in [12]).

Finally, another challenge that arises in the underwater medium is related to the presence of unknown disturbances that affect the motion of the AUV. This brings additional difficulties to the location of the AUV [13,14], and hence, has been addressed by many works, such as [12,15–19] or [20]. However, all these works either consider that the kinematics are known, which simplifies the problem [15,16,18–20], or the algorithm proposed requires a high computational capacity on the AUV (e.g., references [12,15,17] require training neural networks), so these algorithms are unfeasible for low-capacity AUVs.

Hence, in this work, we will address this gap by proposing an efficient navigation method that can deal with both unknown disturbances and kinematics, where our main contributions are:

- The proposed optimal control method is computationally efficient, as we make use of a Model-Predictive Control (MPC) method as controller, that can achieve a very high computational efficiency for low capacity devices as shown in [21,22].
- In order to deal with unknown kinematics, we use a linear approximation that is based only on linear algebra, and hence, it is computationally efficient. In case that the AUV has enough computational capacity, it is possible to replace this module by other techniques such as LWPR (Locally Weighted Projection Regression) [23,24], XCSF (eXtended Classifier Systems for Function approximation) [25,26] or ISSGPR (Incremental Sparse Spectrum Gaussian Process Regression) [27], to mention some.
- Although it is possible to include the effect of the disturbances in the kinematics estimator, we derive a specific non-linear disturbance estimator based on optimization which allows for significantly improving the results, with increases in cost of up to a 47%.

The rest of the paper is structured as follows. In Section 2, we describe the setup of the navigation problem we are going to solve, which is general and can adapt to different situations. Then, Section 3 introduces our main result: the estimators proposed for the kinematics and the disturbances. This method is then tested in Section 4, where a set of extensive simulations is driven in order to evaluate the performance and bounds of the proposed methods. Finally, some conclusions are drawn in Section 5.

2. Setup Description

We first describe the setup proposed for our method in this Section.

2.1. General Diagram

Let us start by introducing the main block diagram for our setup, which is composed of three main blocks and can be seen in Figure 1:

- The environment block, that simulates the underwater conditions, that is, the kinematics. This block takes as input the control $\mathbf{u}$, that contains the change in the AUV actuators, and returns an observation $\mathbf{o}$, which reflect the information that the sensors of the AUV capture.
- The estimation block, whose main purpose is to extract meaningful information $\hat{\mathbf{s}}$ for the controller block from the signals provided by the AUV sensors $\mathbf{o}$.
- The controller block, which is devoted to decide which controls $\mathbf{u}$ are to be taken at each time instant. This block receives as input the information extracted from the estimation block, and outputs the adequate control.

Note that this definition of the three main blocks of our system is very general, hence, it can adapt to different concrete settings. Let us now delve into more detail of each block in the next Sections.

$$\begin{tikzcd}
 & \boxed{\text{Environment}} \arrow[dl, "\mathbf{o}"'] & \\
\boxed{\text{Estimation}} \arrow[rr, "\hat{\mathbf{s}}"] & & \boxed{\text{Controller}} \arrow[ul, "\mathbf{u}"']
\end{tikzcd}$$

Figure 1. Schematic diagram for our setting, where the three main blocks and their relations are depicted. Note that they relate in a closed loop way, i.e., the controller produces a control **u** based on the current $\hat{\mathbf{s}}$ that causes a change in the AUV state, according to the environment block that simulates the underwater medium. Hence, the value **o** observed by the AUV sensors change and the estimation block updates its estimate input to the controller $\hat{\mathbf{s}}$, and then the whole process starts over.

2.2. *Environment Block*

This block simulates the underwater medium where our AUV is. This block is inspired by [12], and contains three main ideas: the dynamical system that models the underwater kinematics, an underwater disturbances model and an observation model.

2.2.1. Underwater Navigation Model

The first element of the environment block is the dynamical system that reflects the physics of the underwater environment, i.e., the kinematics. In this work, we focus on discrete-time dynamical systems that can be expressed using the next equation:

$$\mathbf{s_{n+1}} = f(\mathbf{s_n}, \mathbf{u_n}) = \mathbf{A}\mathbf{s_n} + \mathbf{B}\mathbf{u_n} + \mathbf{d}(\mathbf{s_n}) \tag{1}$$

where f is the transition function, $n \in \{0, 1, 2...\}$ denotes the time index, the column vector $\mathbf{s_n}$ is the state of the system at time n, the column vector $\mathbf{u_n}$ denotes the controls (e.g., the effects of the actuators), and the column vector $\mathbf{d}(\mathbf{s_n})$ represents the disturbances. We define $|S|$ as the state dimension, so $\mathbf{s_n} \in \mathbb{R}^{|S|}$ (also, $\mathbf{d}(\mathbf{s_n}) \in \mathbb{R}^{|S|}$), and $|U|$ as the control dimension, so $\mathbf{u_n} \in \mathbb{R}^{|U|}$. The matrix $\mathbf{A} \in \mathbb{R}^{|S|\times|S|}$ contains the state change due to the previous state $\mathbf{s_n}$, and the matrix $\mathbf{B} \in \mathbb{R}^{|S|\times|U|}$ contains the state change due to the control $\mathbf{u_n}$. Finally, note that (1) represents a class of linear dynamical systems, but it can also accommodate non-linear dynamical systems by means of linearizing the transition function using a first-order Taylor expansion (as in [21] or [22]).

2.2.2. Disturbance Models

Following [12], we consider the following three main disturbances that may appear in an underwater environment: swirls, currents and constant fields. As in [12], these disturbances are modelled as non-linear vector fields that affect the acceleration of the AUV in a two-dimensional system. The constant field disturbance is:

$$\begin{bmatrix} d_{x,n}(\mathbf{s_n}) \\ d_{y,n}(\mathbf{s_n}) \end{bmatrix} = \begin{bmatrix} k_1 \cdot \cos\alpha \\ k_1 \cdot \sin\alpha \end{bmatrix} \tag{2}$$

where $k_1 \in \mathbb{R}$ is the strength of the acceleration produced by the field and $\alpha \in [0, 2\pi]$ is its angle of orientation, which is independent of the position of the AUV. Note that depending on the sign of k_1, the angle may be α or $\alpha + \pi$. Additionally, we remark that $d_{x,n}(\mathbf{s_n})$ and $d_{y,n}(\mathbf{s_n})$ are the components of the vector $\mathbf{d}(\mathbf{s_n})$ affected by the disturbances.

A current is defined depending on whether it has horizontal (i.e., on the X-axis) or vertical (i.e., on the Y-axis) orientation as follows:

$$\begin{bmatrix} d_{x,n}(\mathbf{s_n}) \\ d_{y,n}(\mathbf{s_n}) \end{bmatrix} = \begin{bmatrix} k_2 \cdot e^{-\dfrac{(y_n - y_0)^2}{\omega^2}} \\ 0 \end{bmatrix} \quad \begin{bmatrix} d_{x,n}(\mathbf{s_n}) \\ d_{y,n}(\mathbf{s_n}) \end{bmatrix} = \begin{bmatrix} 0 \\ k_2 \cdot e^{-\dfrac{(x_n - x_0)^2}{\omega^2}} \end{bmatrix} \tag{3}$$

where $x_0 \in \mathbb{R}$ and $y_0 \in \mathbb{R}$ denote the position of maximum strength of the current, $k_2 \in \mathbb{R}$ controls the strength of the current and its direction, and $\omega \in \mathbb{R}$ controls the width of the current.

Finally, a swirl can be modelled as follows:

$$\begin{bmatrix} d_{x,n}(\mathbf{s_n}) \\ d_{y,n}(\mathbf{s_n}) \end{bmatrix} = \begin{bmatrix} \frac{k_3}{\sqrt{(x_n - x_0)^2 + (y_n - y_0)^2}}(y_n - y_0) \\ \frac{-k_3}{\sqrt{(x_n - x_0)^2 + (y_n - y_0)^2}}(x_n - x_0) \end{bmatrix} \tag{4}$$

where x_0 and y_0 denote the position of the vortex of the swirl and $k_3 \in \mathbb{R}$ indicates its strength and direction (clockwise or counter-clockwise).

Note that the disturbance depends on the position (x_n, y_n) of the AUV, and hence, they are position-dependent, except for the constant field disturbance which is position-independent. This justifies that in (1), the vector $\mathbf{d}(\mathbf{s_n})$, whose components are the disturbances, depends on the current state $\mathbf{s_n}$.

Additionally, if we define the vector $\mathbf{p}$ as the vector that contains the parameter of each disturbance, we have that $\mathbf{p}_{cf} = (k_1, \alpha)$ for the constant field, $\mathbf{p}_{hc} = (k_2, y_0, \omega)$ for the horizontal current, $\mathbf{p}_{vc} = (k_2, x_0, \omega)$ for the vertical current, and $\mathbf{p}_{sw} = (k_3, x_0, y_0)$ for the swirl.

2.2.3. Observation Model

The final element in our environment block is the observation model, as the information captured by the AUV sensors need not be the actual next state of the system $\mathbf{s_{n+1}}$, due to the sensors imperfections. For instance, when the AUV is underwater, it cannot use GPS (Global Positioning System) as a location method due to the losses of the radio signal due to the water. Thus, the AUV must resort to other set of techniques to estimate its location, such as inertial based methods, acoustic sensors or beacon methods, among others [3,4]. Hence, this means that in general, the information gathered by the AUV sensors will be an observation $\mathbf{o_n}$, a noisy and/or incomplete version of the actual state of the system. Mathematically, we can say that there exists an observation model g such that:

$$\mathbf{o_{n+1}} = g(\mathbf{s_{n+1}}) \tag{5}$$

where g is a mapping from the state space to the observation space. Depending on the sensors in the AUV and the location method chosen, the observation model g will vary, where we note that each location method brings a certain tradeoff in computational load and precision in the estimation of the state, among others [4].

2.3. Estimation Block

The ideal purpose of the estimation block is to revert the observation model, i.e., it tries to retrieve the actual state $\mathbf{s_{n+1}}$ given an observation $\mathbf{o_{n+1}}$. In general, the estimation block implements a mapping h that goes from the observation space to the state space as:

$$\mathbf{\hat{s}_{n+1}} = h(\mathbf{o_{n+1}}) \tag{6}$$

where, ideally, we have that $h = g^{-1}$. In real life, it is seldom possible to exactly recover the original state $\mathbf{s_{n+1}}$: we can at best provide an estimation $\mathbf{\hat{s}_{n+1}}$ of it. Note that we use the hat notation to denote that the output of this block is an estimation of the actual state. We emphasize that the choice of the estimation block is deeply intertwined with the observation model, as the estimation block tries to cancel out the effect of the observation model.

The most favorable case is the fully observable case, in which the AUV observes the actual state, so we have $\mathbf{o_{n+1}} = g(\mathbf{s_{n+1}}) = \mathbf{s_{n+1}}$. Note that in this case, we do not need any estimation block, but this case seldom, if ever, happens in practice. It is more common to have a partially observable case, in which the AUV observes a noisy and/or incomplete version of the state [28]. A frequent approach consists on assuming that the observation model adds noise to the state, so we would have:

$$\mathbf{o_{n+1}} = g(\mathbf{s_{n+1}}) = \mathbf{s_{n+1}} + \epsilon \tag{7}$$

where ϵ is a noise that follows a Gaussian distribution with a certain mean and covariance [4]. If we know the transition model (e.g., (1)), it is possible to obtain an estimate of the state using a Kalman Filter (KF) [29]. However, a KF can only be applied if the transition model is linear. Two extensions to the non-linear case are the Extended KF (EKF), which employs a first-order approximation of the nonlinear transition model, and the Unscented KF (UKF), which uses an unscented transformation. Due to their properties and simplicity, these filters are ubiquitous in tracking applications [4], and can even be the optimal estimators under certain conditions. However, note that knowledge of the transition function $f(\mathbf{s_n}, \mathbf{u_n})$ is required for these filters.

2.4. Controller Block

The third main block in our model is the controller, which takes as input an state estimation $\hat{\mathbf{s}}_\mathbf{n}$ and returns the adequate control for that state $\mathbf{u_n}$. In this work, we focus on optimal controllers, i.e., controllers that solve a certain optimization problem. Note that the controller will be posed in terms of the actual state, $\mathbf{s_{n+1}}$, but as we have mentioned before, in general we only have access to a state estimation $\hat{\mathbf{s}}_\mathbf{n+1}$. For simplicity, we assume that the target of the AUV is to reach the origin of the state space, so we can pose the optimization problem as follows:

$$\begin{aligned} & arg \min_{\mathbf{u_n}} \sum_{n=0}^{N-1} \left(\mathbf{s_{n+1}}^T Q \mathbf{s_{n+1}} + \mathbf{u_n}^T R \mathbf{u_n} \right) \\ s.t. \quad & \mathbf{s_{n+1}} = f(\mathbf{s_n}, \mathbf{u_n}) \\ & \mathbf{s_0} = \mathbf{s(0)} \\ & \mathbf{Z_s s_{n+1}} \leq \mathbf{z_s}, \quad \forall n \in 0,1,2,\dots,N-1 \\ & \mathbf{Z_u u_n} \leq \mathbf{z_u}, \quad \forall n \in 0,1,2,\dots,N-1 \end{aligned} \tag{8}$$

where N is the horizon, $Q \in \mathbb{R}^{|S|\times|S|}$ is a matrix that contains the cost related to the states, $R \in \mathbb{R}^{|U|\times|U|}$ is a matrix containing the cost related to the control, f is the transition function controlling the next states (1), $\mathbf{s(0)} \in \mathbb{R}^{|S|}$ is the given initial state, $\mathbf{Z_s} \in \mathbb{R}^{|z_s|\times|S|}$ and $\mathbf{z_s} \in \mathbb{R}^{|z_s|}$ define $|z_s|$ inequality restrictions over the states, and $\mathbf{Z_u} \in \mathbb{R}^{|z_u|\times|U|}$ and $\mathbf{z_u} \in \mathbb{R}^{|z_u|}$ define $|z_u|$ inequality restrictions over the controls. Note that the inequalities can enforce that states and controls belong to an admissible region, as would be the case in real AUV with limited actuator capacity (such as limited acceleration).

The problem (8) is of considerable interest for the control theory community, where some of its solutions are well known. For instance, in case that the transition function is linear in the states and controls, and there are no restrictions, the problem is a Linear-Quadratic controller, and its optimal solution can be computed using the Riccati equations [30].

However, in case that there are restrictions, the optimization problem has a higher complexity, that increases with N, as the number of constrains increases linearly with N. In order to address this, Model Predictive Control (MPC) is widely used: this technique alleviates the computational complexity at the cost of obtaining suboptimal controllers. The idea is to solve problem (8) using a limited horizon $N_0 < N$ instead of the actual horizon N, and apply only part of the controllers obtained (e.g., only $\mathbf{u_0}$) in order to progress a certain number of steps, and then solve again the optimization problem and start over. Note that, under this scheme, the parameter N_0 controls the tradeoff between optimality and computational complexity: a low value of N_0 eases the computation, but it is also short-sighed and hence suboptimal. Due to its properties, MPC is widely used in real life. Note that instead of solving a very complex optimization problem over a long horizon, it solves several optimization problems over shorter horizons. The solution to problem (8) is known to be a piecewise linear controller [31], and this can be used to further decrease the computational load by reusing controllers, as can be seen in [21,22]. Note that the solution to (8) is not only optimal, but also stable [31] if the actual transition function is linear. In our case, we may be approximating a non-linear system using a linear model, and the stability strongly depends on the actual system and cannot be stated in the general case.

Let us delve into a detailed formulation of MPC to our problem. First, let us assume a no-disturbance case: in this case, the transition function (1) becomes:

$$\mathbf{s_{n+1}} = f(\mathbf{s_n}, \mathbf{u_n}) = \mathbf{A}\mathbf{s_n} + \mathbf{B}\mathbf{u_n} \tag{9}$$

and we can obtain any state as a function of the initial state $\mathbf{s}(\mathbf{0})$ and the controls as follows:

$$\begin{bmatrix} \mathbf{s_1} \\ \mathbf{s_2} \\ \dots \\ \mathbf{s_{N_0}} \end{bmatrix} = \begin{bmatrix} \mathbf{B} & 0 & \dots & 0 \\ \mathbf{AB} & \mathbf{B} & \dots & 0 \\ \dots & \dots & \dots & \dots \\ \mathbf{A}^{N_0-1}\mathbf{B} & \mathbf{A}^{N_0-2}\mathbf{B} & \dots & \mathbf{B} \end{bmatrix} \begin{bmatrix} \mathbf{u_0} \\ \mathbf{u_1} \\ \dots \\ \mathbf{u_{N_0-1}} \end{bmatrix} + \begin{bmatrix} \mathbf{A} \\ \mathbf{A}^2 \\ \dots \\ \mathbf{A}^{N_0} \end{bmatrix} \mathbf{s}(\mathbf{0}) \tag{10}$$

which can be expressed in more compact form as:

$$\mathbf{s} = \mathcal{B}\mathbf{u} + \mathcal{A}\mathbf{s}(\mathbf{0}) \tag{11}$$

where

$$\mathbf{s} = \begin{bmatrix} \mathbf{s_1} \\ \mathbf{s_2} \\ \dots \\ \mathbf{s_{N_0}} \end{bmatrix} \quad \mathbf{u} = \begin{bmatrix} \mathbf{u_0} \\ \mathbf{u_1} \\ \dots \\ \mathbf{u_{N_0-1}} \end{bmatrix} \quad \mathcal{A} = \begin{bmatrix} \mathbf{A} \\ \mathbf{A}^2 \\ \dots \\ \mathbf{A}^{N_0} \end{bmatrix} \quad \mathcal{B} = \begin{bmatrix} \mathbf{B} & 0 & \dots & 0 \\ \mathbf{AB} & \mathbf{B} & \dots & 0 \\ \dots & \dots & \dots & \dots \\ \mathbf{A}^{N_0-1}\mathbf{B} & \mathbf{A}^{N_0-2}\mathbf{B} & \dots & \mathbf{B} \end{bmatrix} \tag{12}$$

so note that $\mathbf{s} \in \mathbb{R}^{N_0|S|}$, $\mathbf{u} \in \mathbb{R}^{N_0|U|}$, $\mathcal{A} \in \mathbb{R}^{N_0|S|\times|S|}$ and $\mathcal{B} \in \mathbb{R}^{N_0|S|\times N_0|U|}$. If we define similarly $\mathcal{Q} \in \mathbb{R}^{N_0|S|\times N_0|S|}$, $\mathcal{R} \in \mathbb{R}^{N_0|U|\times N_0|U|}$, $\mathcal{Z}_\mathbf{s} \in \mathbb{R}^{N_0|z_s|\times N_0|S|}$, $\mathcal{Z}_\mathbf{u} \in \mathbb{R}^{N_0|z_u|\times N_0|U|}$, $\mathbf{w_s} \in \mathbb{R}^{N_0|z_s|}$ and $\mathbf{w_u} \in \mathbb{R}^{N_0|z_u|}$ as follows:

$$\begin{gathered} \mathcal{Q} = \begin{bmatrix} \mathbf{Q} & 0 & \dots & 0 \\ 0 & \mathbf{Q} & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \mathbf{Q} \end{bmatrix} \quad \mathcal{R} = \begin{bmatrix} \mathbf{R} & 0 & \dots & 0 \\ 0 & \mathbf{R} & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \mathbf{R} \end{bmatrix} \quad \mathcal{Z}_\mathbf{s} = \begin{bmatrix} \mathbf{Z_s} & 0 & \dots & 0 \\ 0 & \mathbf{Z_s} & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \mathbf{Z_s} \end{bmatrix} \\ \mathcal{Z}_\mathbf{u} = \begin{bmatrix} \mathbf{Z_u} & 0 & \dots & 0 \\ 0 & \mathbf{Z_u} & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \mathbf{Z_u} \end{bmatrix} \quad \mathbf{w_s} = \begin{bmatrix} \mathbf{z_s} \\ \mathbf{z_s} \\ \dots \\ \mathbf{z_s} \end{bmatrix} \quad \mathbf{w_u} = \begin{bmatrix} \mathbf{z_u} \\ \mathbf{z_u} \\ \dots \\ \mathbf{z_u} \end{bmatrix} \end{gathered} \tag{13}$$

we can obtain a very compact formulation for problem (8) using an MPC approach as follows:

$$\begin{aligned} arg \min_{\mathbf{u}} \quad & \mathbf{s}^T\mathcal{Q}\mathbf{s} + \mathbf{u}^T\mathcal{R}\mathbf{u} \\ s.t. \quad & \mathbf{s} = \mathcal{B}\mathbf{u} + \mathcal{A}\mathbf{s}(\mathbf{0}) \\ & \mathcal{Z}_\mathbf{s}\mathbf{s} \leq \mathbf{w_s} \\ & \mathcal{Z}_\mathbf{u}\mathbf{u} \leq \mathbf{w_u} \end{aligned} \tag{14}$$

which can be further simplified by noting that we can replace $\mathbf{s}$ by its definition using the transition function in the restrictions, to obtain:

$$\begin{aligned} arg \min_{\mathbf{u}} \quad & \frac{1}{2}\mathbf{u}^T\left(\mathcal{B}^T\mathcal{Q}\mathcal{B}+\mathcal{R}\right)\mathbf{u}+\mathbf{s}(\mathbf{0})^T\mathcal{A}^T\mathcal{Q}\mathcal{B}\mathbf{u} \\ s.t. \quad & \begin{bmatrix}\mathcal{Z}_\mathbf{s}\mathcal{B}\\ \mathcal{Z}_\mathbf{u}\end{bmatrix}\mathbf{u} \leq \begin{bmatrix}\mathbf{w_s}-\mathcal{Z}_\mathbf{s}\mathcal{A}\mathbf{s}(\mathbf{0})\\ \mathbf{w_u}\end{bmatrix} \end{aligned} \tag{15}$$

Note that (15) is a Quadratic Program over the control vector $\mathbf{u}$, that can be solved using standard optimization techniques, available in many software packages, such as SLSQP (Sequential Least Squares Quadratic Programming) or L-BFGS-B (Limited-memory Broyden–Fletcher–Goldfarb–Shanno Bounded) [32]. However, note that there are two very important details that affect our controller: the first one is that we have derived it assuming that there were no disturbances. In case that there are disturbances, Equation (11) would not be linear, as our disturbances are not linear, and hence, we would not be able to obtain a convenient problem formulation as (15). The second one is that we assume known the model transition equation, i.e., $\mathbf{A}$ and $\mathbf{B}$. In order to address both problems, we propose an estimator of the model and the disturbances that allows correcting its effect in the next Section.

3. Estimation of the Model and Disturbances

In the previous Section, we have introduced a general model for the navigation of an AUV, and we have noted that knowledge of the transition function $f(\mathbf{s_n}, \mathbf{u_n})$ is needed both for the estimation and the controller blocks, since both Kalman Filters and MPC controllers require that knowledge. However, in real life applications, we frequently do not know the exact transition function, and thus, we need to estimate it. There are several approaches possible [33], but a very convenient one is to approximate the model as linear (1). Note that this means that we are doing a first order approximation to the real transition model $f(\mathbf{s_n}, \mathbf{u_n})$ if f is non-linear, but in many cases, as our simulations will show, this approximation suffices [21,22]. Moreover, a linear model is required in order to use an MPC controller as the one in (15) [34].

3.1. Linear Transition Model Estimation

Let us assume that we want to estimate the transition model using a linear model. For now, we consider that there are no disturbances, so the model that we want to estimate is:

$$\hat{\mathbf{s}}_{\mathbf{n+1}} = \hat{\mathbf{A}}\hat{\mathbf{s}}_\mathbf{n} + \hat{\mathbf{B}}\mathbf{u_n} \tag{16}$$

where we note that this model is an estimation of the true transition model (1), where all unknown variables are noted using the hat notation. We only have access to a set of M samples $(\hat{\mathbf{s}}_\mathbf{n}, \mathbf{u_n}, \hat{\mathbf{s}}_{\mathbf{n+1}})_{n=0}^M$ collected during past interactions of the AUV, where we recall that $\hat{\mathbf{s}}_\mathbf{n}$ is the output of the estimation block chosen (see Figure 1), so we want to estimate $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$, the transition model matrices. In order to obtain $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$, we can use several approaches, ranging from the Least Squares regression, to more complex methods such as LWPR [23,24], XCSF [25,26] or ISSGPR [27]. In this paper, we focus on the Least Squares regression [35], as it only involves using linear algebra, and hence, it is very efficient. This approach involves solving the next optimization problem:

$$\min_{\hat{\mathbf{A}},\hat{\mathbf{B}}} \sum_{n=0}^{M} \left|\left|\hat{\mathbf{A}}\hat{\mathbf{s}}_\mathbf{n} + \hat{\mathbf{B}}\mathbf{u_n} - \hat{\mathbf{s}}_{\mathbf{n+1}}\right|\right|^2 \tag{17}$$

where $||v||$ denotes the norm-2 of the vector v. We note that $\hat{\mathbf{A}}\hat{\mathbf{s}}_\mathbf{n} + \hat{\mathbf{B}}\mathbf{u_n} = \hat{\mathbf{s}}_{\mathbf{n+1}}$ should not be an undetermined system, so we need to have a number of samples M sufficiently large (otherwise, we would have to use a regularizer or use prior information on the matrices). If we define $\hat{\mathbf{a}}_\mathbf{i}^T$ as the i-th row of the $\hat{\mathbf{A}}$ matrix, and $\hat{\mathbf{b}}_\mathbf{i}^T$ as the i-th row of the $\hat{\mathbf{B}}$ matrix, and define $\mathbf{u} = (\mathbf{u_0}, ..., \mathbf{u_{M-1}})^T$, $\hat{\mathbf{s}} = (\hat{\mathbf{s}}_\mathbf{0}, ..., \hat{\mathbf{s}}_\mathbf{M-1})^T$ and $\hat{\mathbf{s}}' = (\hat{\mathbf{s}}_\mathbf{1}, ..., \hat{\mathbf{s}}_\mathbf{M})^T$, we can rewrite problem (17) as:

$$\min_{\hat{\mathbf{a}}_\mathbf{i}, \hat{\mathbf{b}}_\mathbf{i}} \sum_i \left(\hat{\mathbf{a}}_\mathbf{i}^T \hat{\mathbf{s}} + \hat{\mathbf{b}}_\mathbf{i} \mathbf{u} - \hat{\mathbf{s}}' \right)^T \left(\hat{\mathbf{a}}_\mathbf{i}^T \hat{\mathbf{s}} + \hat{\mathbf{b}}_\mathbf{i} \mathbf{u} - \hat{\mathbf{s}}' \right) \tag{18}$$

which is separable in i, which means that we can solve for each $(\hat{\mathbf{a}}_\mathbf{i}, \hat{\mathbf{b}}_\mathbf{i})$ separately, and note that given i, we have a standard Least Squares problem whose analytical solution is:

$$\begin{bmatrix} \hat{\mathbf{a}}_\mathbf{i} \\ \hat{\mathbf{b}}_\mathbf{i} \end{bmatrix} = \left(\begin{bmatrix} \hat{\mathbf{s}} \\ \mathbf{u} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{s}} \\ \mathbf{u} \end{bmatrix}^T \right)^{-1} \begin{bmatrix} \hat{\mathbf{s}} \\ \mathbf{u} \end{bmatrix} \hat{\mathbf{s}}' \tag{19}$$

Note that, in order to obtain (19), we only need linear algebra operations. The complexity depends on the inversion of a square matrix with dimension $M(|S| + |U|)$. If inverting the matrix poses a problem in terms of computational complexity, we could either reduce the number of samples M, or use Recursive Least Squares [36] to update the estimates.

3.2. Linear Transition Model Estimator with Disturbances

In the previous estimator, we assumed that there were no disturbances. However, this need not be true in real settings, as disturbances are part of the underwater medium. This means that the linear model estimated using (16) does not include the term due to disturbances that appears in (1). However, if we knew the disturbance model, we could obtain $\mathbf{d} = d(\hat{\mathbf{s}}) = (d(\hat{\mathbf{s}}_\mathbf{0}), d(\hat{\mathbf{s}}_\mathbf{1}), ..., d(\hat{\mathbf{s}}_\mathbf{M}))^T \in \mathbb{R}^{M|S|}$ as the effect of the disturbances and obtain a linear model that accounts for the disturbances by solving:

$$\min_{\hat{\mathbf{a}}_\mathbf{i}, \hat{\mathbf{b}}_\mathbf{i}} \sum_i \left(\hat{\mathbf{a}}_\mathbf{i}^T \hat{\mathbf{s}} + \hat{\mathbf{b}}_\mathbf{i} \mathbf{u} - (\hat{\mathbf{s}}' - \mathbf{d}) \right)^T \left(\hat{\mathbf{a}}_\mathbf{i}^T \hat{\mathbf{s}} + \hat{\mathbf{b}}_\mathbf{i} \mathbf{u} - (\hat{\mathbf{s}}' - \mathbf{d}) \right) \tag{20}$$

whose solution is:

$$\begin{bmatrix} \hat{\mathbf{a}}_\mathbf{i} \\ \hat{\mathbf{b}}_\mathbf{i} \end{bmatrix} = \left(\begin{bmatrix} \hat{\mathbf{s}} \\ \mathbf{u} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{s}} \\ \mathbf{u} \end{bmatrix}^T \right)^{-1} \begin{bmatrix} \hat{\mathbf{s}} \\ \mathbf{u} \end{bmatrix} (\hat{\mathbf{s}}' - \mathbf{d}) \tag{21}$$

where we note that it is possible to include the effect of the disturbance, but we need to know the disturbance model in order to obtain $\mathbf{d}$. In other words, this means knowing both the type of disturbance (i.e., swirl, current or constant field) and its parameters. In general, we cannot assume that we have this knowledge, so we propose estimating them as well.

3.3. Disturbance Estimator

In order to estimate the disturbances, we propose using a Least Squares approach again. However, due to the fact that the disturbances functions in (2), (3) and (4) are non-linear, the solutions now become more complex. Again, let us assume that we have access to the same set of M samples $(\hat{\mathbf{s}}_\mathbf{n}, \mathbf{u_n}, \hat{\mathbf{s}}_\mathbf{n+1})_{n=0}^M$ that we used for estimating the model. First, by exploiting the knowledge of the transition function model (1), we can estimate the disturbance $\hat{\mathbf{d}}(\hat{\mathbf{s}}_\mathbf{n})$ as:

$$\hat{\mathbf{d}}(\hat{\mathbf{s}}_\mathbf{n}) = \hat{\mathbf{s}}_\mathbf{n+1} - \hat{\mathbf{A}} \hat{\mathbf{s}}_\mathbf{n} - \hat{\mathbf{B}} \hat{\mathbf{u}}_\mathbf{n} \tag{22}$$

and by using the knowledge that the disturbances affect the acceleration in our model, then we have that $\hat{\mathbf{d}}(\hat{\mathbf{s}}_\mathbf{n}) = (\hat{d}_{x,n}, \hat{d}_{y,n})$. Moreover, we can use these values to estimate the parameters $\mathbf{p}$ of the disturbance as:

$$\min_{\mathbf{p}} \sum_{n=0}^{M} \left(d_{x,n}(\hat{\mathbf{s}}_\mathbf{n}, \mathbf{p}) - \hat{d}_{x,n} \right)^2 + \left((d_{x,n}(\hat{\mathbf{s}}_\mathbf{n}, \mathbf{p}) - \hat{d}_{y,n} \right)^2 \tag{23}$$

where we note that $d_{x,n}(\hat{\mathbf{s}}_\mathbf{n}, \mathbf{p})$ and $d_{y,n}(\hat{\mathbf{s}}_\mathbf{n}, \mathbf{p})$ correspond to the disturbance functions in (2), (3) and (4), but for the sake of clarity we have also make explicit the dependence of the disturbance with $\mathbf{p}$, the parameter vector of the disturbances. Let us see how to solve this problem for each of the disturbances presented.

3.3.1. Constant Field

In case of a constant field, by replacing (2) in (23), we obtain the following problem:

$$\min_{k_1,\alpha} \sum_{n=0}^{M} \left(k_1 \cos\alpha - \hat{d}_{x,n}\right)^2 + \left(k_1 \sin\alpha - \hat{d}_{y,n}\right)^2 \tag{24}$$

and this problem has an analytical solution by computing its gradient with respect to k_1 and α, and then equalling to 0 to obtain:

$$\alpha = \arctan\left(\frac{\sum_{n=0}^{M} \hat{d}_{x,n}}{\sum_{n=0}^{M} \hat{d}_{y,n}}\right) \quad k_1 = \frac{\cos\alpha \sum_{n=0}^{M} \hat{d}_{x,n} + \sin\alpha \sum_{n=0}^{M} \hat{d}_{y,n}}{M+1} \tag{25}$$

3.3.2. Currents

In case of a horizontal current, we replace (3) in (23) to obtain the following problem for a horizontal current:

$$\min_{k_2,y_0,\omega} \sum_{n=0}^{M} \left(k_2 \cdot e^{-\dfrac{(\hat{y}_n - y_0)^2}{\omega^2}} - \hat{d}_{x,n}\right)^2 + \left(\hat{d}_{y,n}\right)^2 \tag{26}$$

whose gradients are:

$$\begin{aligned}
\nabla_{k_2} &= \sum_{n=0}^{M} 2 \cdot \kappa \cdot \left(k_2 \cdot \kappa - \hat{d}_{x,n}\right) \\
\nabla_{y_0} &= \sum_{n=0}^{M} \frac{4 \cdot k_2 \cdot \kappa \cdot (\hat{y}_n - y_0)}{\omega^2} \left(k_2 \cdot \kappa - \hat{d}_{x,n}\right) \\
\nabla_{\omega} &= \sum_{n=0}^{M} \frac{4 \cdot k_2 \cdot \kappa \cdot (\hat{y}_n - y_0)^2}{\omega^3} \left(k_2 \cdot \kappa - \hat{d}_{x,n}\right) \\
\kappa &= e^{-\dfrac{(\hat{y}_n - y_0)^2}{\omega^2}}
\end{aligned} \tag{27}$$

and since (27) does not have an analytical solution in terms of elementary functions, we cannot obtain an analytical solution as in the constant field case. In order to address this, we propose using the gradient expression and a first-order optimization method to arrive at a solution for the parameters. Namely, we use L-BFGS [32], a quasi-Newton method known by its fast convergence rate by making use only of first-order information, e.g., the gradient (27). Note that the computational complexity can be adjusted by setting a maximum number of iterations.

The case for vertical currents is analog: we replace (3) in (24) to obtain the following optimization problem:

$$\min_{k_2,x_0,\omega} \sum_{n=0}^{M} \left(\hat{d}_{x,n}\right)^2 + \left(k_2 \cdot e^{-\dfrac{(\hat{x}_n - x_0)^2}{\omega^2}} - \hat{d}_{y,n}\right)^2 \tag{28}$$

whose gradients are:

$$\begin{aligned}
\nabla_{k_2} &= \sum_{n=0}^{M} 2 \cdot \kappa \cdot \left(k_2 \cdot \kappa - \hat{d}_{y,n} \right) \\
\nabla_{x_0} &= \sum_{n=0}^{M} \frac{4 \cdot k_2 \cdot \kappa \cdot (\hat{x}_n - x_0)}{\omega^2} \left(k_2 \cdot \kappa - \hat{d}_{y,n} \right) \\
\nabla_{\omega} &= \sum_{n=0}^{M} \frac{4 \cdot k_2 \cdot \kappa \cdot (\hat{x}_n - x_0)^2}{\omega^3} \left(k_2 \cdot \kappa - \hat{d}_{y,n} \right) \\
\kappa &= e^{-\frac{(\hat{x}_n - x_0)^2}{\omega^2}}
\end{aligned} \tag{29}$$

and again, we do not have an analytical solution, but we can use any first-order method to estimate the parameters.

3.3.3. Swirls

Finally, in case of a swirl, we again replace (4) in (24) to obtain:

$$\min_{k_3, x_0, y_0} \sum_{n=0}^{M} \left(\frac{k_3(\hat{y}_n - y_0)}{\sqrt{(\hat{x}_n - x_0)^2 + (\hat{y}_n - y_0)^2}} - \hat{d}_{x,n} \right)^2 + \left(\frac{-k_3(\hat{x}_n - x_0)}{\sqrt{(\hat{x}_n - x_0)^2 + (\hat{y}_n - y_0)^2}} - \hat{d}_{y,n} \right)^2 \tag{30}$$

whose gradients are:

$$\begin{aligned}
\nabla_{k_3} &= \sum_{n=0}^{M} \frac{2 \cdot (\hat{y}_n - y_0)}{\kappa} \left(\frac{k_3(\hat{y}_n - y_0)}{\kappa} - \hat{d}_{x,n} \right) + \frac{2 \cdot (\hat{x}_n - x_0)}{\kappa} \left(\frac{k_3(\hat{x}_n - x_0)}{\kappa} - \hat{d}_{y,n} \right) \\
\nabla_{x_0} &= \sum_{n=0}^{M} \frac{2 \cdot k_3 \cdot (\hat{y}_n - y_0)(\hat{x}_n - x_0)}{\kappa^3} \left(\frac{k_3 \cdot (\hat{y}_n - y_0)}{\kappa} - \hat{d}_{x,n} \right) \\
&\quad + \frac{2 \cdot k_3}{\kappa^2} \left(\kappa - \frac{(x_0 - \hat{x}_n)(\hat{x}_n - x_0)}{\kappa} \right) \left(\frac{k_3 \cdot (\hat{x}_n - x_0)}{\kappa} - \hat{d}_{y,n} \right) \\
\nabla_{y_0} &= \sum_{n=0}^{M} \frac{2 \cdot k_3}{\kappa^2} \left(-\kappa - \frac{(\hat{y}_n - y_0)(\hat{y}_n - y_0)}{\kappa} \right) \left(\frac{k_3 \cdot (\hat{y}_n - y_0)}{\kappa} - \hat{d}_{x,n} \right) \\
&\quad + \frac{2 \cdot k_3 \cdot (x_0 - \hat{x}_n)(\hat{y}_n - y_0)}{\kappa^3} \left(\frac{k_3 \cdot (x_0 - \hat{x}_n)}{\kappa} - \hat{d}_{y,n} \right) \\
\kappa &= \sqrt{(\hat{x}_n - x_0)^2 + (\hat{y}_n - y_0)^2}
\end{aligned} \tag{31}$$

and again, we do not have an analytical solution, but we can use any first-order method to estimate the parameters.

3.4. Joint Transition and Disturbance Estimator

As we have mentioned in the previous Sections, it is likely that in a real-world setting we would not know neither the transition model nor the disturbances. Hence, we propose using the tools developed in the previous Sections in order to provide an estimation of both using only a set of M samples $(\hat{\mathbf{s}}_\mathbf{n}, \mathbf{u}_\mathbf{n}, \hat{\mathbf{s}}_{\mathbf{n}+\mathbf{1}})_{n=0}^{M}$ collected during the interactions of the AUV with the environment.

First, we propose an exploration period, where the agent can use any controller (such as a random one), in order to collect an initial set of samples $(\hat{\mathbf{s}}_\mathbf{n}, \mathbf{u}_\mathbf{n}, \hat{\mathbf{s}}_{\mathbf{n}+\mathbf{1}})$. Then, when enough samples have been gathered, we start with the estimation phase, where we use these samples in order to estimate the transition model, the disturbances, or both.

In order to estimate the disturbances, we need both a model of how the disturbances affect the transitions (e.g., (1)) and an estimate of the transition model. With these values, we propose solving problems (24), (26), (28) and (30) in order to obtain a set of parameters for each of the disturbances models we have proposed. In order to select the disturbance that best fits the available data, we select the model that provides a lower objective function for the optimal $\mathbf{p}$. Computationally, this means solving three optimization problems using L-BFGS, as the only disturbance with an analytical solution is the Constant Field. An algorithmic view of the procedure can be observed in Algorithm 1. Since we are using all the disturbances, we name this method Full Disturbance Estimator (FDE).

Algorithm 1 Joint transition and Full Disturbance Estimator (FDE)

```
Input: $(\hat{\mathbf{s}}_\mathbf{n}, \mathbf{u}_\mathbf{n}, \hat{\mathbf{s}}_{\mathbf{n+1}})_{n=0}^{M}$
 1: while Estimation update required do
 2:   if Transition model $(\hat{\mathbf{A}}, \hat{\mathbf{B}})$ has been estimated in a previous update then
 3:     Obtain $\hat{\mathbf{d}}(\hat{\mathbf{s}}_\mathbf{n})$ using (22) and $(\hat{\mathbf{A}}, \hat{\mathbf{B}})$
 4:     for Each disturbance model do
 5:       Estimate the parameters $\mathbf{p}$ of the disturbance using (25), (27), (29) or (31)
 6:       Obtain the error value for the disturbance using (24), (26), (28) or (30) and the
          obtained $\mathbf{p}$
 7:     Compute $\mathbf{d}$ using the $\mathbf{p}$ of the disturbance that yields the minimum error
 8:     Estimate $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$ using $\mathbf{d}$ and (21)
 9:   else
10:     Estimate $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$ using (19)
Output: $\hat{\mathbf{A}}, \hat{\mathbf{B}}$
```

However, it may be the case that the computational capacity of the AUV is highly restricted, and the computation of the disturbances that use a first-order optimization method may be unaffordable. We propose a second disturbance estimator, where we estimate the disturbance using only the Constant Field model, whose solution is analytical and computationally cheap. Note that the Constant Field estimation can be seen as a local approximation to any disturbance, so we are introducing a certain approximation error in order to have a less computationally expensive procedure. We name this estimator Constant Disturbance Estimator (CDE), to differentiate it from the FDE. An algorithmic view of the procedure can be observed in Algorithm 2.

Algorithm 2 Joint transition and Constant Disturbance Estimator (CDE)

```
Input: $(\hat{\mathbf{s}}_\mathbf{n}, \mathbf{u}_\mathbf{n}, \hat{\mathbf{s}}_{\mathbf{n+1}})_{n=0}^{M}$
 1: while Estimation update required do
 2:   if Transition model $(\hat{\mathbf{A}}, \hat{\mathbf{B}})$ has been estimated in a previous update then
 3:     Obtain $\hat{\mathbf{d}}(\hat{\mathbf{s}}_\mathbf{n})$ using (22) and $(\hat{\mathbf{A}}, \hat{\mathbf{B}})$
 4:     Estimate the parameters $\mathbf{p}$ of the Constant Field disturbance using (25)
 5:     Compute $\mathbf{d}$ using the $\mathbf{p}$ of the Constant Field disturbance obtained
 6:     Estimate $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$ using $\mathbf{d}$ and (21)
 7:   else
 8:     Estimate $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$ using (19)
Output: $\hat{\mathbf{A}}, \hat{\mathbf{B}}$
```

4. Simulations

We now turn to evaluate the results that the proposed estimation methods have on the performance of an AUV navigation system. In order to do so, we implement the simulator depicted in Figure 1 and subject it to a set of tests in order to assess the gains that the estimators of Section 3 provide.

4.1. Testbench Description

Let us start by describing the testbench we will use in our simulations. Note that we follow the three main blocks described in Figure 1.

4.1.1. Environment

Inspired by [12], we consider an AUV that moves at a constant depth in a 2-D Cartesian space, where x and y denote the horizontal and vertical coordinates of the plane (in meters), respectively, while v_x and v_y denote the velocities in x and y, respectively (in meters per second). We consider that the AUV controls its acceleration in each coordinate, which we denote as a_x and a_y (in meters per seconds squared). We also consider a friction term modeled by a parameter k_f that depends on the velocity, which correspond to a low velocity setting [37]. Note that this friction term prevents that the velocity grows unbounded [38]. Finally, the disturbances affect the acceleration, and have two components d_x and d_y, respectively, with acceleration units. By considering Δ (in seconds) as the time step and n as the time index, we have the following discrete-time dynamical system:

$$\begin{cases} x_{n+1} = x_n + \Delta \cdot v_{x,n} \\ y_{n+1} = y_n + \Delta \cdot v_{y,n} \\ v_{x,n+1} = v_{x,n} + \Delta \cdot (a_{x,n} + d_{x,n} - k_f \cdot v_{x,n}) \\ v_{y,n+1} = v_{y,n} + \Delta \cdot (a_{y,n} + d_{y,n} - k_f \cdot v_{y,n}) \end{cases} \tag{32}$$

and if we define $\mathbf{s_n} = (x_n, y_n, v_{x,n}, v_{y,n})^T$ as the state, $\mathbf{u_n} = (a_{x,n}, a_{y,n})^T$ as the control, and $\mathbf{d}(\mathbf{s_n}) = (0, 0, \Delta \cdot d_{x,n}(\mathbf{s_n}), \Delta \cdot d_{y,n}(\mathbf{s_n}))^T$ as the disturbance vector, we can rewrite (32) as follows:

$$\mathbf{s_{n+1}} = f(\mathbf{s_n}, \mathbf{u_n}) = \mathbf{A}\mathbf{s_n} + \mathbf{B}\mathbf{u_n} + \mathbf{d}(\mathbf{s_n}) \tag{33}$$

which is the formulation followed in this work (1), where matrices $\mathbf{A}$ and $\mathbf{B}$ are defined as:

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & \Delta & 0 \\ 0 & 1 & 0 & \Delta \\ 0 & 0 & 1 - k_f \cdot \Delta & 0 \\ 0 & 0 & 0 & 1 - k_f \cdot \Delta \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \Delta & 0 \\ 0 & \Delta \end{bmatrix} \tag{34}$$

Additionally, note that $|S| = 4$ and $|U| = 2$. We simulate using $\Delta = 0.1$ and $k_f = 0.5$.

Regarding the disturbances, we initialize their parameters randomly. Parameters k_1, k_2 and k_3 reflect the intensity of the disturbance, so they are sampled from a uniform distribution in the range $[-1.5, 1.5]$, where the value is chosen so that the disturbance does not exceed the acceleration capacity of the AUV, but affects its trajectory. Parameters x_0 and y_0 denote the position of the disturbances, and are sampled from a Gaussian distribution of mean 0 and standard deviation 3, so that they cover a wide set of points around the origin. The angle α is sampled from a uniform distribution in the range $[0, 2\pi)$, and the current width ω is sampled from a Gaussian distribution with mean 5 and standard deviation 1.

Finally, we consider two cases regarding the observation model: a full observation case where $\mathbf{o_n} = \mathbf{s_n}$, and a partial observation case following (7), where the Gaussian noise added to each state component is zero-mean and with standard deviation 0.1. All the experiments have been implemented using the programming language Python.

4.1.2. Estimation

In our testbench, the estimation block has two main components: an state estimator that implements (6), and the transition estimator described in Section 3. Regarding the transition estimator, we consider the following cases:

- The case with full knowledge of the transition model: in this case, the matrices $\mathbf{A}$ and $\mathbf{B}$ from (34) are assumed known. Note that this is the less realistic case, as this knowledge seldom happens, but it serves as the best case baseline to which we can compare our results.
- The case in which the transition model is estimated without the help of the disturbance estimator. In this case, we estimate $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$ using (19).
- The case in which the transition model is estimated with the help of the disturbance estimator. In this case, we estimate $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$ using (21), as described in Algorithm 1 in case of using FDE for the disturbances, and follow Algorithm 2 in case of using CDE for the disturbances.

Regarding the state estimator, which is in charge of obtaining $\hat{\mathbf{s}}_\mathbf{n}$ from $\mathbf{o}_\mathbf{n}$, we consider the following two cases:

- No state estimator, and hence, $\hat{\mathbf{s}}_\mathbf{n} = \mathbf{o}_\mathbf{n}$. This is the case when we have full observability, but when there is partial observability, this estate estimator introduces error.
- Use as state estimator a KF, which relies on the knowledge of $\mathbf{A}$ and $\mathbf{B}$ in case that we have full knowledge, or on the estimated $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$ in case that we use the transition model estimator.

Note that the output of the estimation block is both the estimated state $\hat{\mathbf{s}}_\mathbf{n}$ and the transition model (which is needed by the controller block). The two elements of the estimation block, which are the state and transition estimator, are coupled, and as one improves, the other improves too; note that the transition estimator uses the state estimation to return an estimation of the model, and the estimated model is used by the KF to compute the state estimation.

4.1.3. Controller

We consider that our controller is an MPC which solves (15), where its inputs are given by the estimation block: the initial state $\mathbf{s}(\mathbf{0})$ (or its estimation) and the transition model ($\mathbf{A}$ and $\mathbf{B}$ in case that we have full knowledge, or $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$ otherwise). Regarding the time horizon N_0 that we use to predict using MPC, we set $N_0 = 20$. We also need to define the MPC cost matrices $\mathbf{Q}$ and $\mathbf{R}$, which are:

$$\mathbf{Q} = \begin{bmatrix} 100 & 0 & 0 & 0 \\ 0 & 100 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \mathbf{R} = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix} \tag{35}$$

where we note that $\mathbf{Q}$ and $\mathbf{R}$ are designed to drive the AUV as fast as possible to the origin, as the highest costs are on the positions. Moreover, the restrictions considered for the state $(\mathbf{Z_s}, \mathbf{z_s})$ are $-10 \le x, y \le 10$ and $-2 \le v_x, v_y \le 2$, and for the controls $(\mathbf{Z_u}, \mathbf{z_u})$ are $-2 \le \mathbf{u_n} \le 2$. Note that these restrictions mean that the AUV has a limited position, velocity and acceleration.

4.1.4. Simulation Conditions

Let us now describe the general procedure used for testing. First, for each disturbance considered (no disturbance, a constant field, a current, or a swirl), we randomly generate 100 initial states (i.e., positions and velocities), and for each initial state, we also generate a distribution parameter $\mathbf{p}$ randomly, as indicated in Section 4.1.1. Then, for each initial state and disturbance, we run an MPC controller for each combination of model knowledge (full knowledge, transition estimator, or transition and disturbance estimator for FDE and

CDE) and state estimator (none or KF), and compare the costs obtained for each one. We emphasize that all combinations of model knowledge and state estimator share the same initial state and disturbance parameters, so that the results obtained are directly comparable (i.e., they have been obtained using the same conditions).

In order to compare the costs, we consider that the AUV has reached the origin when the distance to the origin is smaller than 1. We also set a maximum number of 500 time steps: if a simulation reaches 500 steps without having reached the origin, we consider that the execution timed out. The results show that only a 2% of the simulations timed out, while the 98% of the cases the AUV reached the target.

In order to further simulate a computationally constrained system, we update the estimations once every five iterations of the controller, and in the disturbance estimation, we used the previous parameter estimation as initial point for the optimization algorithm in case of FDE. Additionally, to filter the estimation noise in the transition model, we set the entries of $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$ smaller than 0.01 to 0.

4.2. Transition Estimator Accuracy

Let us first analyze the results of the disturbance estimator and its performance, both using FDE and CDE. The summarized results can be observed in Table 1 for the FDE case and Table 2 for the CDE case, where the observation model can be either total (T) or partial (P), and the state estimator can be none (N) or a KF. For each combination of observation model, state estimator and disturbance, we obtain the following conclusions:

- After each simulation, we measured the latest disturbance estimated, and in the column "Disturbance estimated", we reflect the proportion of times that the disturbance estimated was, in this order, a swirl, a horizontal current, a vertical current, and a constant field, in case of using FDE (for CDE, we remind that we only estimate a Constant Field). For ease of visualization, we show in bold the actual disturbance for each case. Note that the actual disturbance is always the one that is estimated most often, with proportions higher than 90% in case of the Constant Field and the swirl, and 60% in case of the currents. We note that the drop in the accuracy of the currents can be explained because they can be confused with a Constant Field if we are far from the current center, as can be seen in Figure 2. Thus, the accuracy of the FDE model to detect the right disturbance is very high, regardless of the observation model and the state estimation used.
- Every time that the FDE model obtained the right disturbance, as seen in the previous point, we also computed the relative error of the estimated parameters $\hat{\mathbf{p}}$ compared to the actual parameters $\mathbf{p}$. In case that there was no actual disturbance, the error is the absolute value of the k_1, k_2 or k_3 parameter estimated (e.g., the strength of the disturbance estimated, related to the error committed). In this metric, we can appreciate the effect of the observation model, as the error is considerably lower if of total observation than when there is partial observation. This is due to the fact that the noise in the observation translates to inaccuracies in the disturbance estimator, and this effect is particularly noticeable in the currents, which contain the highest errors.
- For both FDE and CDE, we have also measured the norm between the estimated model and the real model as:

$$\text{Error gain} = 100 \cdot \frac{(||\mathbf{A} - \hat{\mathbf{A}}_{\mathbf{T}}||^2 + ||\mathbf{B} - \hat{\mathbf{B}}_{\mathbf{T}}||^2) - (||\mathbf{A} - \hat{\mathbf{A}}_{\mathbf{TD}}||^2 + ||\mathbf{B} - \hat{\mathbf{B}}_{\mathbf{TD}}||^2)}{||\mathbf{A} - \hat{\mathbf{A}}_{\mathbf{T}}||^2 + ||\mathbf{B} - \hat{\mathbf{B}}_{\mathbf{T}}||^2} \tag{36}$$

  where the subscript T stands for the case of using only the transition model estimator, and TD for the case of using also a disturbance estimator (e.g., FDE or CDE). When this gain is positive, it means that the disturbance estimator provides a more accurate model estimator than the transition estimator alone (thus, higher is better). Note that this gain is always positive (or very close to 0) in FDE, meaning that using FDE provides a consistent improvement in the transition estimator. The gains are particularly dramatic in case of total observability, where the gain is around 90%, while when there is noise, the maximum gain reduces to around 8%. This is due to the presence of noise, and it is to be expected, but observe that the disturbance estimator still manages to improve the model estimation. In case of CDE, it is remarkable that the error gains are similar to the FDE case, except for the swirls: as seen in Figure 2, the swirls are the most non-linear disturbance, and hence, CDE commits a higher error.

- We have also measured the number of steps that MPC takes to reach the origin, and we have defined the steps gain as:

$$\text{Steps gain} = 100 \cdot \frac{L_T - L_{TD}}{L_T} \tag{37}$$

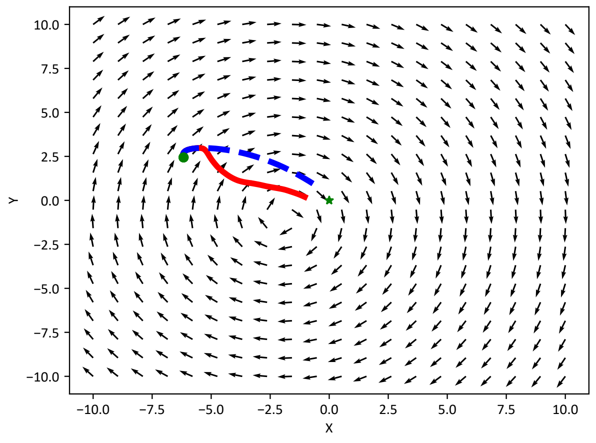
where L_T is the average number of steps that took the transition model alone to reach the origin, and L_{TD} is the average number of steps that took the transition model with disturbance estimator (e.g., FDE or CDE) to reach the origin. Again, higher is better: a positive number indicates that the transition estimator alone took more steps, which means that the disturbance estimator actually takes less steps to reach the target position, as the model is estimated better. The gain without noise for FDE and CDE is around 45% and reduces to around 15–30% in case that there is observation noise. Note that in case that there is no actual disturbance, the gains are close to 0. Again, the only significant difference between FDE and CDE are in case of having a swirl, where CDE performance degrades significantly.

- We have also measured the simulation time between using a transition estimator alone or with disturbance estimator as

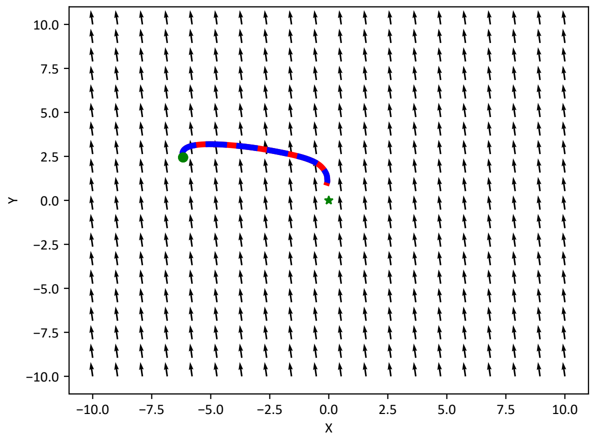
$$\text{Time gain} = 100 \cdot \frac{t_T - t_{TD}}{t_T} \tag{38}$$

where t_T is the average simulation time that took the transition model alone to reach the origin, and t_{TD} is the average simulation time that took the transition model with disturbance estimator (e.g., FDE or CDE) to reach the origin. Again, higher is better, as a positive number indicates that the transition estimator with FDE/CDE took less simulation time than the transition estimator without. Note that for FDE, there are many positive numbers, which indicate that the use of FDE, which translates in less steps to the origin as we have seen, also translates to less simulation time (e.g., less computational load). In other words, the extra computation time of the disturbance estimator is compensated by needing less calls to the optimizer to reach the origin. Additionally, note that the cases with negative gain are correlated to the cases with low step gain. In case of CDE, the gains are dramatic, due to its reduced computational complexity which, nonetheless, allows it to require fewer steps to reach the origin (the exception is, again, the swirl).

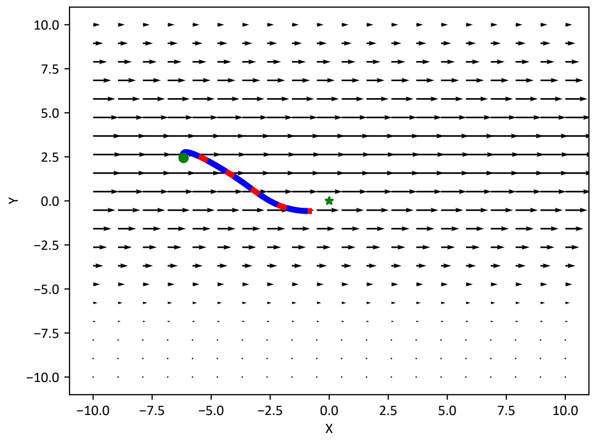
Hence, the results of our simulations show that the use of the proposed disturbance estimators provides solid gains in accuracy, steps needed to reach the target position, and even in computation time. It is very remarkable that the CDE estimator, which only considers a Constant Field, is able to obtain good performance also with currents, but fails with the most non-linear disturbance, which is the swirl. Thus, both methods return solid gains in every metric used.



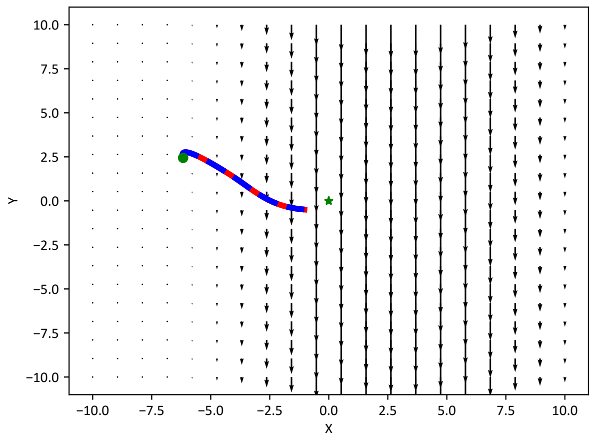
Swirl.



Constant field.



Horizontal current.



Vertical current.

Figure 2. Sample trajectories obtained for different disturbances. In black, we have the disturbance field, the green point is the starting position, and the green star (center) is the target. In all cases, we consider partial observability and use a Kalman filter for state estimation, as well as our proposed transition and disturbance estimator. In red, we show results for the CDE, whereas dashed blue is for the FDE: the differences are only notable on the swirl case, which is consistent with the results shown in Table 3. Note that horizontal and vertical current trajectories look alike, but are different.

Table 1. FDE results, where observation is total (T) or partial (P), and the state estimator used is none (N) or a Kalman filter (KF). For each disturbance, we show the proportion of times that the disturbance estimator detected it as a [Swirl, Horizontal Current, Vertical Current, Constant Field], and in bold, we highlight the highest proportion. Note that, in all cases, the true disturbance is the one detected the highest number of times. For all gains, higher is better.

Obs	State	Disturbance	Disturbance Estimated	*p* Error	Error Gain	Time Gain	Steps Gain
T	N	None	[0.04 0.01 0.03 0.92]	0.00	−0.09	−0.36	0.00
T	N	Swirl	[**0.94** 0.01 0.01 0.04]	8.44	85.51	44.24	48.74
T	N	H. Current	[0.00 **0.86** 0.00 0.14]	8.60	89.33	15.81	43.94
T	N	V. Current	[0.00 0.00 **0.67** 0.33]	12.09	87.52	29.01	44.73
T	N	Constant	[0.00 0.00 0.00 **1.00**]	0.00	93.31	34.79	40.62
P	N	None	[0.00 0.45 0.39 0.16]	3.19	−0.12	−11.47	0.09
P	N	Swirl	[**0.94** 0.02 0.01 0.03]	12.59	3.70	−5.68	15.81

Table 1. *Cont.*

Obs	State	Disturbance	Disturbance Estimated	*p* Error	Error Gain	Time Gain	Steps Gain
P	N	H. Current	[0.00 **0.64** 0.15 0.21]	81.88	6.90	−2.79	30.98
P	N	V. Current	[0.00 0.14 **0.59** 0.27]	118.96	0.49	−3.70	22.69
P	N	Constant	[0.00 0.03 0.03 **0.94**]	0.51	7.85	10.21	25.98
P	KF	None	[0.00 0.33 0.23 0.44]	0.54	0.04	−6.61	0.03
P	KF	Swirl	[**0.93** 0.03 0.02 0.02]	30.37	5.14	0.42	15.97
P	KF	H. Current	[0.02 **0.59** 0.14 0.25]	75.99	6.60	−1.28	28.97
P	KF	V. Current	[0.00 0.12 **0.62** 0.26]	95.53	−0.28	−11.69	16.89
P	KF	Constant	[0.00 0.04 0.00 **0.96**]	1.19	7.51	−25.41	−1.30

Table 2. CDE results, where observation is total (T) or partial (P), and the state estimator used is none (N) or a Kalman filter (KF). For all gains, higher is better. Note that the results are competitive against FDE, but having a significantly lower computational cost (except on the swirl).

Obs	State	Disturbance	Error Gain	Time Gain	Steps Gain
T	N	None	−0.12	−1.42	0.00
T	N	Swirl	−10.77	−26.62	−28.85
T	N	H. Current	53.07	39.32	41.14
T	N	V. Current	57.04	32.91	32.56
T	N	Constant	93.31	52.98	40.62
P	N	None	−0.03	0.38	0.00
P	N	Swirl	−7.58	−37.30	−32.00
P	N	H. Current	6.38	29.63	29.67
P	N	V. Current	2.12	23.15	20.16
P	N	Constant	7.86	31.83	25.98
P	KF	None	0.06	5.47	0.00
P	KF	Swirl	−8.82	−37.73	−37.74
P	KF	H. Current	6.04	23.72	21.09
P	KF	V. Current	1.61	3.42	8.87
P	KF	Constant	7.53	10.39	−1.30

Table 3. Total cost results, where observation is total (T) or partial (P), and the state estimator used is none (N) or a Kalman filter (KF). The cases tested are full knowledge of the model (K), transition estimator without disturbance estimator (T), transition estimator and CDE (TCDE), and transition estimator and FDE (TFDE). In all cases, higher means that the first model is better, and results can be interpreted as percentages. Note that the gains of using both disturbances estimators are significant compared to only using the transition estimator, and may obtain similar results to actually knowing the model.

Obs	State	Disturbance	Gain T/K	Gain TCDE/K	Gain TFDE/K	Gain TCDE/T	Gain TFDE/T
T	N	None	0.00	0.00	0.00	0.00	0.00
T	N	Swirl	−54.33	−149.55	−3.15	−61.70	33.16
T	N	H. Current	−46.24	−2.85	−1.92	29.67	30.30
T	N	V. Current	−59.70	−2.99	−1.71	35.51	36.31
T	N	Constant	−91.37	−0.33	−0.33	47.57	47.57
P	N	None	3.60	3.60	3.62	−0.00	0.02
P	N	Swirl	−0.36	−12.38	1.92	−11.98	2.27
P	N	H. Current	−2.18	2.19	2.36	4.27	4.44
P	N	V. Current	−5.92	3.26	3.55	8.67	8.94
P	N	Constant	−21.40	2.85	2.85	19.98	19.98
P	KF	None	5.93	5.93	5.93	−0.00	0.00
P	KF	Swirl	4.80	−8.85	9.21	−14.34	4.64
P	KF	H. Current	−0.48	5.62	6.40	6.07	6.85
P	KF	V. Current	0.80	5.74	6.52	4.98	5.77
P	KF	Constant	−15.17	10.41	10.41	22.21	22.21

4.3. Cost Gains

We now turn our attention to the total cost that MPC obtains for each simulation. The results can be seen in Table 3, where the gains are the relative costs of the two models being compared (a higher number indicates the first model is better). A positive gain means that the first method is better, and a negative gain means that the second method is better, and the gain can be interpreted as a percentage. The different models being compared are:

- K: the model with full knowledge of the model, e.g., the actual model matrices **A** and **B** are known, but without knowing nor estimating the disturbances.
- T: Transition Estimator alone, i.e., without any disturbance estimator.
- TFDE: Transition Estimator and FDE model for estimating the disturbances.
- TCDE: Transition Estimator and CDE model for estimating the disturbances.

The results obtained in Table 3 can be summarized as follows:

- If we compare the transition estimator alone with knowing the model, we see that the latter has a clear advantage in most cases, specially when there is no noise or there is a constant disturbance. However, the advantage of knowing the model vanishes if we also estimate the disturbances. This is a very important conclusion, as in terms of cost, it is similar to know the model than to estimate it using our proposed disturbance estimator. In a real environment, when the model is unknown, our results suggest that estimating the disturbance has a consistent advantage in terms of costs. Moreover, the results from Tables 1 and 2 indicate that this advantage also extends to the computational load.
- The previous conclusion is reinforced by observing that using the disturbance estimator provides a consistent gain against using the transition estimator alone. Note that, in case of estimating the disturbances using CDE, the only case where the gain is negative (e.g., the transition estimator alone is better) is when the disturbance is a swirl, which is to be expected. However, when using FDE, the gain compared to the transition estimator alone is always positive, and ranges goes up to a 47% with perfect observability and up to a 22% improvement in case that there is observation noise.

5. Conclusions

In this work, we have proposed a control method which fills a gap in current methods: it is computationally suitable for devices with low computational capacities, yet it is able to estimate the kinematics (i.e., the transition function) and the disturbances of the underwater medium. We have proposed two different disturbance estimators, one that tries to classify the disturbance as a swirl, current or constant field (FDE) and another that only takes into account constant fields (CDE). We have seen solid gains in both cases compared to the case in which no disturbance is estimated, being able to match the performance of knowing the actual kinematics. Additionally, the increase in the computational load due to the disturbance estimation is compensated by taking fewer steps to reach the target. Hence, the proposed methods are promising for real-life environments, where both disturbances and kinematics are unknown.

There are also several future lines that can be explored from our work. It could be possible to further enhance the performance of the method by including the disturbance information in the optimizer, and hence, take advantage of it when possible. Another possible option would be to test using other disturbances, and checking what is the advantage of using FDE against CDE in these cases. Moreover, it could be possible to extend this work to more complex transition models, such as the ones shown in [39] or [40]. However, note that it would be needed to extend the disturbance models from the 2D models used in this work to 3D ones in order to match the aforementioned works. Finally, we have shown the importance of being able to estimate the disturbance, as the gains provided are solid: it could be possible to use general-purpose approximation methods in order to predict the disturbances in a more general way, although these methods should be computationally efficient in order to match the spirit of this work.

Author Contributions: Conceptualization, J.P. and S.Z.; formal analysis, P.B. and J.P.; funding acquisition, S.Z.; investigation, P.B. and J.P.; methodology, P.B., J.P. and S.Z.; project administration, S.Z.; resources, S.Z.; software, P.B. and J.P.; supervision, S.Z.; validation, P.B., J.P. and S.Z.; visualization, J.P.; writing—original draft, P.B. and J.P.; writing—review and editing, P.B., J.P. and S.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Spanish Ministry of Science and Innovation under the grant PID2020-112502RB-C41 (NAUTILUS).

Institutional Review Board Statement: Not applicable.

Data Availability Statement: Not applicable.

Informed Consent Statement: Not applicable.

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

References

1. Shakhatreh, H.; Sawalmeh, A.H.; Al-Fuqaha, A.; Dou, Z.; Almaita, E.; Khalil, I.; Othman, N.S.; Khreishah, A.; Guizani, M. Unmanned aerial vehicles (UAVs): A survey on civil applications and key research challenges. *IEEE Access* **2019**, *7*, 48572–48634. [CrossRef]
2. Winston, C.; Karpilow, Q. *Autonomous Vehicles: The Road to Economic Growth?*; Brookings Institution Press: Washington, DC, USA, 2020.
3. Sahoo, A.; Dwivedy, S.K.; Robi, P. Advancements in the field of autonomous underwater vehicle. *Ocean Eng.* **2019**, *181*, 145–160. [CrossRef]
4. Paull, L.; Saeedi, S.; Seto, M.; Li, H. AUV navigation and localization: A review. *IEEE J. Ocean. Eng.* **2013**, *39*, 131–149. [CrossRef]
5. Stutters, L.; Liu, H.; Tiltman, C.; Brown, D.J. Navigation technologies for autonomous underwater vehicles. *IEEE Trans. Syst. Man Cybern. Part C (Appl. Rev.)* **2008**, *38*, 581–589. [CrossRef]
6. Schillai, S.M.; Turnock, S.R.; Rogers, E.; Phillips, A.B. Evaluation of terrain collision risks for flight style autonomous underwater vehicles. In Proceedings of the 2016 IEEE/OES Autonomous Underwater Vehicles (AUV), Tokyo, Japan, 6–9 November 2016; pp. 311–318.
7. Zhang, Y.; Liu, X.; Luo, M.; Yang, C. MPC-based 3-D trajectory tracking for an autonomous underwater vehicle with constraints in complex ocean environments. *Ocean Eng.* **2019**, *189*, 106309. [CrossRef]
8. Xiang, X.; Lapierre, L.; Jouvencel, B. Smooth transition of AUV motion control: From fully-actuated to under-actuated configuration. *Robot. Auton. Syst.* **2015**, *67*, 14–22. [CrossRef]
9. Londhe, P.S.; Patre, B. Adaptive fuzzy sliding mode control for robust trajectory tracking control of an autonomous underwater vehicle. *Intell. Serv. Robot.* **2019**, *12*, 87–102. [CrossRef]
10. Yan, Z.; Wang, M.; Xu, J. Robust adaptive sliding mode control of underactuated autonomous underwater vehicles with uncertain dynamics. *Ocean Eng.* **2019**, *173*, 802–809. [CrossRef]
11. Shojaei, K. Three-dimensional neural network tracking control of a moving target by underactuated autonomous underwater vehicles. *Neural Comput. Appl.* **2019**, *31*, 509–521. [CrossRef]
12. Parras, J.; Zazo, S. Robust Deep Reinforcement Learning for Underwater Navigation with Unknown Disturbances. In Proceedings of the ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Toronto, ON, Canada, 6–11 June 2021; pp. 3440–3444.
13. González-García, J.; Gómez-Espinosa, A.; Cuan-Urquizo, E.; García-Valdovinos, L.G.; Salgado-Jiménez, T.; Escobedo Cabello, J.A. Autonomous underwater vehicles: Localization, navigation, and communication for collaborative missions. *Appl. Sci.* **2020**, *10*, 1256. [CrossRef]
14. Hou, X.; Zhou, J.; Yang, Y.; Yang, L.; Qiao, G. Adaptive two-Step bearing-only underwater uncooperative target tracking with uncertain underwater disturbances. *Entropy* **2021**, *23*, 907. [CrossRef] [PubMed]
15. Parras, J.; Apellániz, P.A.; Zazo, S. Deep Learning for Efficient and Optimal Motion Planning for AUVs with Disturbances. *Sensors* **2021**, *21*, 5011. [CrossRef]
16. Yan, Z.; Gong, P.; Zhang, W.; Wu, W. Model predictive control of autonomous underwater vehicles for trajectory tracking with external disturbances. *Ocean Eng.* **2020**, *217*, 107884. [CrossRef]
17. Peng, Z.; Wang, J.; Wang, J. Constrained control of autonomous underwater vehicles based on command optimization and disturbance estimation. *IEEE Trans. Ind. Electron.* **2018**, *66*, 3627–3635. [CrossRef]
18. Liu, S.; Liu, Y.; Wang, N. Nonlinear disturbance observer-based backstepping finite-time sliding mode tracking control of underwater vehicles with system uncertainties and external disturbances. *Nonlinear Dyn.* **2017**, *88*, 465–476. [CrossRef]
19. Kim, J.; Joe, H.; Yu, S.c.; Lee, J.S.; Kim, M. Time-delay controller design for position control of autonomous underwater vehicle under disturbances. *IEEE Trans. Ind. Electron.* **2015**, *63*, 1052–1061. [CrossRef]

20. Kawano, H.; Ura, T. Motion planning algorithm for nonholonomic autonomous underwater vehicle in disturbance using reinforcement learning and teaching method. In Proceedings of the 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292), Washington, DC, USA, 11–15 May 2002; Volume 4, pp. 4032–4038.
21. Desaraju, V.; Michael, N. Leveraging Experience for Computationally Efficient Adaptive Nonlinear Model Predictive Control. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May 2017–3 June 2017; pp. 5314–5320.
22. Desaraju, V.; Spitzer, A.; O'Meadhra, C.; Lieu, L.; Michael, N. Leveraging experience for robust, adaptive nonlinear MPC on computationally constrained systems with time-varying state uncertainty. *Int. J. Robot. Res.* **2018**, *37*, 1690–1712. [CrossRef]
23. Vijayakumar, S.; Schaal, S. Locally weighted projection regression: An O (n) algorithm for incremental real time learning in high dimensional space. In Proceedings of the seventeenth international conference on machine learning (ICML 2000), Stanford, CA, USA, 29 June–2 July 2000; Volume 1, pp. 288–293.
24. Vijayakumar, S.; D'Souza, A.; Schaal, S. Incremental Online Learning in High Dimensions. *Neural Comput.* **2005**, *17*, 2602–2634. [CrossRef] [PubMed]
25. Vijayakumar, S.; D'Souza, A.; Schaal, S. Current XCSF Capabilities and Challenges. *Learn. Classif. Syst.* **2010**, *6471*.
26. Stalph, P.; Rubinsztajn, J.; Sigaud, O.; Butz, M. A comparative study: Function approximation with LWPR and XCSF. In Proceedings of the 12th Annual Conference Companion on Genetic and Evolutionary Computation, Portland, OR, USA, 7–11 July 2010; pp. 1–8.
27. Gijsberts, A.; Metta, G. Real-time model learning using incremental sparse spectrum gaussian process regression. *Neural Netw.* **2013**, *41*, 59–69. [CrossRef]
28. Thrun, S. Probabilistic robotics. *Commun. ACM* **2002**, *45*, 52–57. [CrossRef]
29. Li, Q.; Li, R.; Ji, K.; Dai, W. Kalman filter and its application. In Proceedings of the 2015 8th International Conference on Intelligent Networks and Intelligent Systems (ICINIS), Tianjin, China, 1–3 November 2015; pp. 74–77.
30. Shaiju, A.J.; Petersen, I.R. Formulas for Discrete Time LQR, LQG, LEQG and Minimax LQG Optimal Control Problems. In Proceedings of the 17th World Congress The International Federation of Automatic Control, Seoul, Republic of Korea, 6–11 July 2008; pp. 8773–8778.
31. Bemporad, A.; Morari, M.; Dua, V.; Pistikopoulos, E.N. The explicit linear quadratic regulator for constrained systems. *Automatica* **2002**, *38*, 3–20. [CrossRef]
32. Zhu, C.; Byrd, R.H.; Lu, P.; Nocedal, J. Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. *ACM Trans. Math. Softw. (TOMS)* **1997**, *23*, 550–560. [CrossRef]
33. Barthelemy, J.; Haftka, R. Function approximations. *Prog. Astronaut. Aeronaut.* **1993**, *150*, 51.
34. Wang, W.; Yan, J.; Wang, H.; Ge, H.; Zhu, Z.; Yang, G. Adaptive MPC trajectory tracking for AUV based on Laguerre function. *Ocean Eng.* **2022**, *261*, 111870. [CrossRef]
35. Dismuke, C.; Lindrooth, R. Ordinary least squares. *Methods Des. Outcomes Res.* **2006**, *93*, 93–104.
36. Islam, S.A.U.; Bernstein, D.S. Recursive least squares for real-time implementation [lecture notes]. *IEEE Control Syst. Mag.* **2019**, *39*, 82–85. [CrossRef]
37. Owen, J.P.; Ryu, W.S. The effects of linear and quadratic drag on falling spheres: An undergraduate laboratory. *Eur. J. Phys.* **2005**, *26*, 1085. [CrossRef]
38. Isaacs, R. *Differential Games: A Mathematical Theory with Applications to Warfare and Pursuit, Control and Optimization*; Dover publications: Mineola, NY, USA, 1999.
39. Zhang, R.; Gao, W.; Yang, S.; Wang, Y.; Lan, S.; Yang, X. Ocean Current-Aided Localization and Navigation for Underwater Gliders With Information Matching Algorithm. *IEEE Sens. J.* **2021**, *21*, 26104–26114. [CrossRef]
40. Yang, M.; Wang, Y.; Liang, Y.; Song, Y.; Yang, S. A Novel Method of Trajectory Optimization for Underwater Gliders Based on Dynamic Identification. *J. Mar. Sci. Eng.* **2022**, *10*, 307. [CrossRef]

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.