

Article

# Neural Network-Based Self-Tuning PID Control for Underwater Vehicles

Rodrigo Hernández-Alvarado <sup>1,\*†‡</sup>, Luis Govinda García-Valdovinos <sup>1‡</sup>,  
Tomás Salgado-Jiménez <sup>1‡</sup>, Alfonso Gómez-Espinosa <sup>2‡</sup> and Fernando Fonseca-Navarro <sup>1‡</sup>

<sup>1</sup> Energy Division, Center for Engineering and Industrial Development-CIDESI, Santiago de Queretaro, Queretaro 76125, Mexico; ggarcia@cidesi.edu.mx (L.G.G.-V.); tsalgado@cidesi.edu.mx (T.S.-J.); ffonseca@posgrado.cidesi.edu.mx (F.F.-N.)

<sup>2</sup> Tecnológico de Monterrey, Campus Queretaro, Ave. Epigmenio González 500, Fracc. San Pablo, Santiago de Queretaro, Queretaro 76130, Mexico; agomeze@itesm.mx

\* Correspondence: rodrigoherz@gmail.com or rohernandez@cidesi.edu.mx

† Current address: Av. Playa Pie de la Cuesta No. 702, Desarrollo San Pablo, Santiago de Queretaro, Queretaro 76125, Mexico.

‡ These authors contributed equally to this work.

Academic Editor: Leonhard M. Reindl

Received: 24 May 2016; Accepted: 10 August 2016; Published: 5 September 2016

**Abstract:** For decades, PID (Proportional + Integral + Derivative)-like controllers have been successfully used in academia and industry for many kinds of plants. This is thanks to its simplicity and suitable performance in linear or linearized plants, and under certain conditions, in nonlinear ones. A number of PID controller gains tuning approaches have been proposed in the literature in the last decades; most of them off-line techniques. However, in those cases wherein plants are subject to continuous parametric changes or external disturbances, online gains tuning is a desirable choice. This is the case of modular underwater ROVs (Remotely Operated Vehicles) where parameters (weight, buoyancy, added mass, among others) change according to the tool it is fitted with. In practice, some amount of time is dedicated to tune the PID gains of a ROV. Once the best set of gains has been achieved the ROV is ready to work. However, when the vehicle changes its tool or it is subject to ocean currents, its performance deteriorates since the fixed set of gains is no longer valid for the new conditions. Thus, an online PID gains tuning algorithm should be implemented to overcome this problem. In this paper, an auto-tune PID-like controller based on Neural Networks (NN) is proposed. The NN plays the role of automatically estimating the suitable set of PID gains that achieves stability of the system. The NN adjusts online the controller gains that attain the smaller position tracking error. Simulation results are given considering an underactuated 6 DOF (degrees of freedom) underwater ROV. Real time experiments on an underactuated mini ROV are conducted to show the effectiveness of the proposed scheme.

**Keywords:** neural networks; auto-tuning PID; ROV control; disturbances

## 1. Introduction

Underwater Remotely Operated Vehicles (ROVs) have been widely used in many subsea tasks, ranging from inspection to repair of underwater structures related mainly to the power and oil industry. Very often, according to the task, the ROV is required to continuously change its operating tool and/or to pick up and release loads causing a change in behavior. That results as an inherent change in its weight, buoyancy and hydrodynamic forces; and as a consequence, a decrease in the position tracking performance. In addition, ROVs have to deal with the highly dynamical underwater environment represented in the form of ocean currents and waves in shallow water. With this in mind, when the dynamic characteristics of the system are time dependent or the operating conditions of the system vary,

it is necessary to re-tune the gains to obtain the desired performance, resulting in time consumption. In this paper, a self-tuning algorithm based on Neural Networks (NN) is proposed to automatically tune the gains of a PID (Proportional + Integral + Derivative) controller. The optimal set of gains is computed online with less computation effort by using desired and actual state variables. The self-tuning mechanism will avoid time consuming manual tuning of the PID controller and promises better results by providing PID controller settings as the system dynamics or operating points change.

With this in mind, a mix of control and a smart system might offer an accurate tune of the control gains online. Even when the state of art yields different tuning techniques, it is common to find controls poorly tuned such that their performance is limited. Intelligent control techniques include fuzzy control, neural networks or a mix of them; they have been widely used to control underwater and nonlinear systems, such as in [1–3], and have become an accurate option, though these algorithms do require long periods of training and tuning.

Control schemes vary from tracking to dynamic positioning [4,5] where their main target is to estimate and compensate for the unknown forces of changing environments. Research [6–14] present systems with a mix of neural networks and fuzzy control in which the training and rules of behavior are based on the desired states. Their performance is described as accurate when uncertainty and perturbations take place while performing a trajectory. Although the training periods are extremely long, there are also combinations of PID controls and a smart system aimed to auto-tune the gains of different systems such as: sub-aquatic [15,16], non linear [17–22], and others: [23–26].

In this paper, an auto-tune PID-like controller based on an online Neural Networks (NN) is implemented on Remotely Operated Vehicles (ROVs); for trajectory tracking with unknown disturbances. Simulation results are given considering the non-linear hydrodynamics of ROV Kaxan; including disturbances of ocean currents. Real time experiments on an underactuated mini ROV are conducted to show the effectiveness of the proposed scheme. For the remaining sections of this paper in Section 2 the general system model of 6 DOF underwater vehicles is presented, Section 3 includes the effect of ocean currents, Section 4 presents the Self-tuning Neural Network for PID Control, Section 5 describes the simulation results, and the experimental results are presented in Section 6; Finally in Section 7 the concluding remarks are provided.

## 2. General System Model of 6 DOF Underwater Vehicles

In [27], the nonlinear model of a 6 DOF to build the mathematical model that represents the underwater vehicle dynamics two reference frames were used; one referenced to earth (called the Earth-fixed frame) and another referenced to the vehicle (called the body-fixed frame), Figure 1.

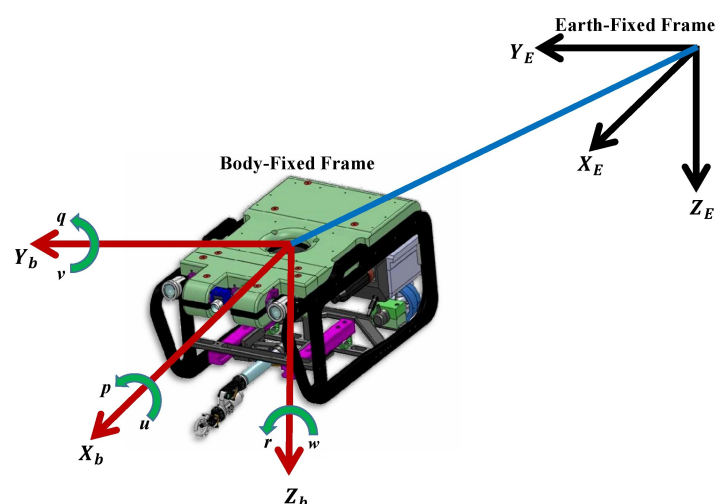


Figure 1. Frame coordinates of an underwater vehicle.

### 2.1. Kinematic Model

The general velocity vector is represented as:

$$\boldsymbol{v} = [\boldsymbol{v}_1 \ \boldsymbol{v}_2]^T = [u \ v \ w \ p \ q \ r]^T \quad (1)$$

where  $u, v$  and  $w$  are components of the linear velocity in surge, sway and heave directions, respectively, and  $p, q$  and  $r$  are components of the angular velocity in roll, pitch and yaw, respectively.

The position vector  $\boldsymbol{\eta}_1 \in \mathbb{R}^3$  and orientation vector  $\boldsymbol{\eta}_2 \in \mathbb{R}^3$  coordinates expressed in the Earth-fixed frame are:

$$\boldsymbol{\eta} = [\boldsymbol{\eta}_1 \ \boldsymbol{\eta}_2]^T = [x \ y \ z \ \phi \ \theta \ \psi]^T \quad (2)$$

where  $x, y$  and  $z$  represent the Cartesian position in the Earth-fixed frame and  $\phi$  represents the roll angle,  $\theta$  the pitch angle and  $\psi$  the yaw angle.

The relationship between velocities on the fixed and Equations are [27,28].

$$\begin{bmatrix} \dot{\boldsymbol{\eta}}_1 \\ \dot{\boldsymbol{\eta}}_2 \end{bmatrix} = \begin{bmatrix} \boldsymbol{J}_1(\boldsymbol{\eta}_2) & \boldsymbol{O}_{3 \times 3} \\ \boldsymbol{O}_{3 \times 3} & \boldsymbol{J}_2(\boldsymbol{\eta}_2) \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_1 \\ \boldsymbol{v}_2 \end{bmatrix} \quad (3)$$

where  $\boldsymbol{J}_1(\boldsymbol{\eta}_2) \in \mathbb{R}^{3 \times 3}$  is the rotation matrix which expresses the transformation from body to fixed frame, and  $\boldsymbol{J}_2(\boldsymbol{\eta}_2) \in \mathbb{R}^{3 \times 3}$  is another transformation matrix that relates the angular velocity  $\boldsymbol{v}_2 \in \mathbb{R}^3$  with the time derivative of  $\boldsymbol{\eta}_2 \in \mathbb{R}^3$ .

### 2.2. Hydrodynamic Model

Equations of motion expressed on the Equation [27],

$$\boldsymbol{M}\dot{\boldsymbol{v}} + \boldsymbol{C}(\boldsymbol{v})\boldsymbol{v} + \boldsymbol{D}(\boldsymbol{v})\boldsymbol{v} + \boldsymbol{G}\boldsymbol{\eta} = \boldsymbol{\tau} \quad (4)$$

$$\dot{\boldsymbol{\eta}} = \boldsymbol{J}(\boldsymbol{\eta})\boldsymbol{v}$$

where  $\boldsymbol{v} \in \mathbb{R}^n$  and  $\boldsymbol{\eta} \in \mathbb{R}^n$  were previously defined,  $\boldsymbol{M} \in \mathbb{R}^{n \times n}$  denotes the inertial matrix (including the added mass),  $\boldsymbol{C} \in \mathbb{R}^{n \times n}$  is the Coriolis matrix and centripetal forces (including the effects of added mass),  $\boldsymbol{D} \in \mathbb{R}^{n \times n}$  refers to the damping matrix,  $\boldsymbol{G} \in \mathbb{R}^n$  represents the vector of gravitational forces, and  $\boldsymbol{\tau} \in \mathbb{R}^n$  is the input control vector.

## 3. Ocean Currents

Ocean current is generated by wind, tides, variation of densities and re-circulation of water, among others. The main objective of this work is not to generate a detailed report of this phenomena; nevertheless, it is appropriate to highlight the model of induced ocean currents proposed by Fossen [27]. In the mentioned work, the equations of motion are represented in terms of relative velocity of the vehicle and the currents,

$$\boldsymbol{v}_r = \boldsymbol{v} - \boldsymbol{v}_{CI} \quad (5)$$

where  $\boldsymbol{v}_{CI} = [u_c \ v_c \ w_c \ 0 \ 0 \ 0]^T$  is a non-rotational vector of the current velocity according to Equation (3). Note that the linear velocity on the fixed frame can be transformed to linear velocity in the equation by applying the elemental rotation matrices. Let  $[u_c^E \ v_c^E \ w_c^E]$  be the current velocity referenced to the Earth-fixed frame. Then the components of the linear velocity on the equation are calculated as follows,

$$\begin{bmatrix} u_c \\ v_c \\ w_c \end{bmatrix} = J_1^T(\eta_2) \begin{bmatrix} u_c^E \\ v_c^E \\ w_c^E \end{bmatrix} \quad (6)$$

Suppose the current velocity in the equation as constant or at least with a minimum variation, so that:

$$\dot{v}_{CI} = 0 \rightarrow \dot{v}_r = \dot{v} \quad (7)$$

Then, the relative equations of motion become:

$$M\dot{v} + C(v_r)v_r + D(v_r)v_r + G(\eta) = \tau \quad (8)$$

Now, the current velocity in the Earth-fixed frame  $\begin{bmatrix} u_c^E & v_c^E & w_c^E \end{bmatrix}$  can be related to the mean velocity of the current  $V_C$  through two angles:  $\alpha$  (angle of attack),  $\beta$  (sideslip angle), describing the orientation of  $v_{CI}$  around the axes  $y$  and  $z$  respectively as follows:

$$\begin{aligned} u_c^E &= V_C \cos(\alpha) \cos(\beta) \\ v_c^E &= V_C \sin(\beta) \\ w_c^E &= V_C \sin(\alpha) \cos(\beta) \end{aligned} \quad (9)$$

where  $V_C$  is the average currents velocity in the earth-fixed reference frame.

#### 4. Self-Tuning Neural Network for PID Control

The tuning of PID (Proportional + Integral + Derivative) controllers depends on adjusting its parameters (i.e.,  $K_p$ ;  $K_i$ ;  $K_d$ ), so that the performance of the system under control becomes robust and accurate according to the established performance criteria. The proposed auto-tuning algorithm is based on NN which exhibit the following characteristics:

1. Parallelism and generalization. A NN are able to produce useful outputs for inputs not provided under the learning phase.
2. Non-linearity. A NN can be linear or not allowing it to represent systems generated by nonlinear guidelines.
3. Adaptability. NN are capable of re-adjusting weights and adapting to new environmental situations. This is specially useful when the system offers non-stationary data, that is, the properties involved by the system vary over time.
4. Fault tolerance. When an operational failure occurs on a local part of the network, it lightly affects the global performance.

This property is because of the distributive nature of stored data processed along the neural network.

Consistent with above, this work is based on a backpropagation neural network, which also meets the desired characteristics to accomplish the goal tasks. Recurrent networks with supervised learning structured with delay are widely used in underwater vehicles as mentioned in [17,18], as well as for linear systems with large uncertainties in their surrounding environment as shown in [5,29–31].

##### 4.1. Control Law

In the discrete time domain, the digital PID algorithm can be expressed as follows [17]:

$$\tau(n) = \tau(n-1) + K_p(e(n) - e(n-1)) + K_i e(n) + K_d(e(n) - 2e(n-1) + e(n-2)) \quad (10)$$

where  $\tau(n)$  is the original control signal,  $e(n) = \eta_d - \eta$  represents the position tracking error,  $\eta_d$  denotes the desired trajectory,  $K_p$  is the proportional gain,  $K_i$  the integral gain,  $K_d$  the derivative gain, and  $n$  the sample time.

A block diagram of the auto-tuning control with artificial neural network (NN) is shown in Figure 2.

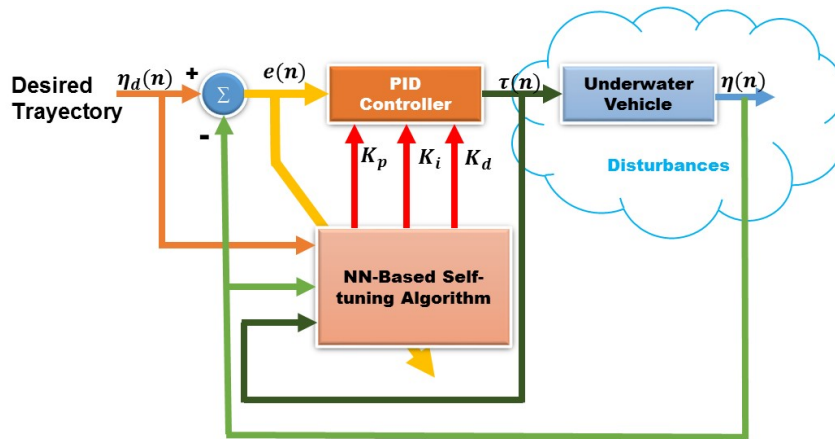


Figure 2. Block diagram of an auto-tuned PID with artificial NN control.

#### 4.2. Algorithm Auto-Tuner

The algorithm used as auto-tuning is the backpropagation method, chosen for its ability to adapt to changing environments. Operation begins applying the inputs to the network (see Figure 3), this is propagated from the first layer to the hidden layers in, up to produce an output ( $K_p, K_i$  and  $K_d$ ). The output signal is compared to the desired output and an error signal is calculated for each of the outputs, this is shown in Figure 2. The error outputs backpropagate, starting from the output layer, to all neurons in the hidden layer that contribute directly to the output; however, the hidden layer neurons receive only a fraction of the total error signal. This process repeats iteratively, layer by layer, until all neurons in the network has received an error signal describing its relative contribution to the total error.

Figure 3 presents the topology of the NN used to auto-tune the PID control gains implemented on the ROV. Its structure shows seven neurons on the input layer, three neurons on the hidden layer, and finally another three neurons on the output layer. The neurons placed on the output layer correspond to the PID gains:  $K_p, K_i, K_d$ .

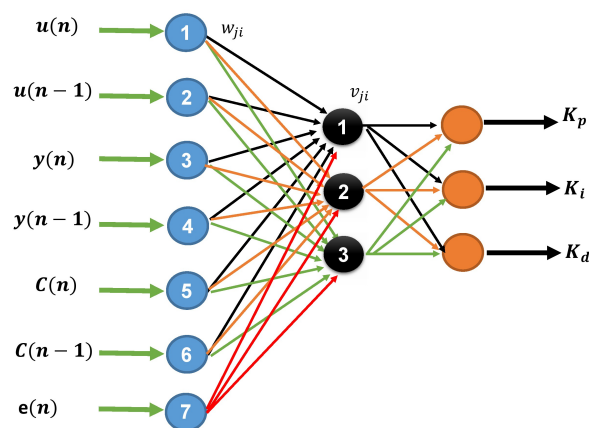


Figure 3. Block diagram of the implemented backpropagation NN.

where  $u(n)$  and  $u(n - 1)$  are reference inputs (desired trajectory),  $y(n)$  and  $y(n - 1)$  are reference outputs (real trajectory),  $C(n)$  and  $C(n - 1)$  correspond to the control signals,  $w_{ji}$  are the weights of the hidden layer, and  $v_{ji}$  are the weights of the output layer.

The back-propagation algorithm looks for the minimum of the error function in weight space using the method of gradient descent [3]. The combination of weights which minimizes the error function is considered to be a solution of the learning problem. The activation functions for back-propagation networks is the sigmoid, a real function  $s_c : \mathbb{R} \rightarrow (0, 1)$  defined by the expression

$$h_j = \frac{1}{1 + e^{-S_j}} \quad (11)$$

The output of the  $j$  hidden layer neuron may be calculated by means of:

$$S_j = \sum_{i=1}^3 w_{ji} x_i \quad (12)$$

The shape of the sigmoid changes according to the value of  $h_j$ . At the same time, the output layer neuron value will be:

$$u(n) = \frac{1}{1 + e^{-r}} \quad (13)$$

where:

$$r = \sum_{j=1}^3 v_j h_j \quad (14)$$

The criteria used to minimize the error correspond to Rojas et al. [32], as:

$$E(n) = \frac{1}{2} \sum_{k=1}^t e_y(k)^2 \quad (15)$$

where  $e_y(k) = y_r(n) - y(n)$ .

The minimization procedure consists, as it is known, in a movement in the negative gradient direction of the function  $E(n)$  with respect to the weighting coefficients  $v_{ji}$  and  $w_{ji}$ .

The  $E(n)$  gradient is a multi-dimensional vector [3] whose components are the partial derivatives  $\frac{\partial E(n)}{\partial v_{ji}}$ ,  $\frac{\partial E(n)}{\partial w_{ji}}$ .

The weighting coefficients of the input layer are

$$\frac{\partial E(n)}{\partial v_{ji}} = \delta^1 h_j \frac{\partial e_y}{\partial e_u} \quad (16)$$

The weighting coefficient of the hidden layer are

$$\frac{\partial E(n)}{\partial w_{ji}} = -\delta_j^2 x_i \frac{\partial e_y}{\partial e_u} \quad (17)$$

Using Equations (16) and (17), the adjustments of weighting coefficients  $v_{ji}$  (Equation (18)),  $w_{ji}$  (Equation (19)) can be made by means of the expressions:

$$v_{ji}(n+1) = v_{ji}(n) + (a \frac{\partial e_y}{\partial e_u}) \delta^1 h_j \quad (18)$$

$$w_{ji}(n+1) = w_{ji}(n) + (a \frac{\partial e_y}{\partial e_u}) \delta_j^2 x_j \quad (19)$$

where  $a$  is the learning coefficient,  $w_{ji}(n+1)$  is a vector of weights for the hidden layer,  $v_{ji}(n+1)$  is the vector of weights of the output layer and equivalent gain  $\frac{\partial e_y}{\partial e_u}$  is unknown.

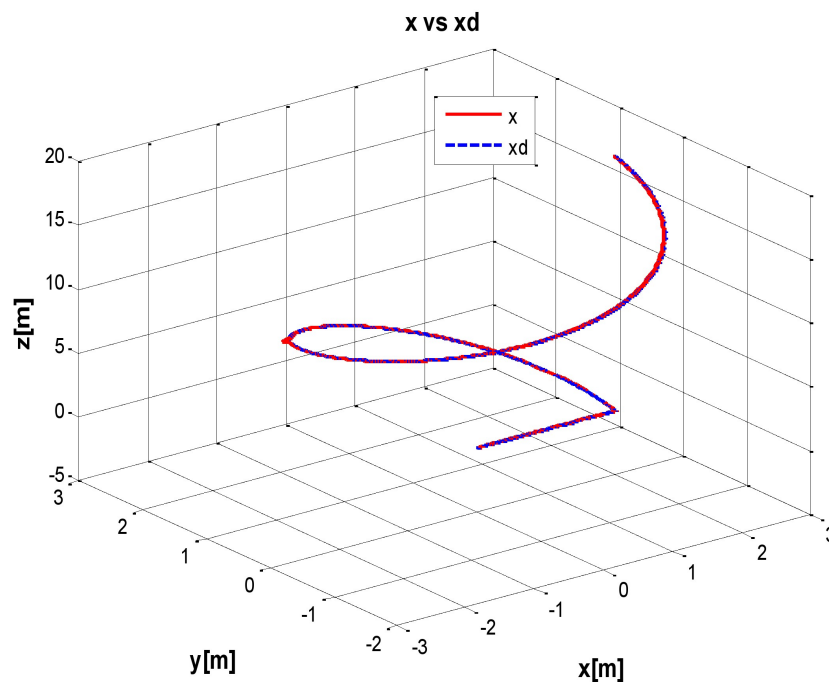
## 5. Simulation Results

The auto-tuned PID was evaluated using Matlab/Simulink software. The ODE 45 with a variable step was used, setting the maximum sample step as 0.01 s. The first proposed task consists of moving the robot in a straight line from its start position to a set point, letting  $x, y, z$  remain constant while  $\psi$  is varying. The next task is that the robot begins rising in a spiral motion, perturbed by water currents of considerable intensity. The first perturbation takes place in the first 20 s and its magnitude is  $V_c = 1.1$  m/s with orientation of  $\alpha = 0$  and  $\beta = 0$ . The second perturbation goes from time 20 to 45 s with a magnitude of  $V_c = 1.1$  m/s and an orientation of  $\alpha = 0$  and  $\beta = \pi/2$ , as set in Equation (9).

### 5.1. Underactuated 6 DOF ROV Kaxan

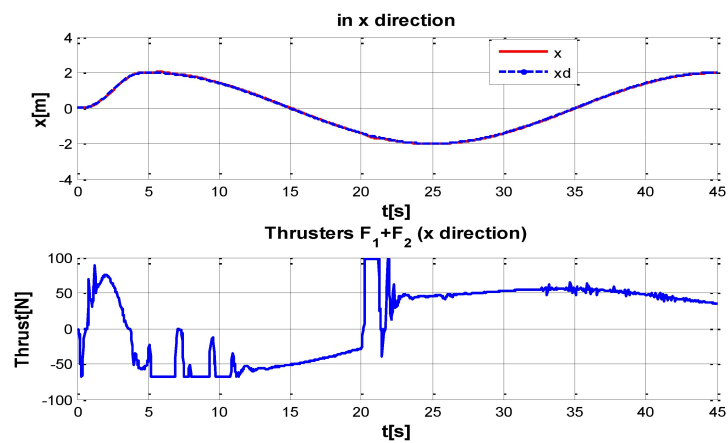
The Kaxan robot hydrodynamic parameters are included [33].

The behaviour can be observed in Figures 4–8. Figure 4 depicts the trajectory in 3D that the Kaxan robot follows.

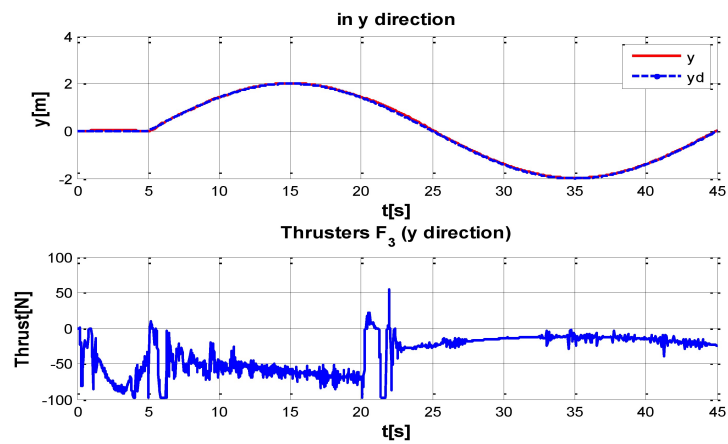


**Figure 4.** Real trajectory ( $X$  Solid line) vs. desired trajectory ( $X_d$  dotted line), with perturbation.

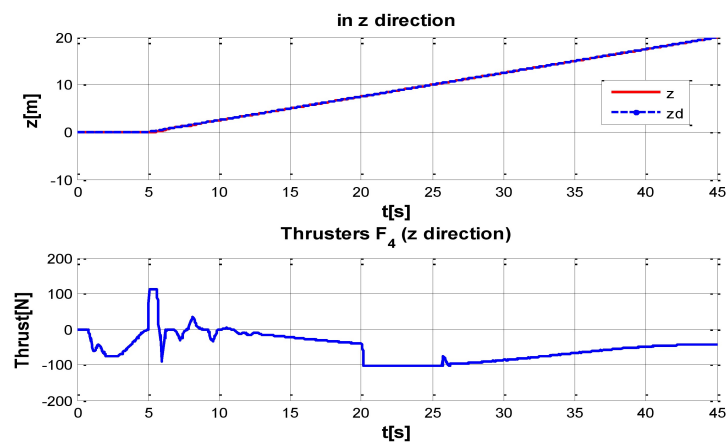
Figures 5–8, present the interaction of the neural network by modifying the gains from the beginning through time until arriving to a steady state. While the second perturbation is introduced (change of ocean current direction at time 20), neurons detect changes and perform compensation by increasing or decreasing as appropriate,  $K_p, K_i, K_d$  gains.



**Figure 5.** Behavior of the system in x-coordinate with perturbations up and (down) control signal of thruster  $F_1 + F_2$ .

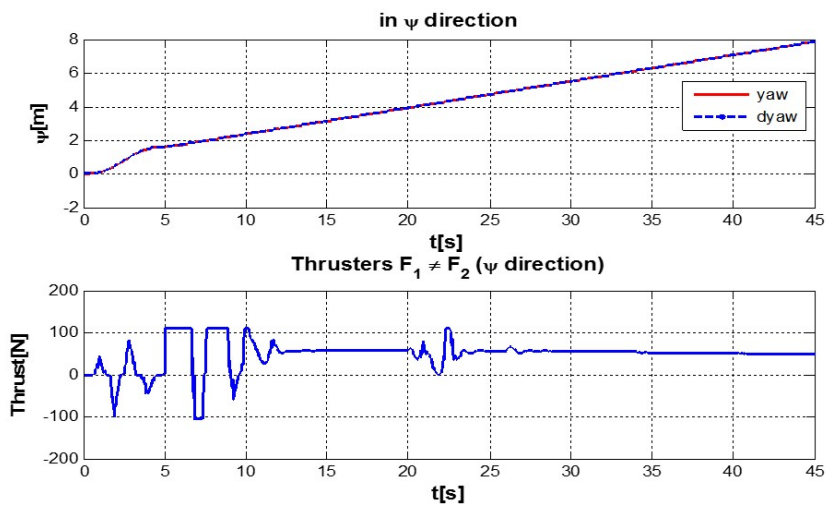


**Figure 6.** Behavior of the system in y-coordinate with perturbations up and (down) control signal of thruster  $F_3$ .



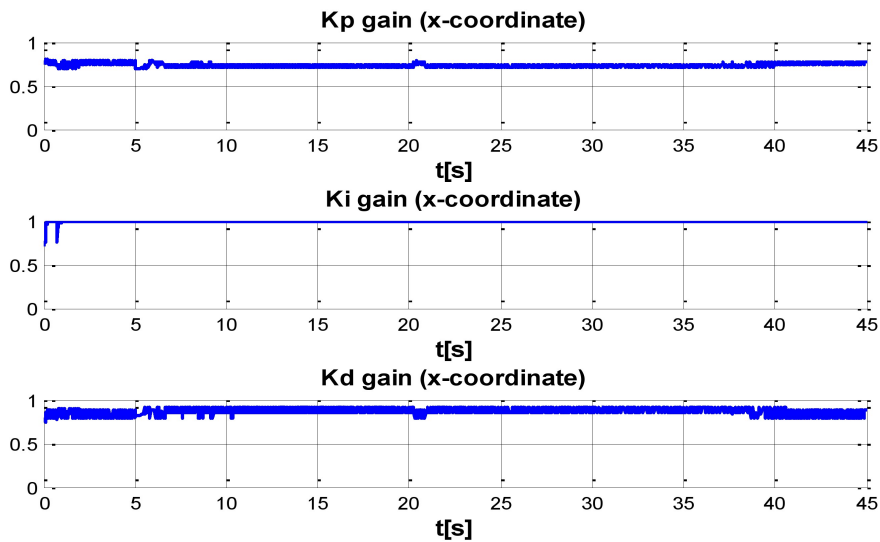
**Figure 7.** Behavior of the system in z-coordinate with perturbations up and (down) control signal of thruster  $F_4$ .





**Figure 8.** Behavior of the system in  $\psi$ -coordinate with perturbations up and (down) control signal of thruster  $F_1 \neq F_2$ .

The next set of Figures 9–12 indicate the gains ( $K_p, K_i, K_d$ ) obtained by the neural network in every DOF,  $\eta = [x; y; z; \psi]$ . As can be seen, neurons start working from time zero since they absorbed the first perturbation of 20 s in length, and respond to the abrupt change presented when the second perturbation is introduced, thus allowing change of the neural network in order to compensate for the lack of gains in the DOF corresponding to the alterations.



**Figure 9.** PID gain time behavior in x-coordinate.

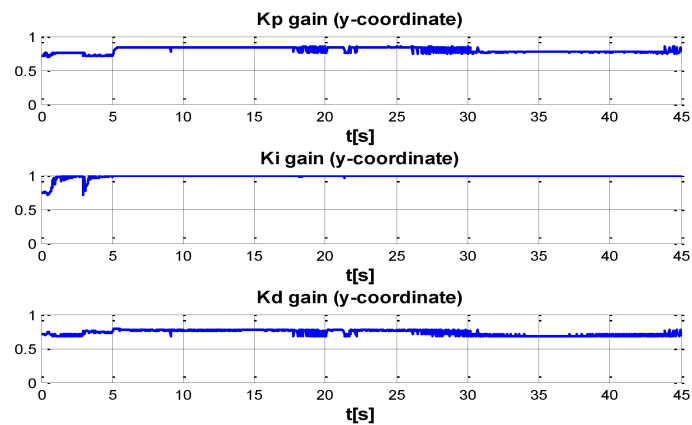


Figure 10. PID gain time behavior in y-coordinate.

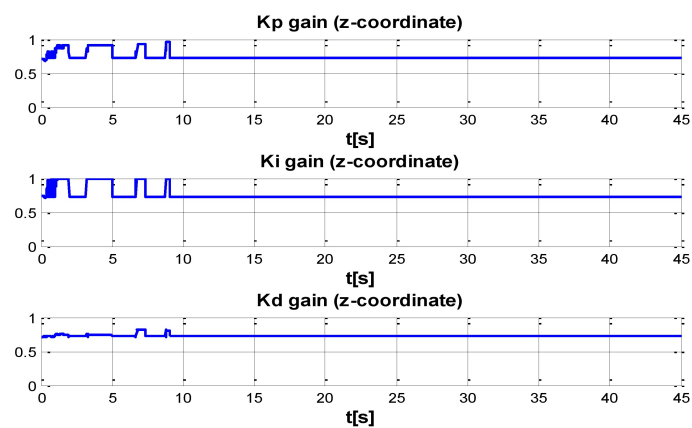


Figure 11. PID gain time behavior in z-coordinate.

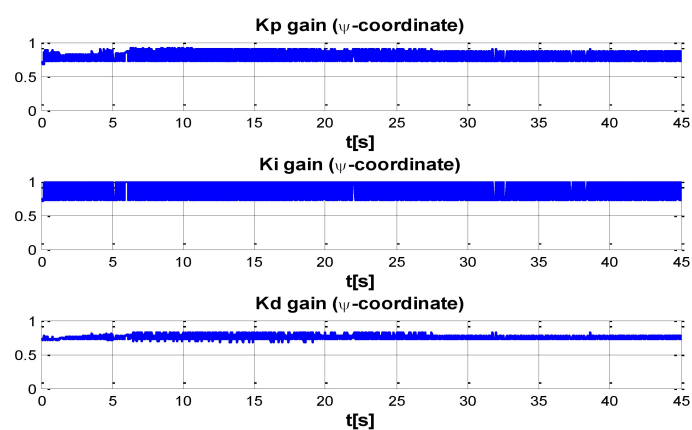


Figure 12. PID gain time behavior in  $\psi$ -coordinate.

## 5.2. PID vs. Auto-Tuned PID

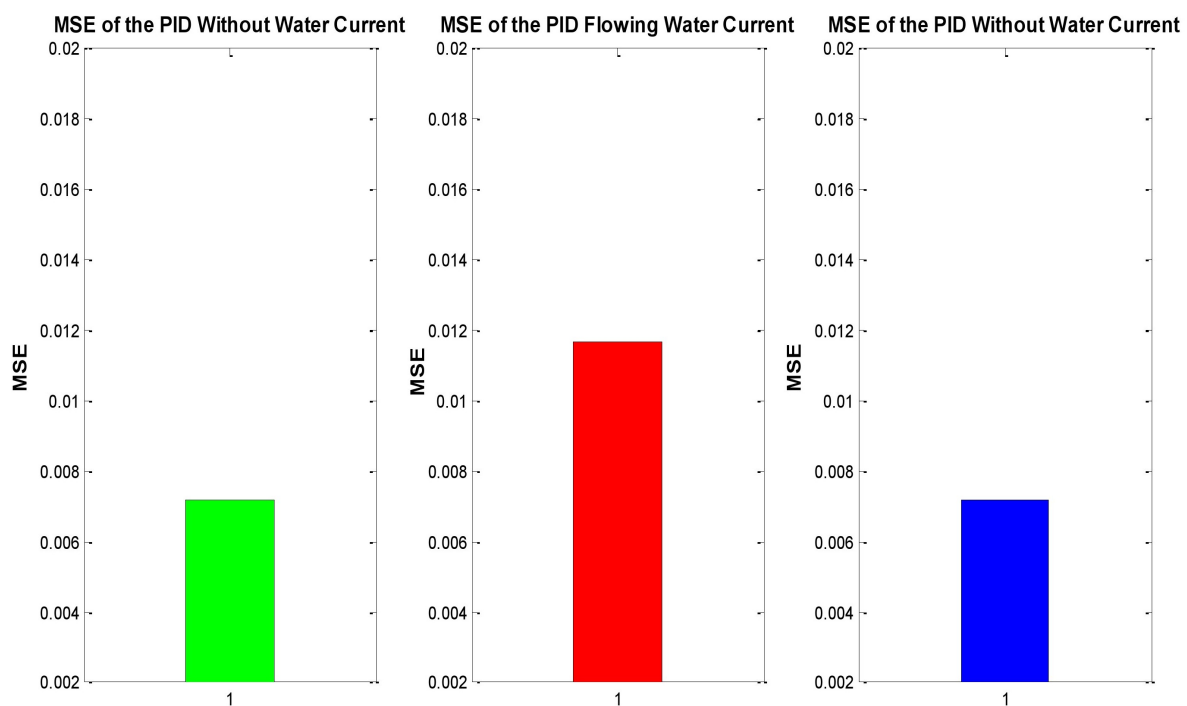
In order to compare the conventional PID vs. the auto-tuned PID, a statistical indicator was implemented, allowing to determine which one has the best behavior following a trajectory. The mean

square error (MSE) lets us estimate the performance of every control by analyzing the error generated in the trajectory tracking.

$$MSE_R = \sqrt{MSE_X + MSE_Y + MSE_Z + MSE_\psi} \quad (20)$$

where  $MSE_x$  is the MSE in  $x$ ,  $MSE_y$  is the MSE in  $y$ , and so on.

As mentioned previously, MSE was used to evaluate the tracking performance. Figure 13 shows the evaluation of the experiment mentioned above, which considers the experiment under initial conditions (without water currents) in a 0 s to 45 s time frame in which, afterwards the ocean current appears from time 45 s to 90 s. Finally, at time 90 s the ocean current stops, with discrete time for the PID control being implemented throughout. Figure 14 demonstrates the same experiment with the auto-tuned PID control.



**Figure 13.** MSE of the conventional PID Controller position tracking trajectory.

Figure 13 describes the increase of MSE of the conventional PID when perturbation occurs (red bar), while green and blue bars, corresponding to the absence of perturbations, remain steady. Moreover, the MSE of the auto-tuned PID [14] is around a 50% less than the one of the conventional PID, due to the self-tuning algorithm with a NN. Additionally, when perturbation happens, the increase and decrease of the auto-tune MSE is minimum. For this reasons, it is feasible to conclude that the auto-tuned PID has better performance facing changes on the hydrodynamic parameters and perturbations of the surrounding environment.

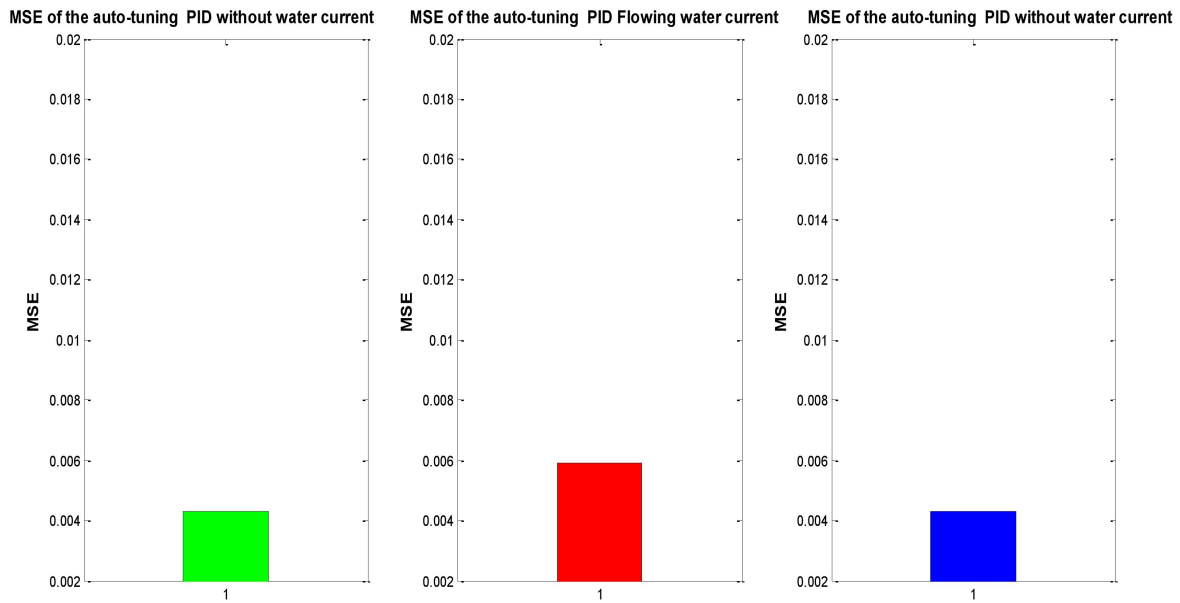


Figure 14. MSE of the auto-tuned PID position tracking.

## 6. Experimental Set up

In this section, the experimental set up as well as the results are discussed. Two sets of experiments are presented, one considering the conventional PID controller and the other one considering the auto-tuned PID proposed in this paper. Both controllers were tested under the same conditions in order to evaluate their performance under disturbances. A comparative analysis in terms of position tracking and energy consumption is given.

The proposed intelligent control was implemented on an underactuated mini-ROV. This vehicle is a ROV developed in CIDESI named Nu'ukul Ja (which in the Mayan language means "water instrument"). Its dimensions are: 50 cm long, 30 cm wide, and 30 cm height; as shown in Figure 15. It has a cylindrical pressure chamber of 15 cm diameter where the major part of the electronic architecture is placed. The total weight of the ROV is 10 kg. According to the experimental environment, it was placed in a pool of 2.5 m where both PID controllers (auto-tuned and conventional) were implemented.

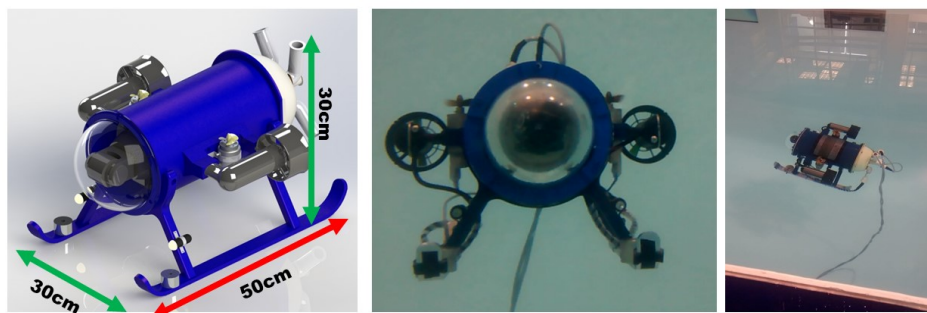


Figure 15. Underactuated mini-ROV under water.

The electronic architecture of the ROV (Figure 16) consist on three groups: instrumentation, signal and data acquisition, and actuators. The instrumentation involves: pressure sensor, leakage sensors, AHRS (Attitude and Heading Reference System), voltage and current sensors. While the signal and data acquisition implicates a micro controller embedded in a development board. Finally, the actuators

consist on 4 thrusters used to provide direction and displacement to the vehicle, and an IP camera for inspection missions.

The ROV is connected to the surface by navel string of thirteen wires. Where eight of them are used to receive video from the IP camera, three for the power connections ( $12 V_{DC}$ ,  $20 V_{DC}$  and Ground), and two for data UART transmission (TX)/reception (RX).

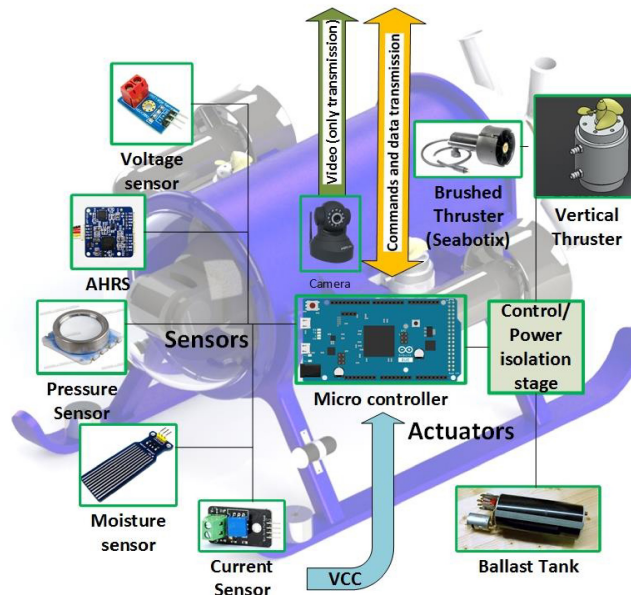


Figure 16. Electronic Architecture of the mini-ROV.

### 6.1. Instrumentation

This ROV has a MS5803 – 05BA pressure sensor which is placed outside the chamber of the submarine, Figure 17. The sensor is a high resolution barometer which obtain data of the surrounding hydrostatic pressure, acquiring frequencies up to 50 kHz by I2C protocol. Once the hydrostatic pressure is obtained the depth level is calculated by:  $h = \frac{P - P_0}{\rho g}$ , where  $h$  = depth (m),  $P$  = hydrostatic pressure (bar),  $P_0$  = atmospheric pressure (bar),  $\rho$  = water density ( $\text{kg}/\text{m}^3$ ).

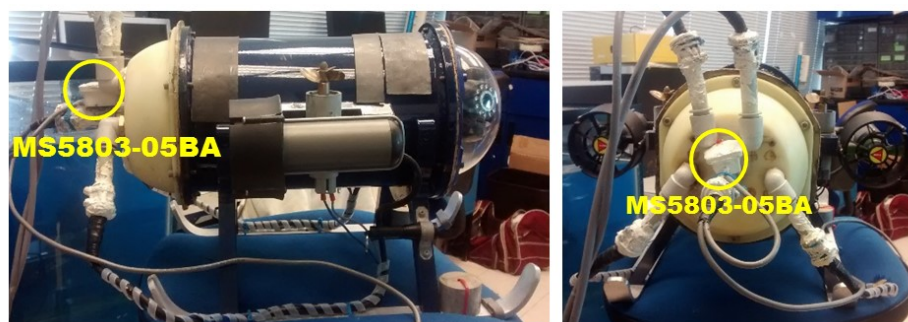


Figure 17. Location Sensor MS5803-14BA.

In order to sense 3 DOF's of the ROV (pitch, yaw and roll), the AHRS UM7 is a CHRobotics device is used. The AHRS sends NMEA serial packages with a frequency up to 100 Hz. To prevent malfunction of the electronics due to water presence, printed electrodes connected to the controller represent the leakage sensors. Also, 4282 voltage sensor (5 to  $1 V_{DC}$  divider) offers an analogical signal of the batteries voltage. Pololu's AC715, a Hall Effect current sensor, allows to monitor the operation of the thrusters.

### 6.2. Signal and Data Acquisition

The ROV has a SAM3X8E ARM Cortex embedded in the Arduino Due's developing board. It has 54 general purpose inputs and outputs, 12 of them PWMs, 12 analog inputs, 4 UART ports and one I2C bus. This controller is used to manage communication between the user and the mini-ROV.

### 6.3. Actuators

As mentioned above, to displace and lead the submarine in one plane, two brushed *SeaBotix* BTD150 thrusters are placed horizontally on each side of the underwater robot. These thrusters are powered by  $20 V_{DC}@4 A$ . To dive the submarine, 2 brushed thrusters were placed vertically on each side of the ROV (beside the lateral thrusters, Figure 18); these are basically modified fuel pumps with 4 cm propellers attached to their shaft. Each brushed motor consume  $12 V_{DC}@1.5 A$ .

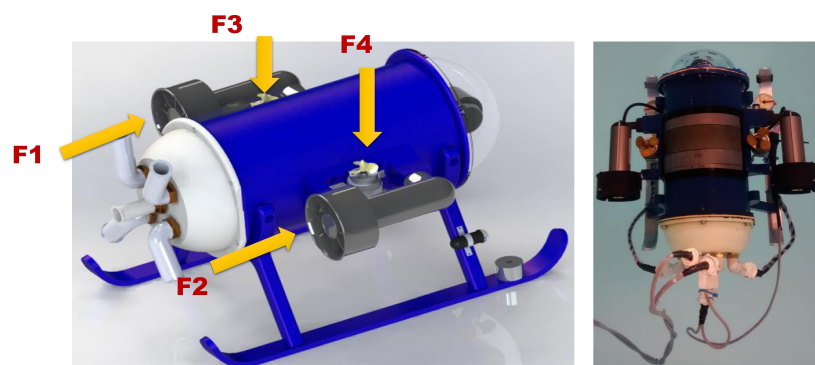


Figure 18. Thruster location.

### 6.4. Results

The controls were evaluated by performing a data capture of 3 m, once the ROV was placed 1m underwater. In the first minute non disturbance took place. After this time, the weight was increased by 400 g (Figure 19) until a two minute mark.

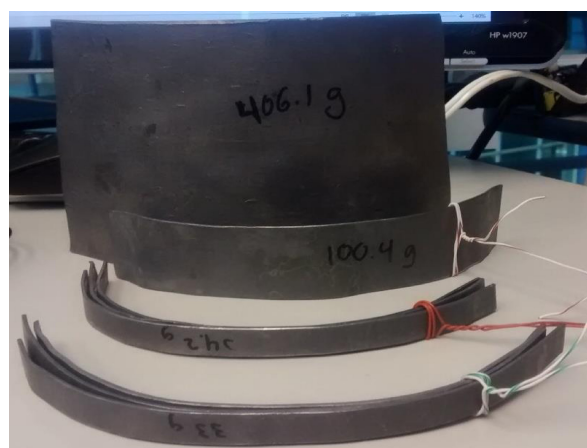


Figure 19. Weight (disturbance).

The gains of the conventional PID control were obtained by means of the NN. The ROV was requested to get the set point of 1 m depth by using the Auto-tuned PID controller. Once the ROV reached the stability and the PID gains, computed by the NN, became stationary, these gains were programmed into the conventional PID as  $K_p$ ,  $K_d$ , and  $K_i$ . This the way the conventional PID gains were tuned. It is important to remark that once the conventional PID was tuned, the gains

remained constant along the experiment, even when the disturbances took place, unlike the Auto-tuned PID controller wherein gains were dynamically changing to attain the better performance along the experiment.

Finally, in the last minute of data capture the weight was removed. Figure 20 shows the desired trajectory (Solid line) vs. the real path (dotted line), also the control signals for thrusters F1 and F2 are displayed.

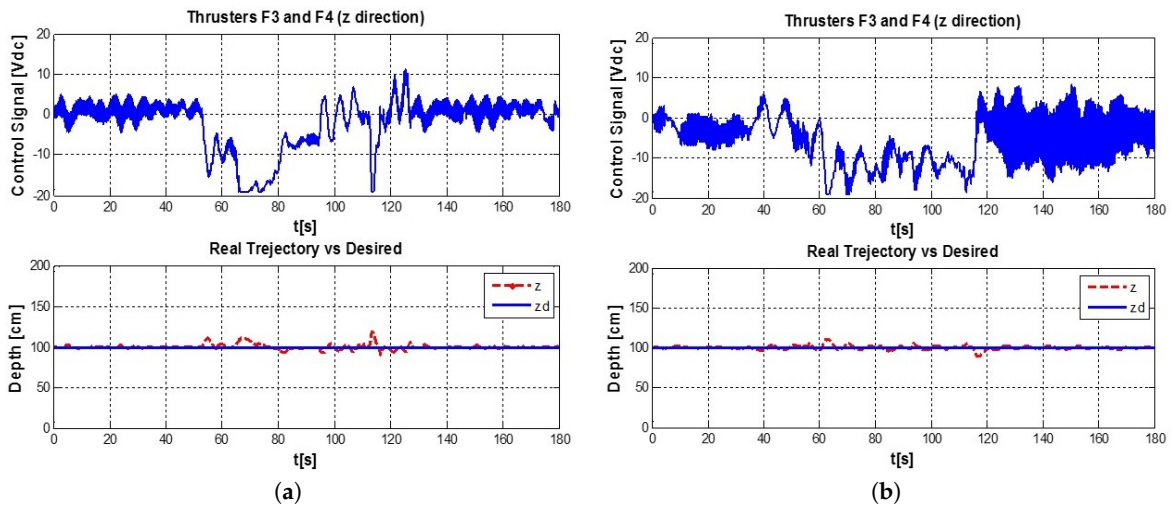


Figure 20. (a) Conventional PID Controller, (b) auto-tuned PID.

Apparently, the control signal given by the auto-tuned PID (shown in Figure 20), seems to be more active than the conventional PID’s signal; though, the root mean square (RMS) value of each one (the complete experiment), shows that the RMS of the conventional PID is 6.8874 whereas in the auto-tuned PID is 6.6781. The auto-tuned PID has a 3.038% Of energy saving against the conventional PID.

MSE offers a better notion of the results, leading to the conclusion that the neuronal PID is better than PID fixed gains, as can be seen in the Figure 21.

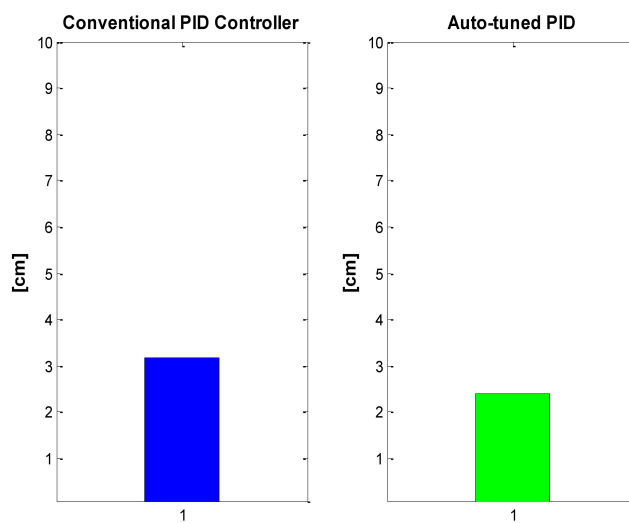


Figure 21. MSE: Conventional PID Controller (left) vs. auto-tuned PID (right).

In order to determine where the neural PID has a better performance, the test was divided in three phases corresponding to: where no disturbance took place, when the disturbance is added and again

when the disturbance stopped, as can be seen in Figure 22. Once again, to compare the auto-tuned PID vs. the conventional PID controller with fixed gains, the MSE of every phase was obtained and it is shown in Figure 23.

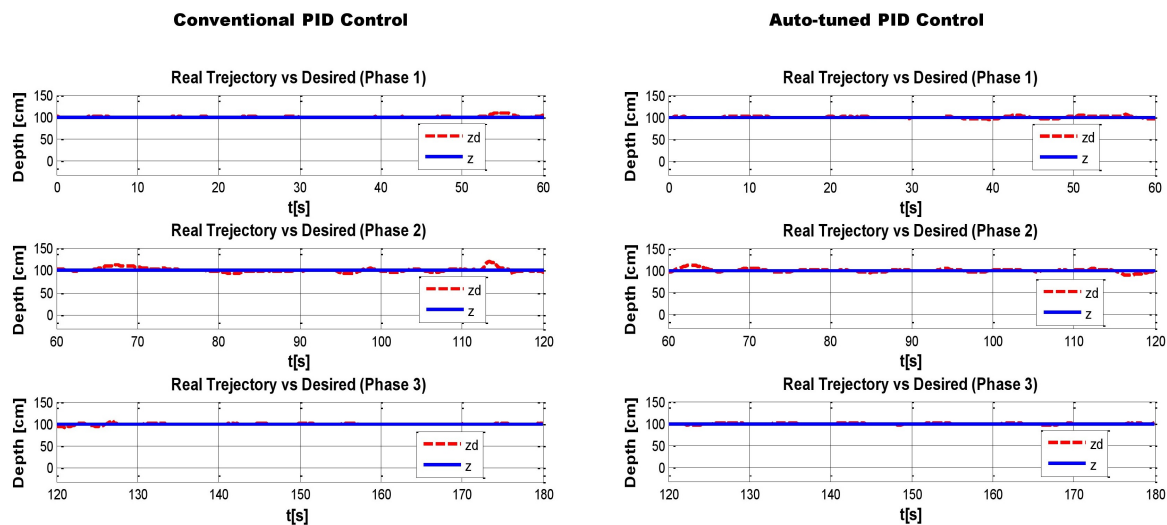


Figure 22. Conventional PID Controller vs auto-tune PID, segmented in 3 phases of 1 m each.

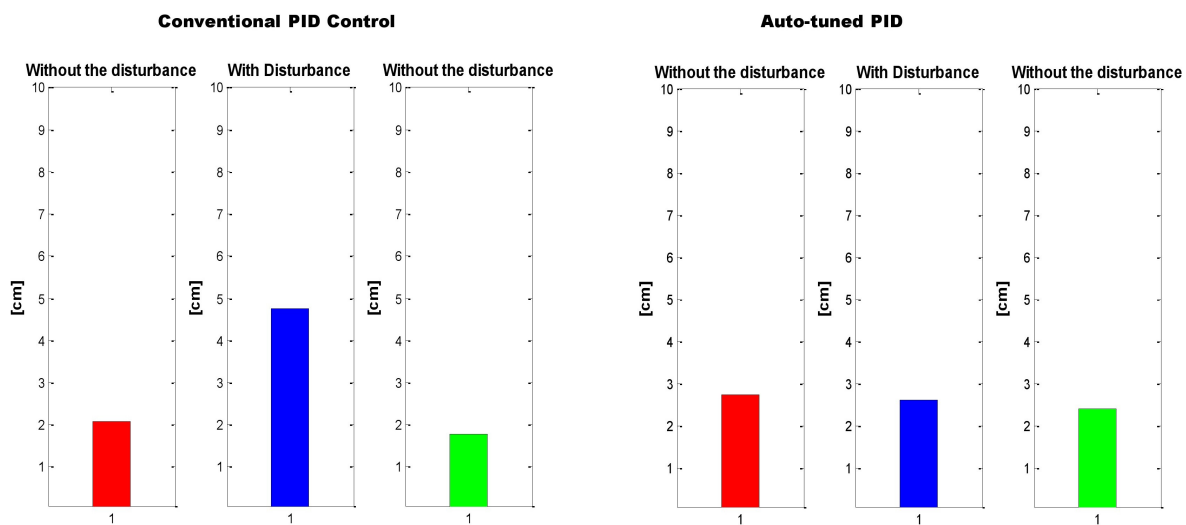


Figure 23. MSE: Conventional PID Controller (left) vs. auto-tune PID (right).

## 7. Conclusions

The actual work presents the development of a control algorithm to automatically tune the gains of a PID control, based on a neural network. The control algorithm was implemented on ROVs for trajectory tracking with unknown disturbances. The algorithm performance was evaluated in two instances: a numerical simulation and implemented on a ROV in real-time. The numerical simulation took place with the non-linear hydrodynamics of ROV Kaxan with 4 of the 6 DOF actuated; including disturbances of ocean currents in different directions. In reference of the second validation, it was implemented in a mini-ROV for the depth DOF, in order to validate in real-time the auto-tuned PID control. A comparative study between the conventional PID and the auto-tuned PID (proposed here) was discussed. The study took into consideration two criterions to assess the performance of each controller: position tracking error and energy consumption, leading to the conclusion that the proposed controller attained the best performance with less energy.



**Acknowledgments:** Research supported by CONACYT projects: PDCPN2013-01-215770 and No. 201441.

**Author Contributions:** Rodrigo Hernández-Alvarado, Luis García-Valdovinos, Tomas Salgado-Jiménez conceived and designed the control scheme. Rodrigo hernandez-Alvarado, Fernando Fonseca-Navarro design and cored out experiment, Alfonso Gómez-Espinosa contributed with the synthesis of the neuronal networks and the analysis of experiment results.

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

## References

1. Kuo, T.; Huang, Y.; Chen, C.; Chang, C. Adaptive Sliding Mode Control with PID Tuning for Uncertain Systems. *Eng. Lett.* **2008**, *16*, 311–315.
2. Karakuzu, C. Parameter Tuning of Fuzzy Sliding Mode Controller Using Particle Swarm Optimization. *Int. J. Innov. Comput. Inform. Control* **2010**, *6*, 4755–4770.
3. Ponce, A.N.; Behar, A.; Hernández, A.; Sitar, V. Neural Networks for Self-Tuning Control Systems. *Acta Polytech.* **2004**, *44*, 49–52.
4. Terán-Picón, J.; Camacho, O. Genetics Algorithms as a Tuning Tool for PID and Sliding Mode Controllers. *Rev. Téc. Fac. Ing Univ.* **2005**, *28*, 200–209.
5. Suárez, J.G.M.; Torales, G.G.; Quitarte, R.X.M.; Ortega, H.M.; González, E.F.R. Sintonización del Controlador PID con un Elemento Lineal Adaptivo, Adaline. In Proceedings of the Instrumentation Conference SOMI XX, Leon, Mexico, 24–28 October 2006.
6. Song, X.; Liu, F.; Zou, Z.; Zhu, Y.M.; Yin, J.; Xu, F. Nonlinear Underwater Robot Controller Design With Adaptive Disturbance Prediction and Smoother. *Int. J. Comput. Intell. Syst.* **2011**, *4*, 634–643.
7. Marzbanrad, A.; Eghtesad, M.; Kamali, R. A Robust Adaptive Fuzzy Sliding Mode Controller For Trajectory Tracking of ROVs. In Proceedings of the 50th IEEE Conference on Decision and Control and European Control Conference, Orlando, FL, USA, 12–15 December 2011; pp. 2863–2870.
8. Wu, B.; Li, S.; Wang, X. Discrete-Time Adaptive Sliding Mode Control of Autonomous Underwater Vehicle in the Dive Plane. In *Intelligent Robotics and Applications*; Springer: Heidelberg, Germany, 16–18 December 2009; pp. 157–164.
9. Mannan, M.A.; Murata, T.; Tamura, J.; Tsuchiya, T. A Fuzzy-Logic-Based Self-Tuning PI Controller for High-Performance Vector Controlled Induction Motor Drive. *Electric Power Compon. Syst.* **2006**, *34*, 471–481.
10. Kha, N.; Ahn, K.K. Position Control of Shape Memory Alloy Actuators by Using Self Tuning Fuzzy PID Controller. In Proceedings of the 1st IEEE Conference on Industrial Electronics and Applications, Ulsan, Korea, 24–26 May 2006; pp. 1–5.
11. Mudi, R.; Pal, N. A robust self-tuning scheme for PI- and PD-type fuzzy controllers. *IEEE Trans. Fuzzy Syst.* **1999**, *7*, 2–16.
12. Loc, M.B.; Choi, H. Design of self-tuning gain depth controller for an autonomous underwater vehicle with mass shifter mechanism. In Proceedings of the 12th International Conference on Control, Automation and Systems, Jeju Island, South Korea, 17–21 October 2012; pp. 1742–1746.
13. Paulusova, J.; Orlicky, L.; Dubravská, M. Self-tuning fuzzy PID controller. In Proceedings of the International Conference on Process Control, Štrbské Pleso, Slovakia, June 2013; pp. 304–308.
14. Amer, A.; Sallam, E.; Elawady, W. Fuzzy pre-compensated fuzzy self-tuning fuzzy PID controller of 3 DOF planar robot manipulators. In the Proceedings of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics, Montreal, QC, Canada, 6–9 July 2010; pp. 599–604.
15. Shimmin, D.; Stephens, M.; Swainston, J. Adaptive control of a submerged vehicle with sliding fuzzy relations. *Fuzzy Sets Syst.* **1996**, *79*, 15–24.
16. Hung, L.; Chung, H.Y. Auto-Tuning Neuron with Sliding Mode Control for robotic manipulator. In Proceedings of the 25th National Symposium on Automatic Control, Tainan, Taiwan, March 2004.
17. Kawafuku, M.; Sasaki, M.; Kato, S. Self-Tuning PID Control of a Flexible Micro-Actuator Using Neural Networks'. In the Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, San Diego, CA, USA, 11–14 October 1998; pp. 3067–3072.
18. Dong, E.; Guo, S.; Lin, X.; Li, X.; Wang, Y. A Neural Network-Based Self-Tuning PID Controller of an Autonomous Underwater Vehicle. In the Proceedings of the International Conference on Mechatronics and Automation, Chengdu, China, 5–8 August 2012; pp. 898–903.

19. Cordova, H.; Wijaya, A.F. Self-Tuning PID Neural Network Controller to Control Nonlinear Ph Neutralization in Waste Water Treatment. *IPTEK J. Technol. Sci.* **2007**, *18*, 89–95.
20. Sato, T.; Inoue, A.; Hirashima, Y. Self-tuning two-degree-of-freedom PID controller reducing the effect of disturbances. In Proceedings of the American Control Conference, Anchorage, AK, USA, 8–10 May 2002; pp. 3997–4002.
21. Tan, K.; Huang, S.; Ferdous, R. Robust self-tuning PID controller for nonlinear systems. *J. Process Control* **2001**, *1*, 758–763.
22. Xue, S.; Fan, H.; Xu, H. Study on ann-based intelligent self-tuning PID controller. In Proceedings of the International Conference on Mechatronics and Automation, Changchun, China, 9–12 August 2009; pp. 524–529.
23. Luoren, L.; Jinling, L. Research of PID Control Algorithm Based on Neural Network. *Energ. Proced.* **2011**, *13*, 6988–6993.
24. Sinthipsomboon, K.; Hunsacharoonroj, I.; Khedari, J.; Pongaen, W.; Pratumswan, P. A hybrid of fuzzy and fuzzy self-tuning PID controller for servo electro-hydraulic system. In Proceedings of the 6th IEEE Conference on Industrial Electronics and Applications, Beijing, China, 21–23 June 2011; pp. 220–225.
25. Ai, T.; Yu, J.; Liu, Y.; Zhou, J. Study on Neural Network Self-Tuning PID Control for Temperature of Active Solar House Heating System. In Proceedings of the International Workshop on Intelligent Systems and Applications, Wuhan, China, 22–23 May 2010; pp. 1–4.
26. Sasaki, M.; Asai, A.; Shimizu, T.; Ito, S. Self-tuning control of a two-link flexible manipulator using neural networks. In Proceedings of the ICCAS-SICE, Fukuoka, Japan, 18–21 August 2009; pp. 2468–2473.
27. Fossen, T.I. *Marine Control Systems: Guidance, Navigation and Control of Ships, Rigs and Underwater Vehicles*; Marine Cybernetics Trondheim: Trondheim, Norway, 2002.
28. Yuh, J. Modeling and control of underwater robotic vehicles. *IEEE Trans. Syst. Man Cybern.* **1990**, *20*, 1475–1483.
29. Aggarwal, V.; Mao, M.; O'Reilly, U. A Self-Tuning Analog Proportional-Integral-Derivative (PID) Controller. In Proceedings of the First NASA/ESA Conference on Adaptive Hardware and Systems, Istanbul, Turquia, 15–18 June 2006; pp. 12–19.
30. Kim, J.S.; Kim, J.H.; Park, J.M.; Park, S.M.; Choe, W.Y.; Heo, H. Auto Tuning PID Controller based on Improved Genetic Algorithm for Reverse Osmosis Plant. *Int. J. Intell. Technol.* **2008**, *3*, 232–237.
31. Hendookolaei, A.; Ahmadi, N.B. PID Controller with Neural Auto Tuner Applied in Drum Type Boilers. *Canadian J. Electr. Electron. Eng.* **2012**, *3*, doi:10.1016/j.protcy.2016.05.164.
32. Rojas, R. *Neural Networks: A Systematic Introduction*; Springer Science & Business Media: Heidelberg, Germany, 2013.
33. García-Valdovinos, L.G.; Salgado-Jiménez, T.; Bandala-Sánchez, M.; Nava-Balanzar, L.; Hernández-Alvarado, R.; Cruz-Ledesma, J.A. Modelling, Design and Robust Control of a Remotely Operated Underwater Vehicle. *Int. J. Adv. Robot Syst.* **2014**, *11*, doi:10.5772/56810.



© 2016 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 (<http://creativecommons.org/licenses/by/4.0/>).