Journal of
Marine Science and Engineering

MDPI

Article

Enhancing Underwater Robot Manipulators with a Hybrid Sliding Mode Controller and Neural-Fuzzy Algorithm

Duc-Anh Pham and Seung-Hun Han *

Department of Mechanical System Engineering, Gyeongsang National University, Tongyeong 53064, Republic of Korea; pamanh97@gmail.com
* Correspondence: shhan@gnu.ac.kr; Tel.: +82-55-772-9105

Abstract: The sliding mode controller stands out for its exceptional stability, even when the system experiences noise or undergoes time-varying parameter changes. However, designing a sliding mode controller necessitates precise knowledge of the object's exact model, which is often unattainable in practical scenarios. Furthermore, if the sliding control law's amplitude becomes excessive, it can lead to undesirable chattering phenomena near the sliding surface. This article presents a new method that uses a special kind of computer program (Radial Basis Function Neural Network) to quickly calculate complex relationships in a robot's control system. This calculation is combined with a technique called Sliding Mode Control, and Fuzzy Logic is used to measure the size of the control action, all while making sure the system stays stable using Lyapunov stability theory. We tested this new method on a robot arm that can move in three different ways at the same time, showing that it can handle complex, multiple-input, multiple-output systems. In addition, applying LPV combined with Kalman helps reduce noise and the system operates more stably. The manipulator's response under this controller exhibits controlled overshoot (Rad), with a rise time of approximately 5 ± 3% seconds and a settling error of around 1%. These control results are rigorously validated through simulations conducted using MATLAB/Simulink software version 2022b. This research contributes to the advancement of control strategies for robotic manipulators, offering improved stability and adaptability in scenarios where precise system modeling is challenging.

Keywords: robot manipulator; neural network; fuzzy logic controller; MATLAB/Simulink; sliding mode control

Citation: Pham, D.-A.; Han, S.-H. Enhancing Underwater Robot Manipulators with a Hybrid Sliding Mode Controller and Neural-Fuzzy Algorithm. *J. Mar. Sci. Eng.* **2023**, *11*, 2312. https://doi.org/10.3390/jmse11122312

Academic Editor: Weicheng Cui

Received: 14 November 2023
Revised: 3 December 2023
Accepted: 5 December 2023
Published: 6 December 2023

1. Introduction

This scientific analysis focuses on addressing the challenges of precisely controlling underwater robotic arms amidst environmental fluctuations and limitations of traditional control algorithms. This study emphasizes the significance of unmanned underwater machines. These scholarly documents collectively explore remotely operated vehicles (ROVs) for inspection purposes, encompassing a comprehensive review of inspection-class ROVs, the development of an intelligent support system for ROV operators to enhance real-world performance, and the introduction of a novel control method utilizing terminal sliding mode and dynamic damping to improve fault tolerance and predictive capabilities for tracked ROVs [1–3], highlighting the roles of robotic arms. These studies also investigate the impact of machining trajectory on grinding force for complex-shaped stone by a robotic manipulator, apply adaptive neural-PID visual servoing tracking control using an Extreme Learning Machine (ELM), and model impedance control with limited interaction power for a 2R planar robot arm [4–6] or manipulators [7,8] in conducting complex underwater tasks beyond human reach. These manipulators, operated via controllers like joysticks, are designed for diverse missions, varying in size and strength, and are equipped with hydraulic or electric power, along with sensory or visual aids.

This paper presents a solution that integrates Sliding Mode Control (SMC) [9–14] and Radial Basis Function Neural Networks (RBFNN) to enhance the stability and accuracy of a three-degree-of-freedom robotic arm in underwater environments. SMC, known for its stability amidst noise and variable parameters, requires accurate model knowledge, which is complemented by RBFNN's ability to estimate nonlinear functions online. Additionally, fuzzy logic is proposed to mitigate oscillations around the sliding surface [15,16].

This research incorporates dynamic equations development for the ROV manipulator arm, designing sliding mode, fuzzy logic, and neural-network control systems, and includes noise calculations. It compares the efficacy of these proposed methods with traditional PID control through simulations. Results demonstrate superior performance of the SMC system in amplitude and settling time, with fuzzy logic control achieving the best tracking accuracy, and the neural-network control showing equivalent performance. This research demonstrates the application of neural network-based terminal sliding mode control to space robots actuated by control moment gyros, a novel approach to coordinating large-scale systems with a focus on interaction prediction principles, the utilization of a fuzzy sliding mode controller based on RBF neural network for the control of a three-link robot, the design of adaptive fuzzy sliding-mode control for high-performance islanded inverters in micro-grids, and a comprehensive examination of variable structure control applied to complex systems [17–21].

This paper also acknowledges the use of PID controllers for their simplicity and effectiveness, while highlighting the exploration of neural network and machine-learning approaches for more sophisticated nonlinear control strategies. This study contributes to the field by detailed modeling of robot dynamics, simplifying system-based controller development, and validating the controller on a real robot for balance and path following. This research enhances the understanding and capabilities in designing controllers for self-balancing robots, pushing the boundaries of traditional control methods.

This study addresses the challenges in controlling underwater robotic arms, stemming from the dynamic aquatic environment and the limitations of traditional control methods. It introduces a novel solution combining Sliding Mode Control (SMC) and Fuzzy Neural Network (FNN) to enhance flexibility, precision, and adaptability of the control system [22–24]. SMC effectively manages rapid environmental changes, while FNN optimizes the system based on real-time conditions, leading to a more accurate and stable robotic arm control [25] in unpredictable underwater settings.

Experiments conducted under simulated underwater conditions demonstrate that this integrated system significantly outperforms traditional control methods in accuracy and stability. Additionally, this study incorporates the Linear Parameter-Varying (LPV) and Kalman Filter into the system, further improving adaptability and precision. The inclusion of LPV and Kalman Filter makes the control system more versatile, capable of handling underwater variability, and elevates its reliability for high-precision tasks. This advanced approach not only enhances existing systems but also paves the way for developing more accurate and reliable underwater robotic technologies.

This research paper begins with an introduction highlighting the challenges in precisely controlling underwater robotic arms due to environmental fluctuations and limitations of traditional control algorithms. It emphasizes the importance of remotely operated vehicles (ROVs) and robotic manipulators for complex underwater tasks. The second section describes the dynamic modeling of a three-degree-of-freedom robotic arm, outlining its kinetic equations and Denavit–Hartenberg parameters. Next, the adaptive sliding control method is presented, integrating the Sliding Mode Control (SMC) and Radial Basis Function Neural Networks (RBFNN) to enhance system stability and accuracy. Fuzzy logic control is incorporated to mitigate oscillations. Meanwhile, the RBFNN estimates nonlinear functions within the control laws. The Results and Discussion section analyzes the simulation outcomes, comparing the efficacy of the proposed approach to conventional PID controllers in parameters such as rise time, settling error and noise elimination. Finally, the conclusion summarizes the vital contributions of this research in advancing underwater robotic control strategies amidst environmental unpredictability and model limitations.

Overall, this study aims to develop an adaptive, stable and high-precision control solution for underwater manipulators.

2. Robot Manipulator of Fancon 1263 Model

2.1. A Brief Overview of the Robotic Arm in This Research Paper

The ROV, or Remotely Operated Vehicle, is a purpose-built underwater apparatus equipped with propellers, cameras, and a manipulator arm. The ROV's body is constructed from a specialized aluminum alloy designed to be both lightweight and durable for marine environments. Its propellers provide omnidirectional mobility, enabling the ROV to move freely in all directions. The manipulator arms of the ROV are highly articulated, often exceeding 1 m in length, and can be fitted with various specialized tools such as welding, cutting, gripping, and holding devices, depending on the specific task requirements.

The simplistic structure of the ROV's robotic arm is illustrated in Figure 1, which serves as the model referred to, and extensively studied by, the author throughout this research paper.

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

Figure 1. The simple structured robotic manipulator of the ROV. (**a**) Top-down perspective; (**b**) View from the right.

This study analyzes a Remotely Operated Vehicle (ROV), an unmanned underwater vehicle, detailing its structure as shown in Figure 1. The ROV features a sturdy, watertight body, essential for underwater operation, housing internal electronics, and powered by thrusters for various underwater movements. It is also equipped with cameras and sensors for capturing and relaying underwater images and data.

A crucial element of the ROV is its manipulator arm, designed for precision tasks like grasping and welding, with a simple structure and three degrees of freedom, each link being about 1 m long. This arm, capable of rotational movements, is crucial for underwater maintenance and is remotely operated. It faces challenges like friction and communication delays, which this study addresses with advanced control methods. The arm's design, affixed to the mobile ROV, allows for a 3 m operational range.

This paper also examines the arm's dynamic model, a system of differential equations incorporating factors like gravity, buoyancy, and hydrodynamics, using the Lagrange–Euler formulation for modeling. The arm's kinematics are described using Denavit–Hartenberg parameters, linking joint torques to movements.

This dynamic model is vital for the control system's development, forming the mathematical basis for controller design, and ensuring optimal performance by accurately modeling system dynamics.

Given the relatively uncomplicated nature of ROV tasks, the authors of this study have utilized a three-degrees-of-freedom manipulator as the subject of investigation. The kinetic model for this three-degree-of-freedom manipulator system is outlined in reference [18]:

$$\mathrm{M}(\theta)\ddot{\theta} + \mathrm{C}\left(\theta, \dot{\theta}\right)\dot{\theta} + \mathrm{F}\left(\theta, \dot{\theta}\right) + \mathrm{G}(\theta) + \mathrm{d_v}(t) = \mathrm{u} \tag{1}$$

With $M = \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{bmatrix}$ is the inertial matrix, $F = \begin{bmatrix} F_1 & F_2 & F_3 \end{bmatrix}^T$ is the friction vector, $G = \begin{bmatrix} G_1 & G_2 & G_3 \end{bmatrix}^T$ is the gravity vector, $d_v(t)$ is the noise signal, $u = \begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix}^T$ is the control signal vector, $\theta = \begin{bmatrix} \theta_1 & \theta_2 & \theta_3 \end{bmatrix}^T$ is the angle of the joints of the manipulator system and $C = \begin{bmatrix} C_1 & C_2 & C_3 \end{bmatrix}^T$ is expresses the coriolis and centrifugal torques.

$$M_{11} = l_1^2\left(\frac{m_1}{3} + m_2 + m_3\right) + l_1 l_2 (m_2 + 2m_3)\cos(q_2) + l_2^2\left(\frac{m_2}{3} + m_3\right) \tag{2}$$

$$M_{12} = M_{21} = m_2\left(l_2^2 + l_1 l_2 \cos(\theta_2\ )\right) + m_3\left(l_2^2 + l_3^2 + l_1 l_2 \cos(\theta_2\ )\right) + m_3(l_1 l_3 \cos(\theta_2 + \theta_3\ ) + 2 l_2 l_3 \cos(\theta_3\ )) \tag{3}$$

$$M_{13} = M_{31} = m_3\left(l_3^2 + l_1 l_3 \cos(\theta_2 + \theta_3\ )\right) + m_3 l_2 l_3 \cos(\theta_3) \tag{4}$$

$$M_{22} = m_2 l_2^2 + m_3\left(l_2^2 + l_3^2 + 2 l_2 l_3 \cos(\theta_3,)\right) \tag{5}$$

$$M_{23} = M_{32} = m_3\left(l_3^2 + l_2 l_3 \cos(\theta_2 + \theta_3\ )\right) \tag{6}$$

$$M_{33} = m_3 l_3^2 \tag{7}$$

$$F_1 = -(m_2 + m_3) l_1 l_2 \left(2\dot{\theta}_1\dot{\theta}_2 + \dot{\theta}_2^2\right)\sin(\theta_2) - m_3 l_1 l_3 q \sin(\theta_2 + \theta_3) - m_3 l_2 l_3 \left(2\dot{\theta}_1\dot{\theta}_3 + 2\dot{\theta}_2\dot{\theta}_3 + \dot{\theta}_3^2\right)\sin(\theta_3) \tag{8}$$

$$F_2 = (m_2 + m_3)\dot{\theta}_1^2 \sin(\theta_2) + m_3 l_1 l_3 \dot{\theta}_1^2 \sin(\theta_2 + \theta_3) - m_3 l_2 l_3 \left(2\dot{\theta}_1\dot{\theta}_3 + 2\dot{\theta}_2\dot{\theta}_3 + \dot{\theta}_3^2\right)\sin(\theta_3) \tag{9}$$

$$F_3 = m_3 l_1 l_3 \dot{\theta}_1^2 \sin(\theta_2 + \theta_3) + m_2 l_2 l_3 \left(\dot{\theta}_1 + \dot{\theta}_2\right)\sin(\theta_3) \tag{10}$$

$$q = 2\dot{\theta}_1\dot{\theta}_2 + 2\dot{\theta}_1\dot{\theta}_3 + 2\dot{\theta}_2\dot{\theta}_3 + \dot{\theta}_2^2 + \dot{\theta}_3^2 \tag{11}$$

$$G_1 = m_1 g l_1 \cos(\theta_1) + m_2 g(l_2 \cos(\theta_1 + \theta_2) + l_1 \cos(\theta_1)) + m_3 g(l_2 \cos(\theta_1 + \theta_2) + l_1 \cos(\theta_1)) + m_3 g(l_3 \cos(\theta_1 + \theta_2 + \theta_3)) \tag{12}$$

$$G_2 = m_2 g l_2 \cos(\theta_1 + \theta_2) + m_3 g(l_2 \cos(\theta_1 + \theta_2)) + m_3 g(l_3 \cos(\theta_1 + \theta_2 + \theta_3)) \tag{13}$$

$$G_3 = m_3 g(l_3 \cos(\theta_1 + \theta_2 + \theta_3)) \tag{14}$$

$$C_1 = \sum_{j=1}^{3}\sum_{k=1}^{3} \Gamma_{1jk}(\theta)\dot{\theta}_j\dot{\theta}_k \tag{15}$$

$$C_2 = \sum_{j=1}^{3}\sum_{k=1}^{3} \Gamma_{2jk}(\theta)\dot{\theta}_j\dot{\theta}_k \tag{16}$$

$$C_3 = \sum_{j=1}^{3}\sum_{k=1}^{3} \Gamma_{3jk}(\theta)\dot{\theta}_j\dot{\theta}_k \tag{17}$$

$$\Gamma_{ijk}(\theta) = \frac{1}{2}\left(\frac{\partial M_{ij}}{\partial \theta_k} + \frac{\partial M_{ik}}{\partial \theta_j} - \frac{\partial M_{jk}}{\partial \theta_i}\right) \tag{18}$$

The parameters of the manipulator considered in this study are:
m_1 = 1 kg is the mass of joint 1,
m_2 = 0.8 kg is the mass of joint 2,
m_3 = 0.5 kg is the mass of joint 3,

l_1 = 1 m is the length of joint 1,
l_2 = 0.8 m is the length of joint 2,
l_3 = 0.6 m is the length of joint 3,
g = 9.81 m/s^2 is the acceleration due to gravity.
$\Gamma_{ijk}(\theta)$ are Christoffel symbols.
M_{ij}, M_{ik}, M_{jk} are the elements of the inertia matrix.

2.2. Forward Kinematics

It has been given that there exists a matrix $\mathrm{T}_{\mathrm{i}}^{\mathrm{i}-1}$ that is called a homogeneous transformation matrix, and it has the form:

$$\mathrm{T}_{\mathrm{i}}^{\mathrm{i}-1} = \begin{bmatrix} \mathrm{R_i} & \mathrm{p_i} \\ 0 & 1 \end{bmatrix} \tag{19}$$

where: R_i (3 × 3 matrix): rotation matrix.
p_i (3 × 1 matrix): translation vector.

$$\mathrm{T}_{\mathrm{n}}^{0} = \prod_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{T}_{\mathrm{i}}^{\mathrm{i}-1} = \begin{bmatrix} \mathrm{n_x} & \mathrm{o_x} & \mathrm{a_x} & \mathrm{p_x} \\ \mathrm{n_y} & \mathrm{o_y} & \mathrm{a_y} & \mathrm{p_y} \\ \mathrm{n_z} & \mathrm{o_z} & \mathrm{a_z} & \mathrm{p_z} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{20}$$

where:
n: the number of robot's joints.
$\mathrm{T}_{\mathrm{i}}^{\mathrm{i}-1}$: the transformation matrix from coordinate system i to coordinate system i − 1.
$\mathrm{T}_{\mathrm{n}}^{0} = \mathrm{f}(q_1, q_2, q_{3,\ldots,} q_n)$; $q_1 \div q_n$ are joint variables; n, s, and a are vectors indicating directions, $\mathrm{p} = \begin{bmatrix} \mathrm{p_x}, \mathrm{p_y}, \mathrm{p_z} \end{bmatrix}^{\mathrm{T}}$ is position vector.

Hence, by having knowledge of the geometric characteristics of the links and the kinematic rules governing the joints, it is possible to fully ascertain the position and orientation of the manipulator links.

The objective of the forward kinematics problem is to determine the position and orientation of the end-effector when the values of the joint variables are known. Typically, these joint values are expressed as a function of time, denoted as $\mathrm{q_i} = \mathrm{q_i}(\mathrm{t})$, and our task is to ascertain the position and orientation of the end-effector corresponding to these specific joint values. To address the forward kinematics problem, it is necessary to establish the kinematic equations of the robot, as outlined in Equation (20).

For reference, we established a coordinate system with its origin at $\mathrm{O_0}$, denoted as $\mathrm{x_0}$, $\mathrm{y_0}$, and $\mathrm{z_0}$, as depicted in Figure 2. Another coordinate system, $\mathrm{O_1}$, $\mathrm{x_1}$, $\mathrm{y_1}$, and $\mathrm{z_1}$, is positioned at joint 2, where the $\mathrm{z_1}$-axis aligns with the second joint axis. Subsequently, coordinate system $\mathrm{O_2}$, $\mathrm{x_2}$, $\mathrm{y_2}$, and $\mathrm{z_2}$, has its origin at joint 3, with the $\mathrm{z_2}$-axis aligned with the third joint axis and the $\mathrm{x_2}$-axis aligned with the gripper direction, pointing toward the jaws. Lastly, coordinate system $\mathrm{O_3}$, $\mathrm{x_3}$, $\mathrm{y_3}$, and $\mathrm{z_3}$, has its origin situated at the center of the gripper. The Denavit–Hartenberg (DH) parameter for a robot in angular coordinate system in Table 1.

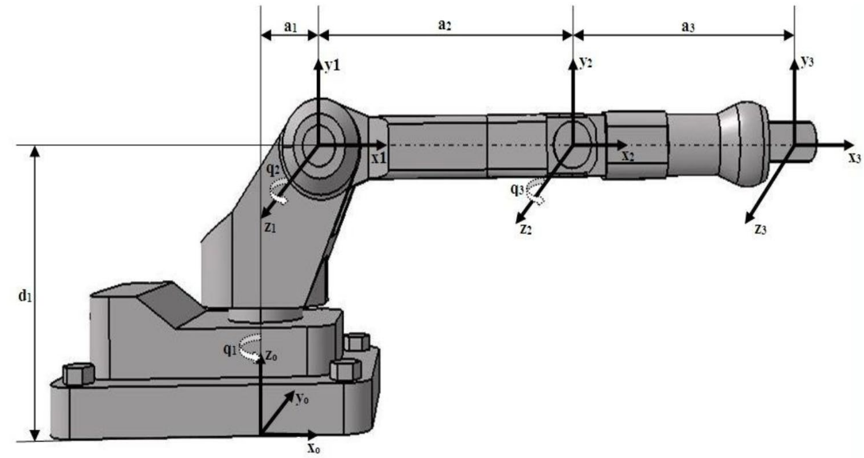

Figure 2. Robot kinematic diagram in angular coordinate system.

Table 1. The Denavit–Hartenberg (DH) parameter table for a robot in angular coordinate system.

Joint	α_i	a_i	d_i	θ_i
1	90°	a_1	d_1	q_1
2	0	a_2	0	q_2
3	0	a_3	0	q_3

From the DH parameter table, it can be determining the transformation matrices between coordinate systems:

$$T_1^0 = \begin{bmatrix} cq_1 & 0 & sq_1 & a_1 \cdot cq_1 \\ sq_1 & 0 & -cq_1 & a_1 \cdot sq_1 \\ 0 & 1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (21)$$

$$T_2^1 = \begin{bmatrix} cq_2 & -sq_2 & 0 & a_2 \cdot cq_2 \\ sq_2 & cq_2 & 0 & a_2 \cdot sq_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (22)$$

$$T_3^2 = \begin{bmatrix} cq_3 & -sq_3 & 0 & a_3 \cdot cq_3 \\ sq_3 & cq_3 & 0 & a_3 \cdot sq_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (23)$$

where, abbreviation symbols are used as follows: $cq_1 = \cos(q_1)$, $sq_1 = \sin(q_1)\cdot$

The homogeneous transformation matrix representing the position and orientation of the end effector with respect to the origin coordinate system can be determined.

$$T_3^0 = T_1^0 \cdot T_2^1 \cdot T_3^2 = \begin{bmatrix} cq_1 \cdot c(q_2+q_3) & -cq_1 \cdot s(q_2+q_3) & sq_1 & cq_1 \cdot [a_3 \cdot s(q_2+q_3) + a_2 \cdot sq_2 + a_1] \\ sq_1 c(q_2+q_3) & -sq_1 \cdot s(q_2+q_3) & -cq_1 & sq_1 \cdot [a_3 \cdot s(q_2+q_3) + a_2 \cdot sq_2 + a_1] \\ s(q_2+q_3) & c(q_2+q_3) & 0 & a_3 \cdot s(q_2+q_3) + a_2 \cdot sq_2 + a_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (24)$$

The dynamic equations of the robot in the angular coordinate system can be acquired by matching corresponding elements of two matrices (T_n^0 and T_3^0).

$$\begin{cases} n_x = \cos(q_1)\cdot\cos(q_2+q_3) \\ n_y = \sin(q_1)\cdot\cos(q_2+q_3) \\ n_z = \sin(q_2+q_3) \\ o_x = -\cos(q_1)\cdot\sin(q_2+q_3) \\ o_y = -\sin(q_1)\cdot\sin(q_2+q_3) \\ o_z = \cos(q_2+q_3) \\ a_x = \sin(q_1) \\ a_y = -\cos(q_1) \\ a_z = 0 \\ p_x = \cos(q_1)\cdot[a_3\cdot\cos(q_2+q_3)+a_2\cdot\cos(q_2)+a_1] \\ p_y = \sin(q_1)\cdot[a_3\cdot\cos(q_2+q_3)+a_2\cdot\cos(q_2)+a_1] \\ p_z = a_3\cdot\sin(q_2+q_3)+a_2\cdot\sin(q_2)+d_1 \end{cases} \tag{25}$$

2.3. Inverse Kinematics

The inverse kinematics problem necessitates determining the robot's joint values based on the known position and orientation of the end effector point. These parameters are deduced from a given motion trajectory of the robot, commonly represented in matrix form.

$$T_0^n = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{26}$$

The solution to the inverse kinematics problem is of paramount significance in robot control. In practical applications, there exists prior knowledge regarding the desired position and orientation of the target end effector point to be reached. The objective is to compute the required joint values for meeting the motion criteria. Consequently, the resolution of the inverse kinematics problem holds pivotal importance in ensuring efficient control of the robot's operations.

From Equations (24) and (26), there are:

$$\begin{aligned} T_3^0 = T_1^0\cdot T_2^1\cdot T_3^2 &= \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} cq_1\cdot c(q_2+q_3) & -cq_1\cdot s(q_2+q_3) & sq_1 & cq_1\cdot[a_3\cdot s(q_2+q_3)+a_2\cdot sq_2+a_1] \\ sq_1 c(q_2+q_3) & -sq_1\cdot s(q_2+q_3) & -cq_1 & sq_1\cdot[a_3\cdot s(q_2+q_3)+a_2\cdot sq_2+a_1] \\ s(q_2+q_3) & c(q_2+q_3) & 0 & a_3\cdot s(q_2+q_3)+a_2\cdot sq_2+a_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{27}$$

Deduce:

$$\left[T_1^0\right]^{-1}\cdot T_3^0 = T_2^1\cdot T_3^2 = T_3^1 \tag{28}$$

A homogeneous matrix is a matrix of the form:

$$T = \begin{pmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{29}$$

According to [21], the inverse matrix of matrix T denoted as T^{-1} will be:

$$\mathrm{T}^{-1} = \begin{pmatrix} n_x & n_y & n_z & -p\cdot n \\ o_x & o_y & o_z & -p\cdot o \\ a_x & a_y & a_z & -p\cdot a \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{30}$$

Substituting into the expression:

$$\mathrm{T}_2^1\cdot\mathrm{T}_3^2 = \begin{pmatrix} cos(q_2+q_3) & -sin(q_2+q_3) & 0 & a_3\cdot cos(q_2+q_3)+a_2\cdot cos(q_2) \\ sin(q_2+q_3) & cos(q_2+q_3) & 0 & a_3\cdot sin(q_2+q_3)+a_2\cdot sin(q_2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{31}$$

Balancing the corresponding element at row 3, column 4 of the two matrices, obtain:

$$sin(q_1)\cdot p_x - cos(q_1)\cdot p_y = 0 \tag{32}$$

Deduce:

$$q_1 = atan2\left(p_y, p_x\right) \tag{33}$$

Through the position of the Robot's end-effector point, also determine the values of the joint variables:

$$\begin{cases} a_3\cdot cos(q_2+q_3)+a_2\cdot cos(q_2) = cos(q_1)\cdot p_x + sin(q_1)\cdot p_y - a_1 \\ a_3\cdot sin(q_2+q_3)+a_2\cdot sin(q_2) = p_z - d_1 \end{cases} \tag{34}$$

Deduce:

$$\begin{cases} q_2 = atan2\left(\frac{p_z-d_1}{cos(q_1)\cdot p_x+sin(q_1)\cdot p_y-a_1}\right) - atan\left(\frac{a_3\cdot sin(q_3)}{a_3\cdot cos(q_3)+a_2}\right) \\ q_3 = \mp acos\frac{\left[cos(q_1)\cdot p_x+sin(q_1)\cdot p_y-a_1\right]^2+(p_z-d_1)^2-\left(a_3{}^2+a_2{}^2\right)}{2\cdot a_2\cdot a_3} \end{cases} \tag{35}$$

2.4. Noise When Working Underwater

The resistance forces on a subsea manipulator include:

Frictional resistance: this is the resistance caused by the friction between the surface of the manipulator and the water. This force depends on the manipulator's speed, the surface area in contact, and the viscosity of the water. The formula for calculating frictional resistance:

$$\mathrm{F}_{\text{frictional}} = k\cdot v \tag{36}$$

where:

F is the frictional resistance.

k is the coefficient of friction.

v is the speed of the manipulator.

Pressure resistance: This is the resistance caused by the water pressure acting on the surface of the manipulator. This force depends on the depth below the sea and the speed of the manipulator. The formula for calculating pressure resistance:

$$\mathrm{F}_{\text{pressure}} = \frac{1}{2}\cdot\rho\cdot\mathrm{A}\cdot v^2 \tag{37}$$

where:

F is the pressure resistance.

ρ is the density of the water.

A is the surface area of the manipulator.

v is the speed of the manipulator.

Molecular resistance: this is the resistance caused by water molecules hindering the movement of the manipulator. This force depends on the viscosity of the water and the speed of the manipulator. The formula for calculating molecular resistance:

$$\mathrm{F}_{\text{molecular}} = 6 \cdot \pi \cdot \eta \cdot r \cdot v \tag{38}$$

where:

F is the molecular resistance.

η is the viscosity of the water.

r is the radius of the manipulator.

v is the speed of the manipulator.

Impact force is when the ROV changes motion suddenly.

In the process of researching and developing manipulators attached to ROVs, identifying and analyzing sudden forces arising from changes in the ROV's motion in the system plays an important role. These forces, including inertial forces, centrifugal forces, and Coriolis forces, have a significant influence on the precision and stability of the manipulator. Static force occurs when the ROV accelerates or decelerates suddenly. It is calculated using formula:

$$F_{inertia} = m \cdot a \tag{39}$$

where: m is the mass of manipulator.

a is the acceleration of the ROV.

This force causes the position of the manipulator to shift. This change can affect the ability to perform tasks accurately, essential in many underwater activities. When the ROV makes rapid changes of direction, the centrifugal force becomes significant. The formula for this force is:

$$F_{centrifugal} = \frac{m \cdot v^2}{r} \tag{40}$$

where v is the velocity of the ROV and r is the turning radius.

Centrifugal force can cause rotation of the manipulator, affecting its angular position and ability to maintain correct orientation. In the rotating reference frame of the ROV, the Coriolis force becomes important, especially when the ROV changes its angular velocity. The formula for calculating this force is:

$$F_{coriolis} = 2m(v_x \times \Omega) \tag{41}$$

where v_x is the velocity of the manipulator handle, and

Ω is the angular velocity of the ROV.

This force causes a deflection in the manipulator's trajectory, requiring adjustment to achieve the correct position. This analysis provides a detailed view of how sudden forces affect the manipulator in the system, highlighting the complexity of underwater robot operations and the need for advanced control strategies to manage these challenges. An in-depth understanding of these effects is necessary to develop control systems capable of eliminating their effects, ensuring that the manipulator operates accurately and stably in full underwater environments.

The total resistive force on the manipulator is calculated by adding up all the resistive forces, namely:

$$\mathrm{F}_{\text{total}} = \mathrm{F}_{\text{frictional}} + \mathrm{F}_{\text{pressure}} + \mathrm{F}_{\text{molecular}} + F_{inertia} + F_{centrifugal} + F_{coriolis} \tag{42}$$

3. Adaptive Sliding Control Using Neural Network and Logic Fuzzy Controller

3.1. Sliding Mode Control

Sliding mode control (SMC) represents a control system approach employed for stabilizing dynamic systems, particularly those characterized by uncertain or nonlinear dynamics. The fundamental concept underlying SMC is to compel the system to adhere to a specific trajectory or sliding surface within the state space, irrespective of initial system conditions or external disturbances.

Sliding mode control is introduced as a powerful control method capable of achieving stable control of nonlinear systems even in the presence of model uncertainty and external disturbances. The core idea of sliding mode control is to force the system's states to "slide" along a switching hyperplane (referred to as the sliding surface) toward an equilibrium point. This is achieved by applying a discontinuous control law to switch control actions based on the sign of the error between the current state and the desired state. The switching ensures that the system's states converge and remain on the sliding surface, providing robustness against model uncertainties and disturbances. To implement sliding mode control, an appropriate sliding surface is first designed based on the system's dynamics and relative degree. Then, a switching control law is derived, comprising equivalent control terms to compensate for model uncertainties and switching terms to counteract disturbances and bring states onto the sliding surface. In this research paper, the sliding mode control method is applied to control a three-degree-of-freedom robotic arm for underwater operations. Challenges related to model uncertainty and noise in the underwater environment drive the use of sliding mode control. Let θ_d denote the desired manipulator response, while θ represents the actual output of the manipulator. The primary control objective entails minimizing the error between θ and θ_d, as defined in Equation (43).

$$e = \theta - \theta_d \tag{43}$$

Sliding surface s_I (with $i = \overline{1,3}$):

$$s_i = \dot{e}_i + a_i e_i \tag{44}$$

where:

$$a_i > 0$$

Choose the control law u such that derivative of the sliding surface:

$$\dot{e} = \begin{bmatrix} \dot{s}_1 \\ \dot{s}_2 \\ \dot{s}_3 \end{bmatrix} = -\begin{bmatrix} k_1 \cdot \text{sign}(s_1) \\ k_2 \cdot \text{sign}(s_2) \\ k_3 \cdot \text{sign}(s_3) \end{bmatrix} \tag{45}$$

The control goal is to design u such that $\dot{s}$ is driven to zero exponentially fast, ensuring the system states converge to the sliding surface.

Sliding control law:

$$u_s = \left[F\left(\theta, \dot{\theta}\right) + G(\theta)\right] + M \begin{bmatrix} \ddot{\theta}_{1d} - a_1\left(\theta_1 - \dot{\theta}_{1d}\right) - k_1 \cdot \text{sign}(s_1) \\ \ddot{\theta}_{2d} - a_2\left(\theta_2 - \dot{\theta}_{2d}\right) - k_2 \cdot \text{sign}(s_2) \\ \ddot{\theta}_{3d} - a_3\left(\theta_3 - \dot{\theta}_{3d}\right) - k_3 \cdot \text{sign}(s_3) \end{bmatrix} \tag{46}$$

3.2. Sliding Mode Control-Fuzzy Logic Control

Fuzzy logic is applied to estimate the amplitude of the sliding mode control law in a dynamic, adaptive manner. This helps reduce chattering in the control output. A fuzzy inference system (FIS) is developed using input of error (s) and change in error. Gaussian membership functions are defined for the fuzzy sets of the input and output variables. IF-THEN rules are constructed to relate the fuzzy sets of error to the fuzzy sets of the control gain (k). The FIS evaluates the rules using fuzzy logic operations (in this paper it is AND) to map crisp error values to crisp control gain values. This adaptive, fuzzy-estimated control gain is then used within the overall sliding mode control law to regulate the system states. The use of fuzzy logic is rationalized as it allows modeling nonlinear dynamics and uncertainties in a simple, intuitive way using linguistic rules. This helps sliding mode control accommodate unknown factors in the underwater manipulator system. By integrating fuzzy logic within sliding mode control, the approach aims to leverage the robustness of SMC while reducing chattering through adaptive gain estimation.

Consider the sliding mode control system-Fuzzy Logic control [26–29]. The amplitude of the control rule is dynamically estimated through a fuzzy inference system utilizing the rules structured as follows:

IF s_I is A_i^m THEN k_I is B_i^m

Within $i = \overline{1,3}$

Where A_i^m and B_i^m are fuzzy sets (Figure 3).

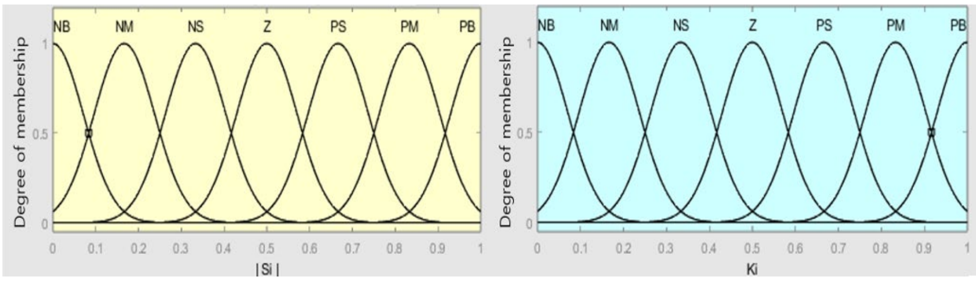

Figure 3. Fuzzy set of s_I and k_i.

The membership functions used are Gaussian functions [18].

$$\mu_A(s_i) = \exp\left(-\left(\frac{s_i - \alpha_i}{\sigma_i}\right)^2\right) \tag{47}$$

where:

$i = \overline{1,3}$, $k_I \geq 0$

α_i, σ_I are the center and width of the ith Gaussian function, respectively.

Choose a positive definite function.

$$V = \frac{1}{2}\sum_{i=1}^{3} s_i^2 \tag{48}$$

From (47) deduce:

$$\dot{V} = -\sum_{i=1}^{3} s_i \cdot k_i \cdot \mathrm{sign}(s_i) \tag{49}$$

$$\dot{V} = -\sum_{i=1}^{3} |s_i| \cdot k_i \tag{50}$$

Therefore, the fuzzy rule is defined as follows:

IF $|s_I|$ is NB THEN k_I is NB

IF $|s_I|$ is NM THEN k_I is NM

IF $|s_I|$ is NS THEN k_I is NS

IF $|s_I|$ is Z THEN k_I is Z

IF $|s_I|$ is PS THEN k_I is PS

IF $|s_I|$ is PM THEN k_I is PM

IF $|s_I|$ is PB THEN k_I is PB

Where:

NB: Negative Big

NM: Negative Medium

NS: Negative Small

Z: Zero

PS: Positive Small
PM: Positive Medium
PB: Positive Big

Program for Fuzzy Logic Control in MATLAB version 2022b that is presented in Appendix A section.

3.3. Sliding Control Using RBFNN Network

An RBFNN is used to estimate nonlinear functions within the sliding mode control law, since an exact mathematical model of the underwater manipulator system is difficult to obtain. The RBFNN consists of three parts—an input layer, a hidden layer with radial basis activation functions, and an output layer. It is trained online using a gradient descent-based algorithm to minimize the error between the actual and desired sliding surfaces. The RBFNN learns and adapts the control law based on current state feedback without requiring an exact model of the system dynamics. This helps make the sliding mode control more robust by compensating for model uncertainties and nonlinearities in the real-world underwater system. The RBFNN also allows the control law to be adjusted continuously in real-time as the system operates, improving tracking performance.

In this research, the strategic selection of the Radial Basis Function Neural Network (RBFNN) marks a significant development in the control of underwater robotic arms. RBFNN is highly valued for its efficient handling of unstable and fluctuating data, a critical characteristic in the challenging underwater environment. This choice presents a clear advantage over other neural-network architectures like Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM). While CNNs excel in image processing due to their feature detection capabilities, they may not fully capture the continuous variations of the underwater environment. Similarly, while LSTMs are effective in processing sequential data and long-term information, they might struggle with the rapid and abrupt changes typical in underwater settings. In contrast, RBFNN, with its rapid adaptability and effective handling of unstructured data, emerges as the optimal choice in this scenario. This decision not only reflects a deep understanding of the specific challenges in underwater robotic control but also demonstrates flexibility and creativity in applying AI technology to address complex issues.

In summary, RBFNN is an effective choice in solving prediction, classification, and control problems, especially in real-world environments with noise and uncertainty. In this study, we consider the application of the sliding control system using RBF neural network [29–31]. The nonlinear system is given as (51).

$$\begin{cases} \dot{x} = f(x) + g(x)u \\ y = h(x) \end{cases} \tag{51}$$

where x = $[x_1, x_2, \ldots, x_n]^T$ is the state vector, u is the input signal, y is the output signal, f(x) and g(x) are the nonlinear functions describing the characteristic kinematics of the system.

Suppose (52) does not contain u and (46) contains u.

$$\frac{d^{r-1}y}{dt^{r-1}} = L_f^{(r-1)}h(x) + L_f^{(r-2)}L_g h(x) \tag{52}$$

$$\frac{d^r y}{dt^r} = L_f^{(r)}h(x) + L_f^{(r-1)}L_g h(x)u \tag{53}$$

where:

$L_f h(x) = \frac{\partial h(x)}{\partial x} f(x)$ is the Lie derivative in the f(x) direction.

$L_g h(x) = \frac{\partial h(x)}{\partial x} g(x)$ is the Lie derivative in the g(x) direction.

From (53) deduce:

$$y^{(r)} = a(x) + b(x)u \tag{54}$$

where:

$$a(x) = L_f^{(r)} h(x) \tag{55}$$

$$b(x) = L_f^{(r-1)} L_g h(x) \tag{56}$$

The control law u is determined such that:

$$y^{(r)} = v(t) \tag{57}$$

where $v(t)$ is the new control signal.

From (55)–(57) deduce:

$$u^*(x) = \frac{1}{b(x)}[-a(x) + v(t)] \tag{58}$$

The control law $v(t)$ is determined based on the pole assignment method with the characteristic equation:

$$e^{(r)} + k_1 e^{(r-1)} + \ldots + k_{r-1} e^1 = 0 \tag{59}$$

where:

$y_m(t)$ is the desired output value.

$y(t)$: actual output value

$k_1, k_2, \ldots, k_{r-1}$: is chosen so that (59) has a negative definite solution.

$$e(t) = y_m(t) - y(t) \tag{60}$$

From (57), (59) and (60) deduce:

$$y_m^{(r)}(t) - v(t) + k_1 e^{(r-1)} + \cdots + k_{(r-1)} e = 0 \tag{61}$$

Or:

$$y_m^{(r)}(t) + k_1 e^{(r-1)} + \cdots + k_{(r-1)} e = v(t) \tag{62}$$

Set [28]:

$$\bar{e}_s + \eta e_s(t) = k_1 e^{(r-1)} + \cdots + k_{(r-1)} e \tag{63}$$

where:

$$e_s(t) = k_1 e^{(r-1)} + k_2 e^{(r-2)} + \cdots + k_{(r-1)} e$$

η: is a positive constant.

$$\bar{e}_s = \dot{e}_s - e^{(r)}$$

The control law (58) is approximated by the RBFNN network:

$$\bar{u}(x) = \varnothing_u^T \Psi_u(x) \tag{64}$$

where:

$\Psi_u(x)$: is a Gaussian basis function.

$\varnothing_u$: is the weight that updates such that $\bar{u}$ approaches u^*

Because $\bar{u}$ is estimated by RBFNN network with a finite number of neurons in the hidden layer, so errors are unavoidable. Let $\delta_u(x)$ be the structural error:

$$u^*(x) = \varnothing_u^{*T} \Psi_u(x) + \delta_u(x) \tag{65}$$

Difference between $\hat{u}$ and u^*

$$\hat{u}(x) - u^*(x) = \widetilde{\varnothing}_u^T \Psi_u(x) - \delta_u(x) \tag{66}$$

where $\widetilde{\varnothing}_u^T = \varnothing_u - \varnothing_u^*$ estimated parameter error. Due to the structural error, the control law has the following form.

$$u = \hat{u} + u_s \tag{67}$$

where u_s is the sliding control law chosen so that the closed system is stable.

From (54) deduce:

$$y^{(r)} = a(x) + b(x)u(t) = a(x) + b(x)u^*(t) + b(x)[u(t) - u^*(t)] \tag{68}$$

Compare (58) and (68), there are:

$$y^{(r)} = v(t) + b(x)[u(t) - u^*(t)] \tag{69}$$

Output error is set:

$$e^{(r)} = y_m^{(r)} - y^{(r)} \tag{70}$$

Replace (69) in (70)

$$e^{(r)} = -\bar{e}_s - \eta e_s - b\widetilde{\varnothing}_u^T \Psi_u(x) + b\delta_u - bu_s \tag{71}$$

From (71) there are:

$$\dot{e}_s + \eta e_s = -b\widetilde{\varnothing}_u^T \Psi_u(x) + b\delta_u - bu_s \tag{72}$$

Choose a positive definite function:

$$V = \frac{1}{2b}e_s^2 + \frac{1}{2}\widetilde{\varnothing}_u^T Q_u \dot{\varnothing}_u \tag{73}$$

Deduce:

$$\dot{V} = \frac{1}{b}e_s\dot{e}_s - \frac{\dot{b}}{2b^2}e_s^2 + \widetilde{\varnothing}_u^T Q_u \dot{\varnothing}_u \tag{74}$$

where $\widetilde{\varnothing}_u = \dot{\varnothing}_u$, Q_u is a positive definite matrix.

Replace (73) into (75):

$$\dot{V} = -\frac{\eta e_s^2}{b} - e_s u_s + e_s\delta_u + \widetilde{\varnothing}_u^T\left(Q_u\dot{\varnothing}_u - \Psi_u e_s\right) - \frac{\dot{b}}{2b^2}e_s^2 \tag{75}$$

Based on [32], (70) adaptive parameter update rule

$$\dot{\varnothing}_u = Q_u^{-1}\Psi_u e_s \tag{76}$$

Replace (72) in (73), there are:

$$\dot{V} = -\frac{\eta e_s^2}{b} - e_s u_s + e_s\left(\delta_u - \frac{\dot{b}}{2b^2}e_s\right) \le -\frac{\eta e_s^2}{b} - e_s u_s + |e_s|\left(\overline{\delta}_u + \frac{D_b}{2b^2}|e_s|\right) \tag{77}$$

where $\dot{b}(x) \le D_b$, D_b is a continuous function [33].

If $u_s = \left(\overline{\delta}_u + \frac{D_b}{2b^2}|e_s|\right)\mathrm{sign}(e_s)$ is chosen, (73) becomes $\dot{V} = -\frac{\eta e_s^2}{b} \le 0$, so the closed system is stable according to Lyapunov's criterion.

The RBFNN network (Figure 4) plays the role of estimating nonlinear functions in the control law so that the closed system is stable. The input to the network is $\left[\theta_i, \dot{\theta}_i\right]^T$, the output is the adaptive control law. It updated weights online based on (77).

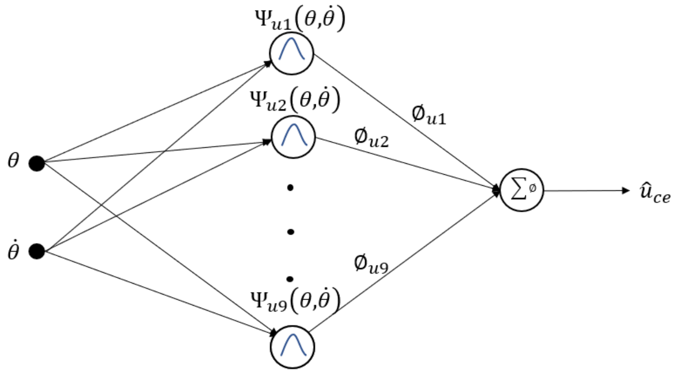

Figure 4. RBFNN Network.

$\Psi_i = [\Psi_{1i}\ \Psi_{2i}\ \ldots\ \Psi_{9i}]^T$ is Gauss base function, $\varnothing_i = [\varnothing_{1i}\ \varnothing_{2i}\ \ldots\ \varnothing_{9i}]^T$ is the update parameter. The jth Gaussian basis function is defined as follows:

$$\Psi_{ji} = \exp\left(-\frac{(\theta_i - c_{1j})^2 + \left(\dot{\theta}_i - c_{2j}\right)^2}{\delta_j^2}\right) \tag{78}$$

where

$$j = \overline{1,9}$$

The widths of the Gaussian basis functions are chosen equally: $\sigma_j = 0.9$

c_{kj} is the base function center $k = \overline{1,2}$

3.4. Adaptive Sliding Controller Using Fuzzy Neural Model

The adaptive sliding controller utilizing the fuzzy neural model represents a fusion of the fuzzy sliding controller and the sliding controller employing the RBFNN network [34–38]. Fuzzy logic dynamically assesses the controller described in Equation (66) to mitigate oscillations. Meanwhile, the RBFNN network is tasked with identifying the control law by estimating the nonlinear functions within the control rule, thus ensuring stability in the closed system. A visual representation of the adaptive sliding controller using the fuzzy neural model is illustrated in Figure 5.

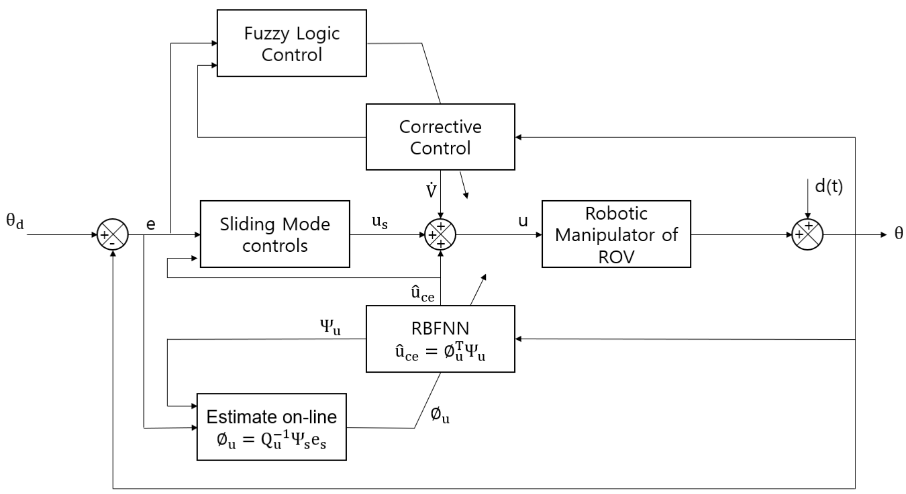

Figure 5. Schematic of adaptive sliding mode control using the fuzzy neural model.

The control formula of the controller:

$$u = u_s + \dot{V} + \overline{u}_{ce} + d(t) \tag{79}$$

4. Results and Discussion

4.1. Simulation and Simulation Results

When d(t) = 0.01 × sin(0.06 πt) noise or random noise (with zero mean, 0.01 variance, initial value is random) affects the manipulator and object parameters change as follows: the weight of joint 1 increased by 50% at 10 s, joint 2 increased by 10% at 20 s and joint 3 increased by 20% at 40 s. The results show that the response of the manipulator system has no oscillations and overshoots, the steady-state error approaches zero (see Table 2). The concordance between the response of the manipulator and the reference signal was 96% (Figure 6). The adaptive sliding control law using fuzzy neural model is presented in Figure 7. This result is obtained due to the online identification of the control rule based on the estimation of nonlinear functions using the RBFNN network (Figure 8) and the online estimation of the control rule using fuzzy logic (Figure 9). The phase trajectory has no oscillation around the slip surface (Figure 10).

Table 2. Quality parameters of the manipulator response.

	Joint 1	Joint 2	Joint 3
Overshoot	$\pm\frac{\pi}{2} \pm 3\%$ (Rad)	$\pm\frac{\pi}{2} \pm 4\%$ (Rad)	$\pm\frac{\pi}{2} \pm 5\%$ (Rad)
Risetime	$5 \pm 3\%$ (s)	$5 \pm 3\%$ (s)	$5 \pm 3\%$ (s)
Ess	0.68%	1.24%	1.32%

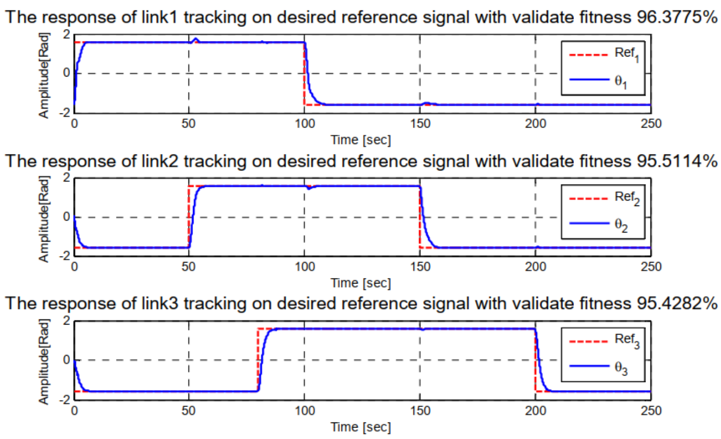

Figure 6. Response of the manipulator.

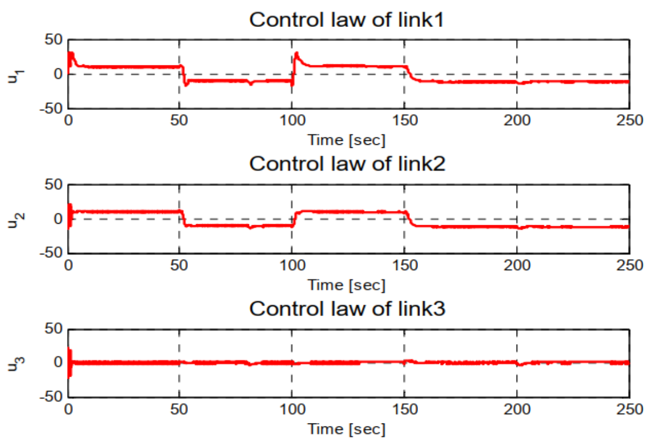

Figure 7. Control law $u = u_s + \dot{u} + \hat{u}_{ce} + d(t) \cdot$

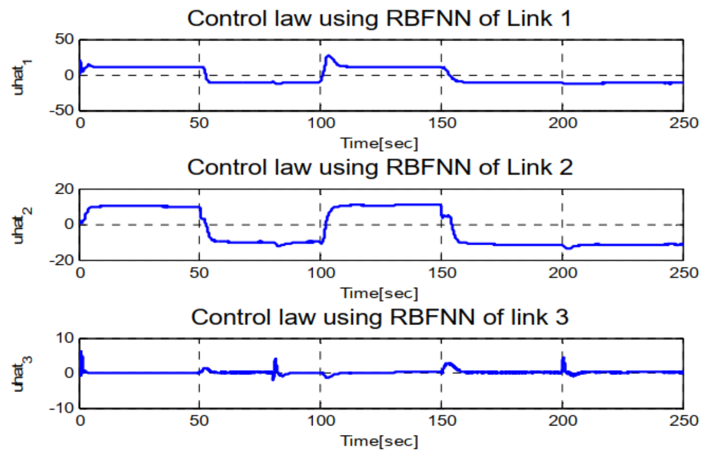

Figure 8. The adaptive control rule is recognized by the RBFNN network.

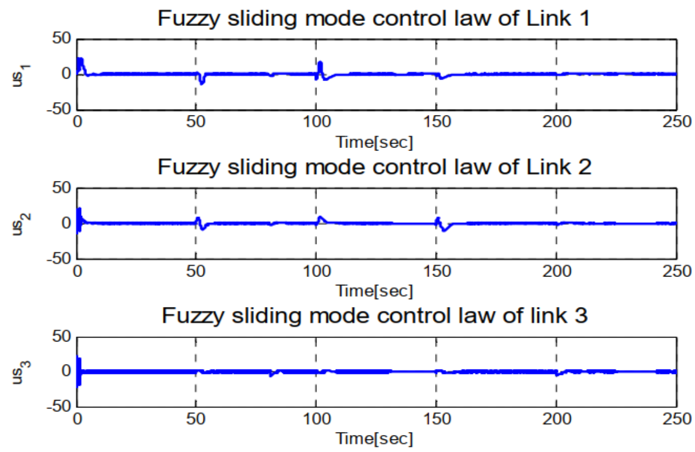

Figure 9. Sliding control law is estimated by fuzzy logic.

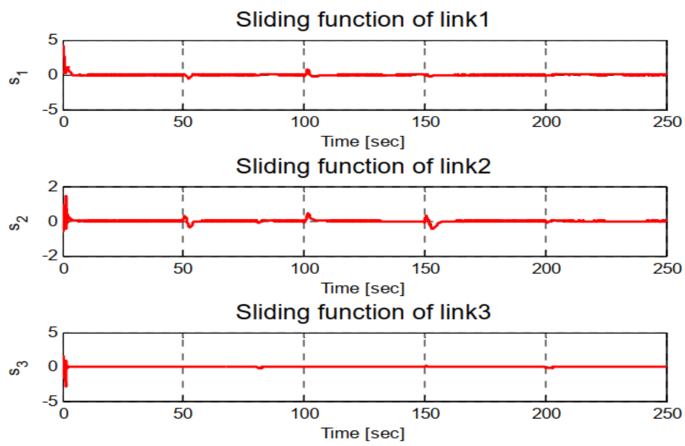

Figure 10. Sliding surface of manipulator system.

In order to assess the effectiveness of the controller proposed in this study, the author conducted simulations comparing it to the traditional PID controller with P = 1.43, I = 0.005, D = 0.5. Define the control objective: in this research, the aim is to manage the direction of movement effectively.

Establish the Kp coefficient: begin with the proportional coefficient, Kp, initially set to 0, and progressively increase it until achieving system stability. If the system exhibits slow response or instability, raise Kp. Conversely, if it displays excessive oscillations or an overly aggressive response, reduce Kp. The authors identified the optimal proportional coefficient as 1.43.

Determine the Ki coefficient: following the determination of Kp, ascertain the integral coefficient, Ki. Commence with an initial Ki value of 0 and incrementally raise it until achieving system stability and minimizing the error between output and desired values. In this article, a very small value of 0.005 was selected for the integral coefficient.

Specify the Kd coefficient: lastly, define the derivative coefficient, Kd. Initiate with an initial Kd value of 0 and progressively raise it until attaining system stability and reducing oscillations or waveform distortions in the response. The desired response necessitated a Kd coefficient of 0.5.

Testing and fine-tuning: once the Kp, Ki, and Kd coefficients are determined, assess the system to ensure it meets control requirements. Adjust the coefficient values as needed and retest until optimal performance is achieved for the robot arm.

The results indicated that the intelligent controller outperforms the traditional one, showcasing enhanced stability and precise tracking of the specified signal values as depicted in Figures 11–16. It is worth noting that all simulations were conducted under the presence of noise, simulating real-world underwater conditions (Section 2.4).

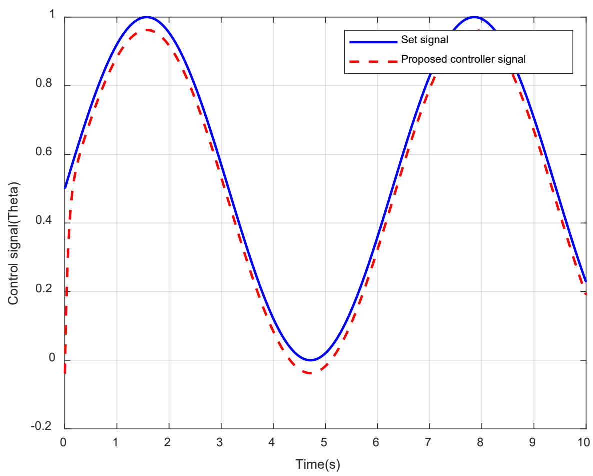

Figure 11. The control signal joint 1 of the Proposed controller signal.

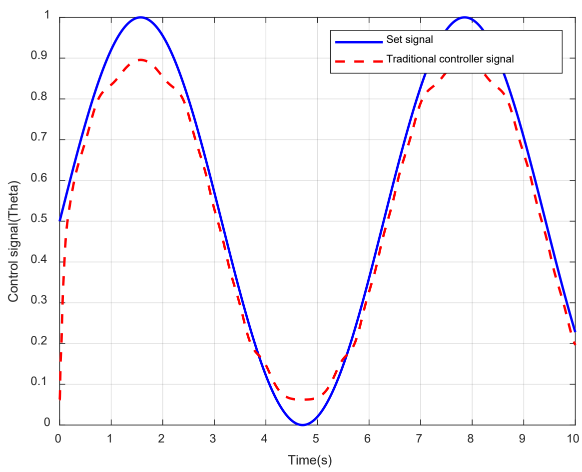

Figure 12. The control signal joint 1 of the Traditional PID controller signal.

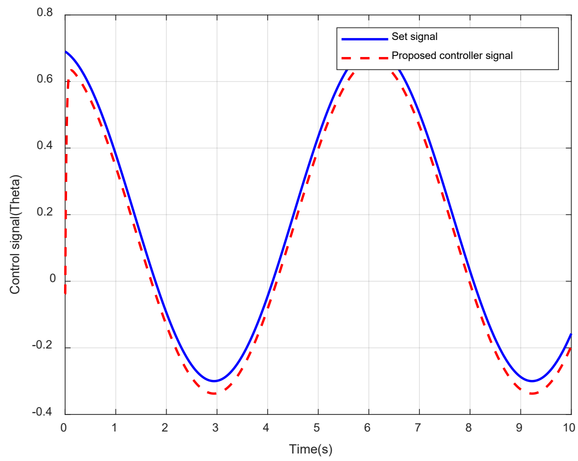

Figure 13. The control signal joint 2 of the Proposed controller signal.

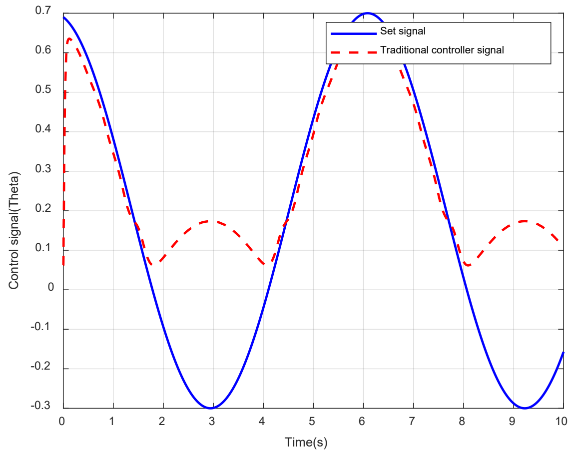

Figure 14. The control signal joint 2 of the Traditional PID controller signal.

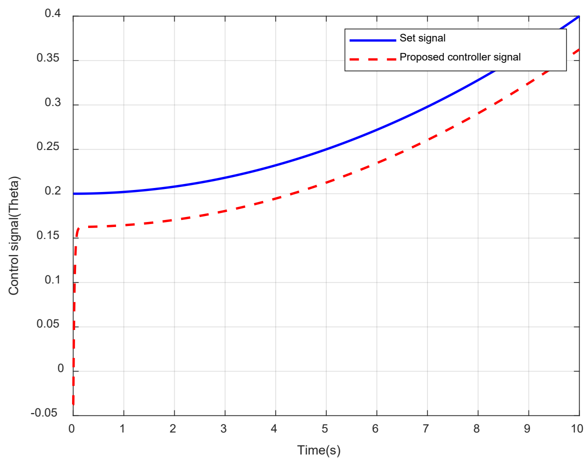

Figure 15. The control signal joint 3 of the Proposed controller signal.

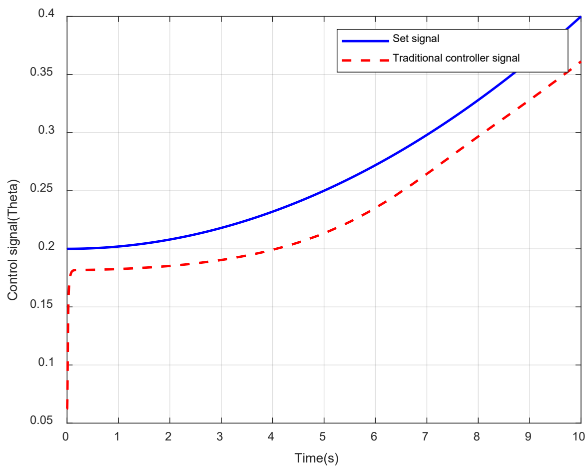

Figure 16. The control signal joint 3 of the Traditional PID controller signal.

Figures 11–14 clearly illustrate the significant improvement in the effectiveness of the proposed controller compared to the PID controller. Furthermore, Figures 15 and 16 exhibit nearly identical results, with the proposed controller demonstrating greater stability in the presence of noise compared to the PID controller.

The above examples highlight the enhanced noise elimination and stability achieved through the results.

Additionally, it is worth noting that, in the paper titled "Neural adaptive robust motion-tracking control for robotic manipulator systems" [39], the proposed control method employs neural networks to learn and adapt control parameters. However, this method does not utilize the sliding mode technique, whereas our proposed approach combines both sliding mode and fuzzy logic algorithms.

Similarly, in the paper titled "Robust control based on Adaptive Neural Network for the process of steady formation of continuous contact force in Unmanned Aerial Manipulator" [40], the control method relies on adaptive neural networks to determine control parameters. Nevertheless, it does not incorporate the sliding mode technique and is not designed to address the challenges of precise and stable control for underwater manipulators.

In our proposed control method, we integrate both sliding mode and fuzzy logic algorithms to achieve superior performance in controlling underwater manipulators. Additionally, we calculate and minimize the impact of noise on the control process.

To implement the controller, a combination of sliding mode control, neural network, and fuzzy logic control can be applied to control robotic arms in practical applications. Here are some fundamental steps for implementing this control system: define the robotic arm system objective and determine the specific objective of the robotic arm system, such as reaching a particular position or maintaining stability at a specific location. Identify the robotic arm system model: identify and understand the model of the robotic arm system, including crucial parameters such as position, velocity, and acceleration. Design the control model: based on the identified system model, design the control model, including the necessary parameters to achieve the system's objective. Calculate and program control equations: utilize the previously designed control equations to calculate the control parameters and program them into the control system. Set up the neural network: configure the neural network with the required parameters for controlling the robotic arm. Implement the fuzzy logic control system: set up the fuzzy logic control system and adjust related parameters to optimize the control system's performance. Combine control methods: integrate various control methods, including sliding mode control, neural network, and fuzzy logic control, to maximize efficiency in controlling the robotic arm. Test and fine-tune: after implementing the control methods, conduct testing and fine-tuning of parameters to achieve optimal efficiency in controlling the robotic arm system. In this study, the following results are displayed: Concordance with Reference Signal: 96%. $\pm\pi/2 \pm 3\%$ (Joint 1),

$\pm\pi/2 \pm 4\%$ (Joint 2), $\pm\pi/2 \pm 5\%$ (Joint 3), Rise Time: $5 \pm 3\%$ seconds for all joints, Steady-State Error (Ess): 0.68% (Joint 1), 1.24% (Joint 2), 1.32% (Joint 3). Comparing with existing research [41–43], the superiority of the proposed controller in this paper can be shown in terms of effective noise reduction and stability from the response time. Especially in the reference document [44] with a Response Accuracy of 93.4%, it is slightly lower compared to the proposed controller. Although it exhibits a superior Steady-State Error compared to the proposed controller, the Rise Time is somewhat longer and delayed compared to the controller addressed in the article.

4.2. Modeling Noise Signal

In this section, based on Section 2.4 and references [45–47], we aim to demonstrate the superiority of the system. The authors focus on modeling and analyzing common types of disturbances that might affect the robotic arm of the ROV (Remote Operated Vehicle) when operating in a marine environment. The working conditions are assumed to be standard and not extremely harsh, excluding extreme factors such as very high pressures or very low temperatures.

4.2.1. Modeling Disturbances

Authors identified two main types of disturbances: deterministic disturbances and random disturbances.

Deterministic Disturbances

Deterministic disturbances usually arise from fixed or cyclical sources, like the engines and electrical equipment on the ROV. Specifically:

Electrical frequency noise: originates from the ROV's power source, can be modeled using a sine function with a fixed frequency.

Engine noise: caused by engine vibrations, can be simulated using a sine function with varying amplitude and frequency.

Random Disturbances

Random disturbances arise from the marine environment and other unpredictable factors:

Noise from waves and currents: affected by weather conditions and underwater terrain, can be modeled using a random function.

Collision noise: occurs when the robotic arm contacts an underwater object, creating uneven and unpredictable forces.

4.2.2. Origin, Amplitude, Frequency, and Impact of Disturbances

Electrical frequency noise: originates from the ROV's electrical system, has fixed amplitude and frequency. Affects control signals, potentially leading to errors.

Engine noise: comes from engine vibrations, amplitude and frequency and depends on the engine's speed and operational state. Affects the accuracy of movements.

Noise from waves and currents: amplitude and frequency are indeterminate, dependent on weather conditions and terrain. Significantly impacts the robotic arm's positioning and stability.

Collision noise: occurs randomly when the robotic arm interacts with an object, affects stability and precision of operations.

4.2.3. Disturbance Calculation Formulas

To model the aforementioned disturbances, we employ the following formulas:

Electrical Frequency Noise: $N_1(t) = A_1 \sin(2\pi f_1 t + \varnothing_1)$

Engine Noise: $N_2(t) = A_2 \sin(2\pi f_2 t + \varnothing_2)$

Noise from Waves and Currents: $N_3(t) = A_3 \cdot \mathrm{rand}()$

Collision Noise: $N_4(t) = A_4 \cdot \mathrm{rand}()$

Where: A represents amplitude, f is the frequency, ∅ is the phase, and rand() denotes random generation functions.

4.2.4. MATLAB Simulation Results

Using MATLAB, the authors simulated the noise signals to analyze their impact on the robotic arm system (Figure 17).

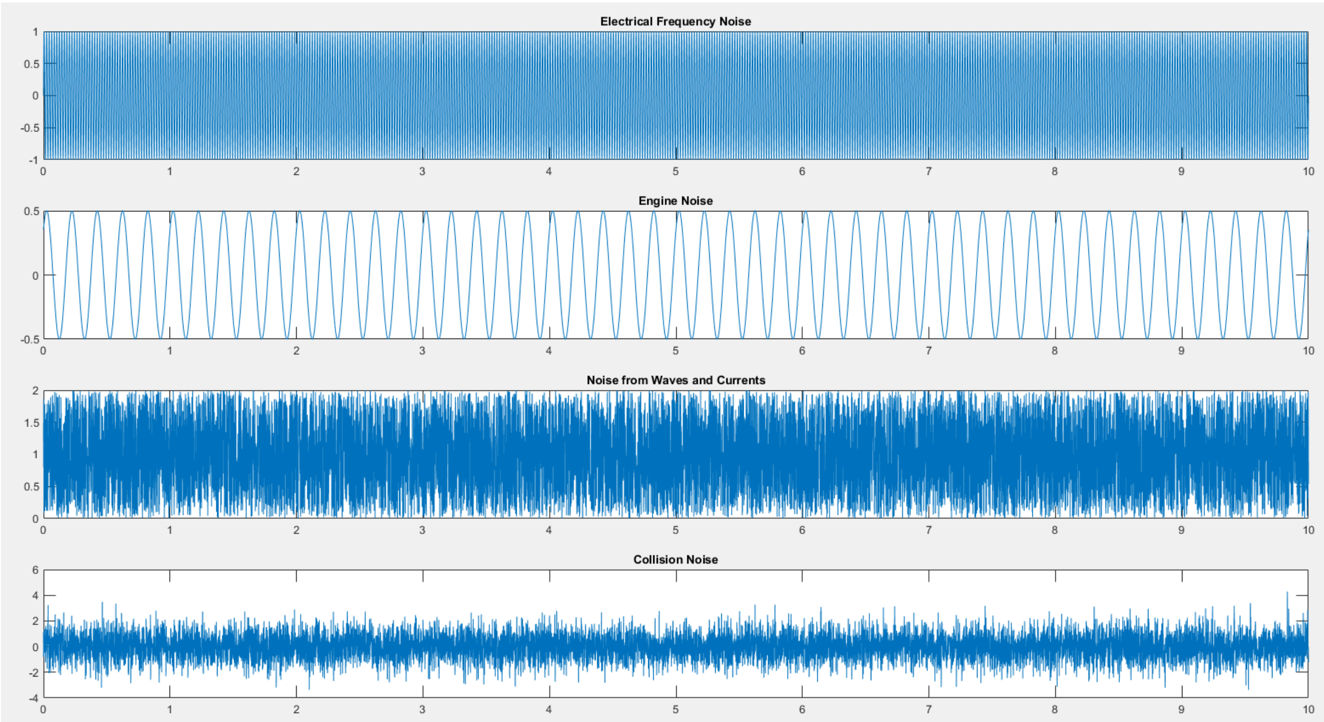

Figure 17. The noise signals to analyze their impact on the robotic arm system.

Based on Figure 17, authors can identify various types of disturbances affecting the underwater robotic arm system. Electrical frequency noise is consistent and unchanging, impacting the system's signal quality. Motor noise occurs periodically, causing instability in the robot's movement. Wave and flow noise is irregular and tends to be random, affecting the robot's control ability. Collision noise is unpredictable and can damage the robot. Recognizing and handling these disturbances is vital to enhance performance and reduce noise. Improving design and employing smart algorithms will boost performance and ensure safety. The authors introduced the use of Linear Parameter-Varying (LPV) Systems and the Kalman filter to reduce noise. LPV helps model the non-linear system under various conditions. Meanwhile, the Kalman filter assists in estimating the system's real state and reducing disturbances. The combination of LPV and the Kalman filter optimizes the operation of the underwater robot in a complex environment.

4.3. Applying Linear Parameter-Varying and Kalman Models to Underwater Robot Manipulators

4.3.1. Approach to the LPV (Linear Parameter-Varying) Model [48,49] for Underwater Robotic Arms

4.3.1.1. Introduction to the LPV Model

The underwater environment is particularly unpredictable with various factors such as currents, pressure, and temperature that may change continuously. The LPV allows controllers to self-adjust their parameters based on measured or estimated variables, enabling the robotic arm to adapt to varying working conditions and maintain stable and accurate performance. LPV provides the capability to optimize the controller based on specific working conditions at any given time, ensuring that the robotic arm operates as efficiently as possible in every situation. The LPV model is used to describe systems whose dynamics vary with time and/or depend on system state and inputs. This makes it particularly suited for modeling nonlinear systems like underwater robotic arms, where environmental factors like pressure, depth, and currents can affect the system's dynamics. LPV offers a flexible control solution, capable of self-adjusting to address the uncertainties and variations of the system and its environment. The controller can be designed to best reflect performance in each specific situation, allowing the robotic arm to operate at its most efficient.

4.3.1.2. Construction of the LPV Model

The LPV model can be represented as a linear equation as follows:

$$\dot{x}(t) = A(p)x(t) + B(p)u(t) \tag{80}$$

$$y(t) = C(p)x(t) + D(p)u(t) \tag{81}$$

where:

$x(t)$ is the system state vector.
$u(t)$ is the system input vector.
$y(t)$ is the system output vector.
p is the vector of varying parameters, dependent on state and inputs.
$A(p)$, $B(p)$, $C(p)$, $D(p)$ are matrices dependent on parameter p.

For underwater robotic arms, p might include factors such as depth, pressure, and water temperature.

Basic parameters of the robotic arm are:
Segment lengths: $l_1 = 1.0$ m, $l_2 = 0.8$ m, $l_3 = 0.6$ m.
Segment masses: $m_1 = 1$ kg, $m_2 = 0.8$ kg, $m_3 = 0.5$ kg.
Joint stiffness: $k_1 = 100$ Nm/rad, $k_2 = 80$ Nm/rad, $k_3 = 50$ Nm/rad.
Arm radii: $r = 0.05$ m
Assuming that state variables are angles and velocities of each joint:

$$x = \begin{bmatrix} \theta_1 & \dot{\theta}_1 & \theta_2 & \dot{\theta}_2 & \theta_3 & \dot{\theta}_3 \end{bmatrix} \tag{82}$$

Inputs are torques applied to each joint:

$$u = \begin{bmatrix} \tau_1 & \tau_2 & \tau_3 \end{bmatrix}^T \tag{83}$$

The system matrices A and B are built based on the linearization of the dynamic system. After calculations and simplification, the matrices are obtained as follows:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ \frac{-k_1}{m_1} & 0 & \frac{k_1}{m_1} & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ \frac{k_2}{m_2} & 0 & \frac{-k_2}{m_2} & 0 & \frac{k_2}{m_2} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & \frac{k_3}{m_3} & 0 & \frac{-k_3}{m_3} & 0 \end{bmatrix} \tag{84}$$

$$B = \begin{bmatrix} 0 & 0 & 0 \\ \frac{1}{m_1} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & \frac{1}{m_2} & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \frac{1}{m_3} \end{bmatrix} \tag{85}$$

Measurement matrix:

$$C = I_6 \tag{86}$$

$$D = 0_{6\times 3} \tag{87}$$

Covariance Matrix:

$$Q = \mathrm{diag}\left(\begin{bmatrix} 0.01 & 0.01 & 0.01 & 0.01 & 0.01 & 0.01 \end{bmatrix}\right) \tag{88}$$

$$R = \left(\begin{bmatrix} 0.05 & 0.05 & 0.05 \end{bmatrix}\right) \tag{89}$$

4.3.2. Development of the Extended Kalman Filter (EKF) for Underwater Robotic Arms

4.3.2.1. Introduction to the Extended Kalman Filter (EKF)

In underwater environments, direct measurement of the states like position, velocity, and direction can be disturbed by factors such as waves, currents, or external influences. The Kalman filter aids in estimating these states more accurately by combining information from the system model and measurement data. It uses both system model information and measurement data to provide high-accuracy state estimates, reducing noise impact. The Extended Kalman Filter (EKF) [50] is a powerful tool for estimating system states in the LPV model, especially when the system is nonlinear and noisy. The EKF is a variant of the standard Kalman filter designed for nonlinear systems. It provides accurate and reliable state estimates, even under noisy conditions. The filter can quickly respond to changes in measurement data and system modeling.

The Kalman Gain K is a crucial factor in the EKF, playing a vital role in balancing between prediction and measurement. It ensures that measurement information is reasonably incorporated into the system's state estimation. K is determined based on the covariance matrix of the prediction, the measurement covariance matrix, and the matrix relating estimated states to measurements. The Kalman Gain K is computed to optimize the state estimation. To ensure stability and efficiency, K should be calculated so that the EKF's efficiency function (estimation error covariance) is minimized. The formula for computing the optimal K is based on the principle of minimizing estimation errors.

4.3.2.2. Extended Kalman Filter (EKF) Formula

A priori estimation (before receiving new measurement data):

$$\hat{x}_{k|k-1} = A_k \hat{x}_{k-1} + B_k u_{k-1} \tag{90}$$

$$P_{k|k-1} = A_k P_{k-1} A_k^T + Q_k \tag{91}$$

Calculating the Kalman Gain coefficient:

$$K_k = P_{k|k-1} C_k^T \left(C_k P_{k|k-1} C_k^T + R_k \right)^{-1} \tag{92}$$

A posteriori update (after receiving new measurement data):

$$\hat{x}_k = \hat{x}_{k|k-1} + K_k \left(y_k - C_k \hat{x}_{k|k-1} \right) \tag{93}$$

$$P_k = (I - K_k C_k) P_{k|k-1} \tag{94}$$

where:

$\overline{x}_k$: State estimation at time k.

$A_k,\ B_k,\ C_k$: System model matrices.

u_{k-1}: Control input at time $k-1$.

P_k: State estimation covariance matrix.
Q_k: Process noise covariance matrix.
R_k: Measurement noise covariance matrix.
y_k: Measurement data at time k.
K_k: Kalman Gain, balancing between model estimation and measurement data.

4.3.3. Simulation of LPV Combined with Kalman

Figure 18 below shows the impact of disturbance on the joints of a manipulator working underwater when applying LPV and Kalman and vice versa.

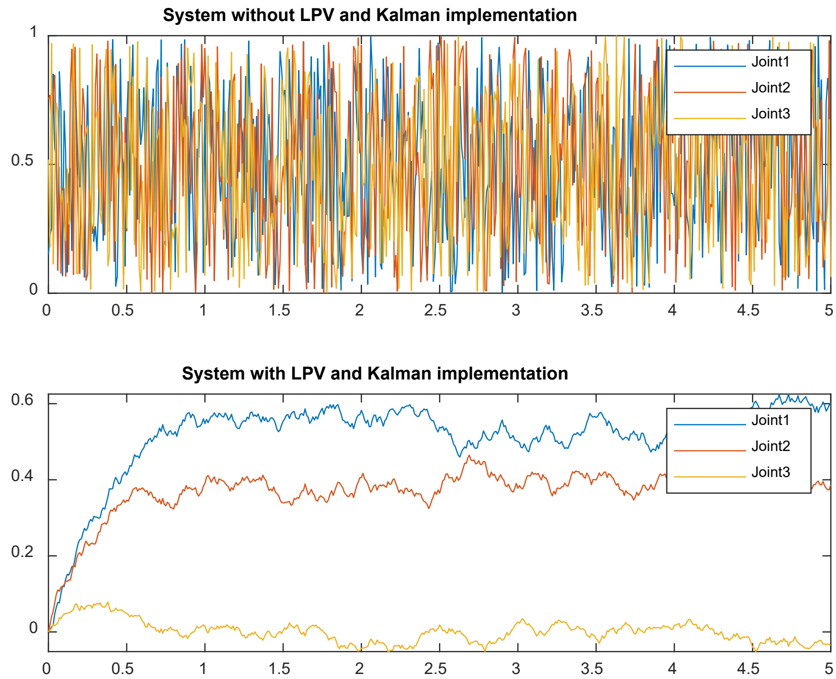

Figure 18. The impact of disturbance on the joints of a manipulator.

Analysis of Figure 18:

System without LPV and Kalman implementation:

The data represents three series (joint 1, joint 2, and joint 3) indicating strong fluctuations and noise affecting the robot arm joints. The variability of these three series is very high, indicating the system is heavily disturbed and it is challenging to track or predict its next state.

System with LPV and Kalman implementation:

Upon employing LPV and the Kalman filter, fluctuations in the three data series (joint 1, joint 2, and joint 3) are significantly reduced. The series joint 1, joint 2, and joint 3 are smoother and easier to follow, demonstrating that the impact of the noise has considerably diminished in frequency and magnitude on the system's operation. Compared to the above graph, noise and variability have greatly decreased, proving the effectiveness of implementing LPV and the Kalman filter.

5. Conclusions

Through this paper, a sliding mode controller based on fuzzy control and RBFNN network is proposed to monitor the operating trajectory of the robot controller. An adaptive rule is used to tune online the weights of the RBFNN, which are used to compute the equivalence control. The adaptive training algorithm is derived from the meaning of the Lyapunov stability analysis, so that the stability of the closed-loop system can be guaranteed even in the case of uncertainty. Using RBFNN, instead of a multi-layer feed-forward network trained with backpropagation, shortens the reach of time. Problem-solving in SMC is minimized with the proposed controller. The simulation results show that the joint position tracking responses follow the desired trajectories of perturbations and frictional forces. In addition, the simulation results demonstrate that the fuzzy sliding mode controller based on the radial basis function neural network proposed in this paper is a stable control scheme for monitoring applications. Close the trajectory of the robot control mechanism. The adaptive sliding controller using a fuzzy neural model has indicated the adaptability of both noise and time-varying object parameters. In this controller, we use fuzzy logic to estimate the control law amplitude online to limit the oscillation. At the same time, using the RBFNN network determines control law based on nonlinear function estimation to ensure a stable closed system.

The chart indicates that the implementation of LPV in conjunction with the Kalman filter has significantly enhanced the system's performance. In underwater environments, where numerous factors introduce noise to the data, employing these methods proves highly effective in noise reduction and bolstering the accuracy of state estimations.

In practice, this results in the underwater robotic arm operating more stably, with greater precision and reliability, subsequently enhancing its operational capability and overall efficiency.

The control strategy we have introduced holds immense potential for its use in real-world underwater robotics. As we look to the future, our next steps involve applying this method to tangible robotic arm models and analyzing its efficacy through hands-on underwater trials. Moreover, with the advent of groundbreaking methods like deep reinforcement learning and bio-inspired optimization, there is ample room to refine our control algorithm. Delving into the incorporation of state-of-the-art sensors for precise state feedback and better disturbance handling is another exciting prospect. Doing so will significantly address the real-world challenges that underwater operations often face. Furthermore, enhancing the controller to manage issues like changes in hydrodynamic parameters, nonlinear friction interferences, and overlooked dynamics becomes a pivotal research avenue. In essence, our efforts pave the way for pioneering intelligent control systems that enable underwater robots to carry out intricate operations seamlessly and with unwavering accuracy.

This project serves as a foundation for future research on a practical model for stabilizing robot arm control through a proposed control system and intelligence, enhancing adaptability through computer vision.

Author Contributions: D.-A.P.; Formal analysis, software, resources, data curation, writing—original draft preparation, data collected and analyzed, visualization, S.-H.H.; supervision, project administration, writing—review and editing. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the research grant of the Gyeongsang National University in 2023.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data and equipment are available at Common Pressure and Marine System Laboratory in Gyeongsang Nation University.

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

Nomenclature

ROV	Remotely Operated Vehicle
RBFNN	Radial Basis Function Neural Networks
SMC	Sliding Mode Control
FIS	Fuzzy Inference System
LPV	Linear Parameter Varying
EKF	Extended Kalman Filter
CNN	Convolutional Neural Networks
LSTM	Long Short-Term Memory
FNN	Fuzzy Neural Network
PID	Proportional Integral Derivative

Appendix A

From all the above data, programming the fuzzy rule definition of the problem in Algorithm A1 below:

Algorithm A1. Fuzzy set algorithm.

```
[System]
Name = 'Fuzzyset'
Type = 'mamdani'
Version = 2.0
NumInputs = 1
NumOutputs = 1
NumRules = 7
AndMethod = 'min'
OrMethod = 'max'
ImpMethod = 'min'
AggMethod = 'max'
DefuzzMethod = 'centroid'

[Si]
Name = '|Si|'
Range = [0 1]
NumMFs = 7
MF1 = 'NB':'gaussmf', [0.07078 0]
MF2 = 'NM':'gaussmf', [0.07078 0.1667]
MF3 = 'NS':'gaussmf', [0.07078 0.3333]
MF4 = 'Z':'gaussmf', [0.07078 0.5]
MF5 = 'PS':'gaussmf', [0.07078 0.6667]
MF6 = 'PM':'gaussmf', [0.07078 0.8333]
MF7 = 'PB':'gaussmf', [0.07078 1]

[Ki]
Name = 'Ki'
Range = [0 1]
NumMFs = 7
MF1 = 'NB':'gaussmf', [0.07078 0]
MF2 = 'NM':'gaussmf', [0.07078 0.1667]
MF3 = 'NS':'gaussmf', [0.07078 0.3333]
MF4 = 'Z':'gaussmf', [0.07078 0.5]
MF5 = 'PS':'gaussmf', [0.07078 0.6667]
MF6 = 'PM':'gaussmf', [0.07078 0.8333]
MF7 = 'PB':'gaussmf', [0.07078 1]

[Rules]
1, 1 (1): 1
2, 2 (1): 1
3, 3 (1): 1
4, 4 (1): 1
5, 5 (1): 1
6, 6 (1): 1
7, 7 (1): 1
```

References

1. Capocci, R.; Dooly, G.; Omerdić, E.; Coleman, J.; Newe, T.; Toal, D. Inspection-Class Remotely Operated Vehicles—A Review. *J. Mar. Sci. Eng.* **2017**, *5*, 13. [CrossRef]
2. Konoplin, A.; Konoplin, N.; Yurmanov, A. Development and Field Testing of a Smart Support System for ROV Operators. *J. Mar. Sci. Eng.* **2022**, *10*, 1439. [CrossRef]
3. Chen, Q.; Hu, Y.; Zhang, Q.; Jiang, J.; Chi, M.; Zhu, Y. Dynamic Damping-Based Terminal Sliding Mode Event-Triggered Fault-Tolerant Pre-Compensation Stochastic Control for Tracked ROV. *J. Mar. Sci. Eng.* **2022**, *10*, 1228. [CrossRef]
4. Yin, F.; Wu, S.; Huang, H.; Cui, C.; Ji, Q. Effect of Machining Trajectory on Grinding Force of Complex-Shaped Stone by Robotic Manipulator. *Machines* **2022**, *10*, 787. [CrossRef]
5. Luo, J.; Zhu, L.; Wu, N.; Chen, M.; Liu, D.; Zhang, Z.; Liu, J. Adaptive Neural-PID Visual Servoing Tracking Control via Extreme Learning Machine. *Machines* **2022**, *10*, 782. [CrossRef]
6. Erdemir, A.; Kalyoncu, M. Modeling Impedance Control with Limited Interaction Power for A 2R Planar Robot Arm. In Proceedings of the 4th Latin American International Congress on Natural and Applied Sciences, Rio de Janeiro, Brazil, 4–6 September 2023.
7. Li, C.; Chen, X.; Ma, X.; Sun, H.; Wang, B. Skill Acquisition and Con-troller Design of Desktop Robot Manipulator Based on Audio–Visual Information Fusion. *Machines* **2022**, *10*, 772. [CrossRef]
8. Moran, M.E. Evolution of robotic arms. *J. Robot. Surg.* **2007**, *1*, 103–111. [CrossRef]
9. Efe, M.O.; Kaynak, O.; Yu, X. Sliding Mode Control of a Three Degrees of Freedom Anthropoid Robot by Driving the Controller Parameters to an Equivalent Regime. *J. Dyn. Syst. Meas. Control* **2000**, *122*, 632–640. [CrossRef]
10. Wilamowski, B.M.; Iplikci, S.; Kaynak, O.; Efe, M.O. An algorithm for fast convergence in training neural networks. In Proceedings of the IJCNN'01 International Joint Conference on Neural Networks Proceedings (Cat. No. 01CH37222), Washington, DC, USA, 15–19 July 2002.
11. Perruquetti, W.; Barbot, J.P. *Sliding Mode Control in Engineering*; Marcel Dekker, Inc.: New York, NY, USA, 2002.
12. Von Benzon, M.; Sørensen, F.F.; Uth, E.; Jouffroy, J.; Liniger, J.; Pedersen, S. An Open-Source Benchmark Simulator: Control of a BlueROV2 Underwater Robot. *J. Mar. Sci. Eng.* **2022**, *10*, 1898. [CrossRef]
13. Miao, H.; Diao, P.; Yao, W.; Li, S.; Wang, W. Stability Study of Time Lag Disturbance in an Automatic Tractor Steering System Based on Sliding Mode Predictive Control. *Agriculture* **2022**, *12*, 2091. [CrossRef]
14. Shi, Q.; Wang, H.; Cheng, H. Multiple Constraints-Based Adaptive Three-Dimensional Back-Stepping Sliding Mode Guidance Law against a Maneuvering Target. *Aerospace* **2022**, *9*, 796. [CrossRef]
15. Bao, H.; Zhu, H.; Li, X.; Liu, J. APSO-MPC and NTSMC Cascade Control of Fully-Actuated Autonomous Underwater Vehicle Trajectory Tracking Based on RBFNN Compensator. *J. Mar. Sci. Eng.* **2022**, *10*, 1867. [CrossRef]
16. Xia, X.; Jia, Y.; Wang, X.; Zhang, J. Neural-Network-Based Terminal Sliding Mode Control of Space Robot Actuated by Control Moment Gyros. *Aerospace* **2022**, *9*, 730. [CrossRef]
17. Sadati, N. A novel approach to coordination of large-scale systems; part I interaction prediction principle. In Proceedings of the 2005 IEEE International Conference on Industrial Technology, Hong Kong, China, 14–17 December 2005.
18. Ak, A.G.; Cansever, G. Three link robot control with fuzzy sliding mode controller based on RBF neural network. In Proceedings of the 2006 IEEE Conference on Computer Aided Control System Design, 2006 IEEE International Conference on Control Applications, 2006 IEEE International Symposium on Intelligent Control, Munich, Germany, 4–6 October 2006.
19. Yang, Y.; Wang, Y.; Zhang, W.; Li, Z.; Liang, R. Design of Adaptive Fuzzy Sliding-Mode Control for High-Performance Islanded Inverter in Micro-Grid. *Energies* **2022**, *15*, 9154. [CrossRef]
20. Yan, X.G.; Spurgeon, S.K.; Edwards, C. Variable structure control of complex systems. In *Communications and Control Engineering*; Springer: Cham, Switzerland, 2017.
21. Elangovan, S.; Woo, P.Y. Adaptive fuzzy sliding control for a three-link passive robotic manipulator. *Robotica* **2005**, *23*, 635–644. [CrossRef]
22. Aydin, M.; Yakut, O. Implementation of Sliding Surface Moving Anfis Based Sliding Mode Control to Rotary Inverted Pendulum. *J. Inst. Sci. Technol.* **2023**, *13*, 1165–1175. [CrossRef]
23. Chand, A.; Khan, Q.; Alam, W.; Khan, L.; Iqbal, J. Certainty equivalence-based robust sliding mode control strategy and its application to uncertain PMSG-WECS. *PLoS ONE* **2023**, *18*, e0281116. [CrossRef]
24. Zheng, W.; Chen, Y.; Wang, X.; Chen, Y.; Lin, M. Enhanced fractional order sliding mode control for a class of fractional order uncertain systems with multiple mismatched disturbances. *ISA Trans.* **2023**, *133*, 147–159. [CrossRef]
25. Phuoc, P.D. *Industrial Robots*; Construction Publishing House: Hanoi, Vietnam, 2007.
26. Wai, R.-J. Fuzzy Sliding-Mode Control Using Adaptive Tuning Technique. *IEEE Trans. Ind. Electron.* **2007**, *54*, 586–594. [CrossRef]
27. Yu, X.; Man, Z.; Wu, B. Design of fuzzy sliding-mode control systems. *Fuzzy Sets Syst.* **1998**, *95*, 295–306. [CrossRef]
28. Alli, H.; Yakut, O. Fuzzy sliding-mode control of structures. *Eng. Struct.* **2005**, *27*, 277–284. [CrossRef]
29. Yoo, B.; Ham, W. Adaptive fuzzy sliding mode control of nonlinear system. *IEEE Trans. Fuzzy Syst.* **1998**, *6*, 315–321.
30. Wang, J.; Zhu, P.; He, B.; Deng, G.; Zhang, C.-L.; Huang, X. An Adaptive Neural Sliding Mode Control with ESO for Uncertain Nonlinear Systems. *Int. J. Control Autom. Syst.* **2021**, *19*, 687–697. [CrossRef]
31. Park, B.S.; Yoo, S.J.; Park, J.B.; Choi, Y.H. Adaptive Neural Sliding Mode Control of Nonholonomic Wheeled Mobile Robots with Model Uncertainty. *IEEE Trans. Control Syst. Technol.* **2009**, *17*, 207–214. [CrossRef]

32. Fei, J.; Ding, H. Adaptive sliding mode control of dynamic system using RBF neural network. *Nonlinear Dyn.* **2012**, *70*, 1563–1573. [CrossRef]
33. Huynh, T.H. *Intelligent Control System*; National University Publishing House: Ho Chi Minh City, Vietnam, 2006.
34. Pham, D.-A.; Han, S.-H. Design of Combined Neural Network and Fuzzy Logic Controller for Marine Rescue Drone Trajectory-Tracking. *J. Mar. Sci. Eng.* **2022**, *10*, 1716. [CrossRef]
35. Fullér, R. *Neural Fuzzy Systems*; Turku Center for Computer Science: Turku, Finland, 1995.
36. Lin, C.-T.; Lee, C.G. *Neural Fuzzy Systems: A Neuro-Fuzzy Synergism to Intelligent Systems*; Computing Reviews; Prentice-Hall, Inc.: Hoboken, NJ, USA, 1996.
37. Juang, C.-F.; Lin, C.-T. A recurrent self-organizing neural fuzzy inference network. *IEEE Trans. Neural Netw.* **1999**, *10*, 828–845. [CrossRef]
38. Lin, C.-T.; Lu, Y.-C. A neural fuzzy system with fuzzy supervised learning. *IEEE Trans. Syst. Man Cybern. Part B (Cybern.)* **1996**, *26*, 744–763.
39. Galvan-Perez, D.; Yañez-Badillo, H.; Beltran-Carbajal, F.; Rivas-Cambero, I.; Favela-Contreras, A.; Tapia-Olvera, R. Neural Adaptive Robust Motion-Tracking Control for Robotic Manipulator Systems. *Actuators* **2022**, *11*, 255. [CrossRef]
40. Fang, Q.; Mao, P.; Shen, L.; Wang, J. Robust Control Based on Adaptive Neural Network for the Process of Steady Formation of Continuous Contact Force in Unmanned Aerial Manipulator. *Sensors* **2023**, *23*, 989. [CrossRef]
41. Malki, H.A.; Misir, D.; Feigenspan, D.; Chen, G. Fuzzy PID control of a flexible-joint robot arm with uncertainties from time-varying loads. *IEEE Trans. Control Syst. Technol.* **1997**, *5*, 371–378. [CrossRef]
42. De Jesús Rubio, J. Sliding mode control of robotic arms with deadzone. *IET Control Theory Appl.* **2017**, *11*, 1214–1221. [CrossRef]
43. Al-Darraji, I.; Piromalis, D.; Kakei, A.A.; Khan, F.Q.; Stojmenovic, M.; Tsaramirsis, G.; Papageorgas, P.G. Adaptive robust controller design-based RBF neural network for aerial robot arm model. *Electronics* **2021**, *10*, 831. [CrossRef]
44. Rakshit, A.; Pramanick, S.; Bagchi, A.; Bhattacharyya, S. Autonomous grasping of 3-D objects by a vision-actuated robot arm using Brain–Computer Interface. *Biomed. Signal Process. Control* **2023**, *84*, 104765. [CrossRef]
45. Fernández, J.J.; Prats, M.; Sanz, P.J.; García, J.C.; Marin, R.; Robinson, M.; Ribas, D.; Ridao, P. Grasping for the seabed: Developing a new underwater robot arm for shallow-water intervention. *IEEE Robot. Autom. Mag.* **2013**, *20*, 121–130. [CrossRef]
46. Phillips, B.T.; Becker, K.P.; Kurumaya, S.; Galloway, K.C.; Whittredge, G.; Vogt, D.M.; Teeple, C.B.; Rosen, M.H.; Pieribone, V.A.; Gruber, D.F.; et al. A dexterous, glove-based teleoperable low-power soft robotic arm for delicate deep-sea biological exploration. *Sci. Rep.* **2018**, *8*, 14779. [CrossRef]
47. Gharaibeh, K.M. *Nonlinear Distortion in Wireless Systems: Modeling and Simulation with MATLAB*; John Wiley & Sons: Hoboken, NJ, USA, 2011.
48. Mohammadpour, J.; Scherer, C.W. (Eds.) *Control of Linear Parameter Varying Systems with Applications*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2012.
49. Zhang, H.; Li, P.; Du, H.; Wang, Y.; Nguyen, A.T. Guest Editorial: Emerging Trends in LPV-Based Control of Intelligent Automotive Systems. *IET Control Theory Appl.* **2020**, *14*, 2715–2716.
50. Ribeiro, M.I. Kalman and extended kalman filters: Concept, derivation and properties. *Inst. Syst. Robot.* **2004**, *43*, 3736–3741.