sensors

Article

Visual Servoing for an Autonomous Hexarotor Using a Neural Network Based PID Controller

Carlos Lopez-Franco [1,2,*], Javier Gomez-Avila [1], Alma Y. Alanis [1], Nancy Arana-Daniel [1] and Carlos Villaseñor [1]

1 Centro Universitario de Ciencias Exactas e Ingenierías, Universidad de Guadalajara, Blvd. Marcelino García Barragán 1421, Guadalajara C.P. 44430, Jalisco, Mexico; javier.ega@hotmail.com (J.G.-A.); alma.alanis@cucei.udg.mx (A.Y.A.); nancy.arana@cucei.udg.mx (N.A.-D.); cavp@outlook.com (C.V.)

2 Avenida Revolución 1500 Modulo "R", Colonia Universitaria, Guadalajara C.P. 44430, Jalisco, Mexico

* Correspondence: carlos.lopez@cucei.udg.mx; Tel.: +52-(33)-1378-5900

Received: 1 July 2017; Accepted: 10 August 2017; Published: 12 August 2017

Abstract: In recent years, unmanned aerial vehicles (UAVs) have gained significant attention. However, we face two major drawbacks when working with UAVs: high nonlinearities and unknown position in 3D space since it is not provided with on-board sensors that can measure its position with respect to a global coordinate system. In this paper, we present a real-time implementation of a servo control, integrating vision sensors, with a neural proportional integral derivative (PID), in order to develop an hexarotor image based visual servo control (IBVS) that knows the position of the robot by using a velocity vector as a reference to control the hexarotor position. This integration requires a tight coordination between control algorithms, models of the system to be controlled, sensors, hardware and software platforms and well-defined interfaces, to allow the real-time implementation, as well as the design of different processing stages with their respective communication architecture. All of these issues and others provoke the idea that real-time implementations can be considered as a difficult task. For the purpose of showing the effectiveness of the sensor integration and control algorithm to address these issues on a high nonlinear system with noisy sensors as cameras, experiments were performed on the Asctec Firefly on-board computer, including both simulation and experimenta results.

Keywords: unmanned aerial vehicle; hexarotor; visual servoing

1. Introduction

The use of Unmanned Aerial Vehicles (UAVs) has been increased over the last few decades. UAVs have shown satisfactory flight and navigation capabilities, which are very important in applications like surveillance, mapping, search and rescue, etc. The ability to move freely in a 3D space represents a great advantage over ground vehicles, especially when the robot is supposed to travel long distances or move in dangerous environments, like in search and rescue tasks. Commonly, UAVs have four rotors; however, having more than four gives them a higher lifting capacity. The hexarotor has some advantages over the highly popular quadrotor, such as their increased load capacity, higher speed and safety, because the two extra rotors allow the UAV landing even if it loses one of the motors. However, the hexarotor is a highly nonlinear and under actuated system because it has fewer control inputs than degrees of freedom and its Lagrangian dynamics contain feedforward nonlinearities; in other words, there are some acceleration directions that can only be produced by a combination of the actuators.

In contrast with ground vehicles, it is not possible to use sensors like encoders to estimate its position. A good alternative is to use visual information as a reference, due to the high amount of information that a camera can provide in contrast with their low power consumption and low weight. Since it is not possible to know the position of a hexarotor with common on-board sensors such as Inertial Measurement Units (IMU), some works use off board sensor systems [1–5]; however, this kind of control limits the application to indoor navigation and adds noise and delays because of the communication between the robot and the ground station.

For such reason, visual control of UAVs has been widely performed. Although stereo vision is extensively used in mapping applications [6,7], when used in UAV navigation like in [8,9], it requires 3D reconstruction or optical flow, which are computationally expensive algorithms. In this approach, monocular vision is used, and the feature error position in the image coordinate plane is related with the robot velocity vector that reduces this error [10–15]. Consequently, we can set the position of the robot based on the camera information to control its navigation not only in indoor environments [16,17]. Classical Image Based Visual Servo (IBVS) control stabilizes attitude and position separately [18], which is not possible for underactuated systems. In [18], an Image Based approach is used for an underactuated system but approximating the depth distance to the features.

In [19], a PID controller is implemented on an hexarotor and comparisons between quaternions and Euler angles are made. In [20], the authors propose a visual servoing algorithm combined with a proportional derivative (PD) controller. However, PID approaches are not effective on highly nonlinear systems with model uncertainties such as the hexarotor [21,22]. According to this, another approach is required. In this paper, we propose the use of a Neural Network based PID. The advantages of using neural networks to control nonlinear systems are that the controller will have the adaptability and learning capabilities of the neural network [23], making the system able to adapt to actuator faults such as loss of effectiveness, which is described in [24] and solves disadvantages of the traditional PID [25] such as uncertainties of the system, communication time-delay, parametric uncertainties, external disturbances, actuator saturations and unmodeled system dynamics, among others. If to all of these issues we add the complexity to integrate servo control algorithms with vision sensors and a neural PID in a real-time implementation, it is required to have a well-designed coordination between all of the elements of this implementation, requiring different processing stages with their respective communication architecture (software and hardware).

The rest of the paper is structured as follows: Section 2 describes the robot and its dynamics. In Section 3, the visual servo control approach is introduced. In Section 5, the design of the PID controller and weights adjustment are shown. Section 4 presents the relationship between the error signals from the visual algorithm and the control signals of the hexarotor. Sections 6 and 7 present the simulation and experimental results of the proposed approach and its comparison with the conventional PID controller. Finally, the conclusions are given in Section 8.

2. Hexarotor Dynamic Modeling

The hexarotor consists of six arms connected symmetrically to the central hub. At the end of each arm, a propeller driven by a brushless Direct Current (DC) motor is attached. Each propeller produces an upward thrust and, since they are located outside the center of gravity, differential thrust is used to rotate the hexarotor. In addition, the rotation of the propellers also produces a torque in the opposite direction of the rotation of the motors; therefore, there must be two groups of rotors spinning in the opposite direction for the purpose of making this reaction torque equal to zero.

The pose of an hexarotor is given by its position $\zeta = [x, y, z]^T$ and its orientation $\eta = [\phi, \theta, \psi]^T$ in the three Euler angles roll, pitch and yaw, respectively. For the sake of simplicity, $sin(\cdot)$ and $cos(\cdot)$ will be abbreviated $s\cdot$ and $c\cdot$. The transformation from world frame O to body frame (Figure 1) is given by

$$\begin{bmatrix} x_B \\ y_B \\ z_B \end{bmatrix} = \begin{bmatrix} c\theta c\psi & c\theta s\psi & -s\theta \\ s\phi s\theta c\psi - c\phi s\psi & s\phi s\theta s\psi + c\phi c\psi & s\phi c\theta \\ c\phi s\theta c\psi + s\phi s\psi & c\theta s\theta s\psi - s\phi c\psi & c\phi c\theta \end{bmatrix} \begin{bmatrix} x_W \\ y_W \\ z_W \end{bmatrix}. \tag{1}$$

The dynamic model of the robot expressed in the body frame in Newton–Euler formalism is obtained as in [26].

$$\begin{bmatrix} m\mathbf{I}_{3\times 3} & 0 \\ 0 & \mathbf{I} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{V}} \\ \dot{\boldsymbol{\omega}} \end{bmatrix} + \begin{bmatrix} \boldsymbol{\omega} \times m\mathbf{V} \\ \boldsymbol{\omega} \times \boldsymbol{I}\boldsymbol{\omega} \end{bmatrix} = \begin{bmatrix} \mathbf{F} \\ \boldsymbol{\tau} \end{bmatrix}, \tag{2}$$

where I is the 3×3 inertia matrix; V the lineal speed vector and ω the body angular speed. The equations of motion for the helicopter of Figure 1 can be written as in [27]

$$\begin{aligned} \dot{\boldsymbol{\zeta}} &= \boldsymbol{v}, \\ \dot{\boldsymbol{v}} &= -ge_3 + \boldsymbol{R}\left(\frac{b}{m}\sum \Omega_i^2\right), \\ \dot{\boldsymbol{R}} &= \boldsymbol{R}\hat{\boldsymbol{\omega}}, \\ \boldsymbol{I}\dot{\boldsymbol{\omega}} &= -\boldsymbol{\omega} \times \boldsymbol{I}\boldsymbol{\omega} - \sum \boldsymbol{J_r}\left(\boldsymbol{\omega} \times e_3\right)\Omega_i + \boldsymbol{\tau_a}, \end{aligned} \tag{3}$$

where $\boldsymbol{\zeta}$ is the position vector, $\boldsymbol{R}$ the rotation matrix from the body frame to the world frame, $\dot{\boldsymbol{R}}$ represents the rotation dynamics, $\hat{\boldsymbol{\omega}}$ represents the skew symmetric matrix, Ω is the rotor speed, $\boldsymbol{I}$ the body inertia, $\boldsymbol{J_r}$ the rotor inertia, b is the thrust factor and τ is the torque applied to the body frame due to the rotors. Since we are dealing with an hexarotor, this torque vector differs from the well-known quadrotor torque vector and, if we are working with a structure like the one in Figure 2, it can be written as

$$\tau_a = \begin{pmatrix} bl\left(-\Omega_2^2 + \Omega_5^2 + \frac{1}{2}\left(-\Omega_1^2 - \Omega_3^2 + \Omega_4^2 + \Omega_6^2\right)\right) \\ \frac{\sqrt{3}}{2}bl\left(-\Omega_1^2 + \Omega_3^2 + \Omega_4^2 - \Omega_6^2\right) \\ d\left(-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2 - \Omega_5^2 + \Omega_6^2\right) \end{pmatrix}, \tag{4}$$

where l is the distance from the center of gravity of the robot to the rotor and b is the thrust factor. The full dynamic model is

$$\begin{aligned} \ddot{x} &= \left(\cos\phi sin\theta cos\psi + \sin\phi sin\psi\right)\frac{U_1}{m}, \\ \ddot{y} &= \left(\cos\phi sin\theta \sin\psi - \sin\phi\cos\psi\right)\frac{U_1}{m}, \\ \ddot{z} &= -g + \left(\cos\phi\cos\theta\right)\frac{U_1}{m}, \\ \ddot{\phi} &= \dot{\theta}\dot{\psi}\left(\frac{I_y - I_z}{I_x}\right) - \frac{J_r}{I_x}\dot{\theta}\Omega + \frac{l}{I_x}U_2, \\ \ddot{\theta} &= \dot{\phi}\dot{\psi}\left(\frac{I_z - I_x}{I_y}\right) - \frac{J_r}{I_y}\dot{\phi}\Omega + \frac{l}{I_y}U_3, \\ \ddot{\psi} &= \dot{\phi}\dot{\theta}\left(\frac{I_x - I_y}{I_z}\right) + \frac{l}{I_z}U_4, \end{aligned} \tag{5}$$

where U_1, U_2, U_3, U_4 and Ω represent the system inputs and in the case of the hexarotor are obtained as follows:

$$\begin{aligned} U_1 &= b\left(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2 + \Omega_5^2 + \Omega_6^2\right), \\ U_2 &= bl\left(-\Omega_2^2 + \Omega_5^2 + \frac{1}{2}\left(-\Omega_1^2 - \Omega_3^2 + \Omega_4^2 + \Omega_6^2\right)\right), \\ U_3 &= \frac{\sqrt{3}}{2}bl\left(-\Omega_1^2 + \Omega_3^2 + \Omega_4^2 - \Omega_6^2\right), \\ U_4 &= d\left(-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2 - \Omega_5^2 + \Omega_6^2\right), \end{aligned} \tag{6}$$

where d is the drag factor.

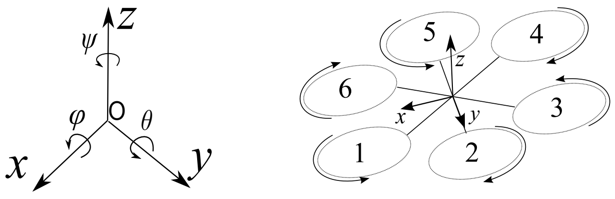


Figure 1. Structure of hexarotor and coordinate frames.

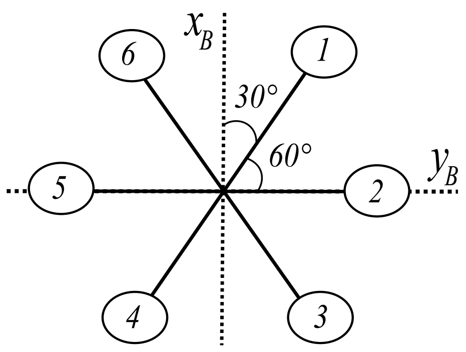


Figure 2. Geometry of hexarotor.

3. Visual Servo Control

In this paper, we use an Image Based Visual Servo control approach and the eye-in-hand case. The camera is mounted on the robot and the movement of the hexarotor induces camera motion [28].

The purpose of the vision based control is to minimize the error

$$\mathbf{e}\,(t) = \mathbf{s}\,(\mathbf{m}\,(t)\,, \mathbf{a}) - \mathbf{s}^{*}, \tag{7}$$

where $\mathbf{s}$ is the vector of captured features and in the function of a vector of 2D points coordinates in the image plane, $\mathbf{m}(t)$ and $\mathbf{a}$ are the set of known parameters of the camera (e.g., camera intrinsic parameters). Vector $\mathbf{s}^*$ contains the desired values. Since the error $\mathbf{e}(t)$ is defined on the image space and the robot moves in the 3D space, it is necessary to relate changes in the image features with the hexarotor displacement. The image Jacobian [29] (also known as interaction matrix) captures the relation between features and robot velocities as shown

$$\dot{\mathbf{s}} = \mathbf{L}_s \mathbf{v}_c, \tag{8}$$

where $\dot{\mathbf{s}}$ is the variation of the features position, $\mathbf{L}_s$ is the interaction matrix and $\mathbf{v}_c = (v_c, \dot{\omega}_c)$ denotes the camera translational ($\dot{\mathbf{v}}_\mathbf{c}$) and rotational ($\dot{\omega}_c$) velocities. Considering $\mathbf{v}_c$ as the control input, we can try to ensure an exponential decrease of the error with

$$\mathbf{v}_c = -\lambda \mathbf{L}_s{}^{+} \mathbf{e}, \tag{9}$$

where λ is a positive constant, $\mathbf{L}_s \in \mathbb{R}^{6\times k}$ is the pseudo-inverse of $\mathbf{L}_s$, k is the number of features and $\mathbf{e}$ the feature error.

To calculate the interaction matrix, consider a 3D point X with coordinates (X, Y, Z) in the camera frame, the projected point in the image plane x with coordinates (x, y) is defined as

$$\begin{aligned} x &= X/Z = (u - c_u)\,/f\alpha, \\ y &= Y/Z = (v - c_v)\,/f, \end{aligned} \tag{10}$$

where (u, v) are the coordinates of the point in the image space expressed in pixel units, (c_u, c_v) are the coordinates of the principal point, α is the ratio of pixel dimensions and f the focal length. If we derive (10), we have

$$\dot{x} = \frac{\dot{X} - x\dot{Z}}{Z} \qquad \dot{y} = \frac{\dot{Y} - y\dot{Z}}{Z}. \tag{11}$$

The relation between a fixed 3D point and the camera spatial velocity is stated as follows:

$$\dot{X} = -v_c - \omega_c \times X. \tag{12}$$

Then, we can write the derivatives of the 3D coordinates as

$$\begin{aligned} \dot{X} &= -v_x - \omega_y Z + \omega_z Y, \\ \dot{Y} &= -v_y - \omega_z X + \omega_x Z, \\ \dot{Z} &= -v_z - \omega_x Y + \omega_y X. \end{aligned} \tag{13}$$

Substituting (13) in (11), we can state the pixel coordinates variation as follows:

$$\begin{aligned} \dot{x} &= -\frac{v_x}{Z} + \frac{x v_z}{Z} + xy\omega_x - \left(1 + x^2\right)\omega_y + y\omega_z, \\ \dot{y} &= -\frac{v_y}{Z} + \frac{y v_z}{Z} + \left(1 + y^2\right)\omega_x - xy\omega_y - x\omega_z, \end{aligned} \tag{14}$$

which can be written

$$\dot{\mathrm{x}} = \mathbf{L}_x \mathbf{v}_c \tag{15}$$

with

$$\mathbf{L}_x = \begin{bmatrix} -\frac{1}{Z} & 0 & \frac{x}{Z} & xy & -\left(1 + x^2\right) & y \\ 0 & -\frac{1}{Z} & \frac{y}{Z} & 1 + y^2 & -xy & -x \end{bmatrix}, \tag{16}$$

where Z is the actual distance from the vision sensor to the feature, for this reason, most of the IBVS algorithms need to approximate this depth. In our case, we use an RGB-D sensor and this distance is known. To control the six degrees of freedom (DoF), at least three points are necessary [28]. In that particular case, we would have three interaction matrices $\mathbf{L}_{x_1}$, $\mathbf{L}_{x_2}$, $\mathbf{L}_{x_3}$, one for each feature, and the complete interaction matrix is now

$$\mathbf{L}_x = \begin{bmatrix} \mathbf{L}_{x_1} \\ \mathbf{L}_{x_2} \\ \mathbf{L}_{x_3} \end{bmatrix}. \tag{17}$$

When using three points, there are some configurations for which $\mathbf{L}_x$ is singular and four global minima [30]. More precisely, there are four poses for the camera such that $\dot{\boldsymbol{s}} = 0$, and these four poses are impossible to differentiate [31]. With this in mind, it is usual to consider more points [28].

On the other hand, only one pose achieves $\boldsymbol{s} = \boldsymbol{s}^*$ when using four points. Moreover, we can use the pseudo-inverse or the transpose of the interaction matrix indistinctly to solve for $\mathbf{v}_c$ in (15) [32,33].

In this paper, four points are used. In addition, our pattern does not move and because of the nature of the hexarotor, we suppose that the pattern will never be rotated since any rotation in roll or pitch will produce a translation. In other words, the hexarotor is an underactuated system, and it means there are some acceleration directions that can be only produced by a combination of the actuators. Most of the time, this is due to a lower number of actuators than degrees of freedom of the system; however, in the hexarotor, there is no actuator that can produce by itself a translational acceleration in the x and y directions. In consequence, it is not possible to have this kind of robot static and tilted at the same time, and, consequently, rotational velocities related to roll and pitch in $\mathbf{v}_c$ are 0.

4. Control of Hexarotor

The hexarotor has four control inputs U_i, U_1, which represents the translation in the z-axis, U_2, which represents the roll torque, U_3 the pitch, and U_4 represents the yaw torque. The visual algorithm act as a proportional controller, where λ in (9) works as a proportional gain. When combined with the Artificial Neural Network (ANN) based PID, we can adapt not only this proportional gain but also the derivative and the integral gains. Since the system is underactuated, we can use the translational velocities $[\dot{x}, \dot{y}]$ computed by IBVS as input roll and pitch torques, and the error will be reduced. This is shown in Figure 3.

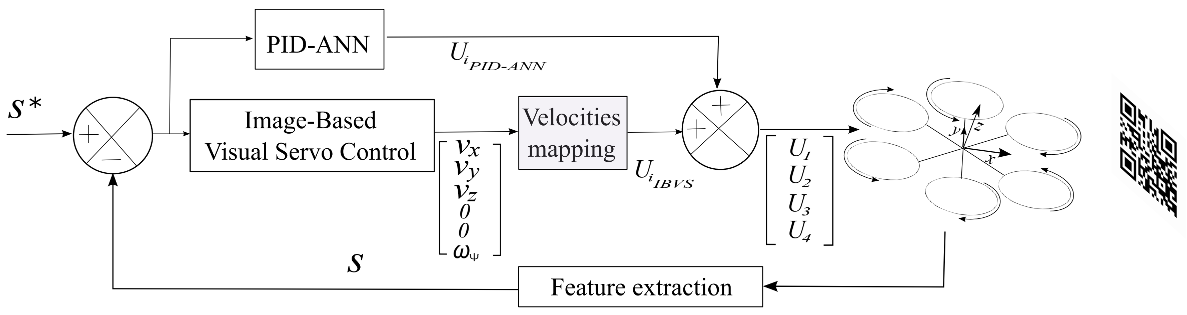


Figure 3. Block diagram of our Image Based Visual Servo (IBVS) control algorithm combined with Artificial Neural Network based PID (PID-ANN). The Artificial Neural Network based PID module consists of four modules (one for each Degree of Freedom). The output of the IBVS block is the velocity vector with angular velocity equal to 0 for roll and pitch since we assume the pattern will never be rotated in those angles. The output of IBVS control block consists of translational velocities v_x, v_y, v_z, which must be mapped to U_2, U_3 and U_1, respectively, in order to be added to the PID-ANN output. These U_i control actions will be traduced to translational displacement of the robot.

The velocity mapping block in Figure 3 traduces the velocities vector from IBVS to hexarotor inputs, i.e., v_x = roll, v_y = thrust, v_z = pitch and ω_ψ = yaw. In our case, $\omega_\phi = 0$ and $\omega_\theta = 0$, since we assume the pattern will never be rotated because it is an underactuated system.

5. Neural Network Based PID

Considering the control loop with unitary feedback as shown in Figure 4.

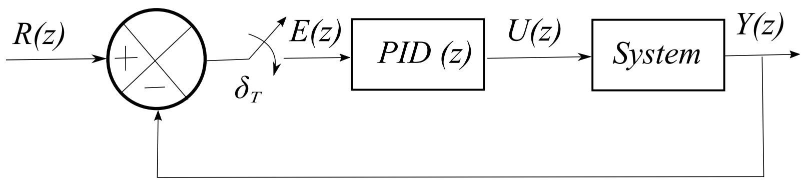


Figure 4. Control loop for conventional PID in discrete time.

Conventional digital Proportional-Integral-Derivative (PID) with unitary feedback is described in [34] and the control law is given by

$$U(z) = \left[K_P + \frac{K_I}{(1 - z^{-1})} + K_D \left(1 - z^{-1}\right) \right] E(z), \tag{18}$$

where $E(z)$ is the error calculated as the difference between the reference signal and the system output $R(z) - Y(z)$. The terms K_P, K_I, and K_D are the proportional, integral and derivative gains, respectively. These gains are related as follows:

$$K_P = K - \frac{K_I}{2} \qquad K_I = \frac{KT}{T_i} \qquad K_D = \frac{KT_d}{T}, \tag{19}$$

where K is the gain, T is the sample time, T_i is the integration time and T_d the derivative time. Applying the inverse Z-transform to (18), the PID sequence $u(k)$ is given by

$$u(k) = u(k-1) + K_P e(k) + K_I[e(k) - 2e(k-1) + e(k-2)] + K_D[e(k) - e(k-1)]. \tag{20}$$

Despite conventional PID being widely used to control these vehicles due to its simplicity and performance, it is not intended to control highly nonlinear systems, such as hexarotors.

In order to handle these nonlinearities, an ANN based PID controller is used. The purpose of the ANN is not only to deal with the nonlinear system, but also adjusting the PID controller gains to the end that it can also handle with uncertainties in the model. The topology of the PID-ANN used is shown in Figure 5.

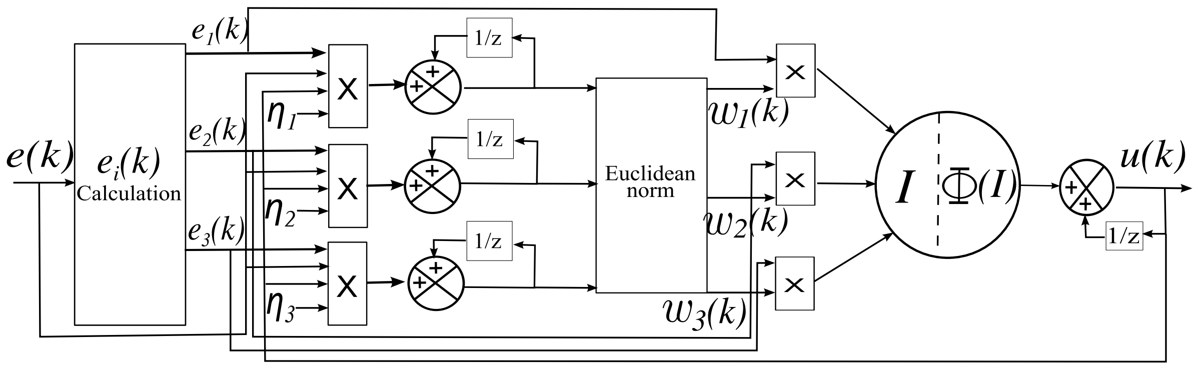


Figure 5. PID-ANN topology. There is one module PID-ANN for every DoF. $e(k)$ represents the error in that DoF, $e_i(k)$ is a vector that represents the error, derivative of the error and the integral of the error and η_i is the learning rate, which must be heuristically selected to be small enough to avoid saturation of the neuron and not necessarily the same for every gain (proportional, derivative and integral). The Euclidean norm block normalizes the neuron weights w_i in order to avoid divergence since the neuron weights are the PID controller gains. The output of the neuron is the control input $u(k)$ that will be traduced as thrust (U_1), roll (U_2), pitch (U_3) or yaw (U_4), depending on the state error at the input of the ANN.

From Figure 5, the $e_i(k)$ vector represents the proportional error, the derivative of the error and the integral of the error. They are defined as follows:

$$\begin{aligned} e_1(k) &= e(k), \\ e_2(k) &= e(k) - 2e(k-1) + e(k-2), \\ e_3(k) &= e(k) - e(k-1). \end{aligned} \tag{21}$$

Accordingly, the control law of the conventional PID can be rewritten as

$$u(k) = u(k-1) + K_P e_1(k) + K_I e_2(k) + K_D e_3(k). \tag{22}$$

The Neuron input is defined as

$$I = \sum_{i=1}^{3} e_i(k)\, w_i(k), \tag{23}$$

where vector $w_i(k)$ represents the weights of the network, which are incremented by

$$\Delta w_i = \eta_i e_i\left(k\right) e\left(k\right) u\left(k\right) \tag{24}$$

with a learning factor η. The new value of $w_i(k)$ will be

$$w_i'\left(k\right) = w_i\left(k-1\right) + \Delta w_i\left(k\right). \tag{25}$$

The Euclidean norm will be used to limit the values of $w_i(k)$ as

$$w_i\left(k\right) = \frac{w_i'\left(k\right)}{\left\| \sum\limits_{i=1}^{3} w_i'\left(k\right) \right\|}. \tag{26}$$

The activation function of the neuron is the hyperbolic tangent; therefore, the output will be

$$\Phi\left(I\right) = A\frac{1-e^{-Ib}}{1+e^{-Ib}}, \tag{27}$$

where A is a gain factor to escalate the maximum value of the activation function, which is between $[-1,1]$ and b is a scalar to avoid saturation of the neuron. The control law of the ANN based PID is expressed as follows:

$$u\left(k\right) = u\left(k-1\right) + \Phi\left(I\right), \tag{28}$$

and there is one PID-ANN module for every U_i to control in (6). U_i has information of the rotor speed combination necessary to achieve a specific rotation, i.e., if the robot needs to move in more than one direction at the same time, there will be more than one U_i with values different to zero.

6. Simulation Results

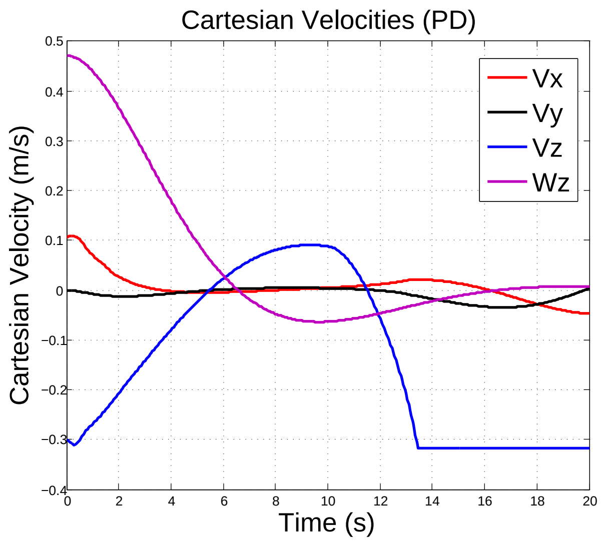
Simulations are implemented in Matlab (Matlab R2016a, The MathWorks Inc., Natick, MA, USA) using the Robotics Toolbox [35]. For the visual servo algorithm, four points are used. In the first experiment, the robot starts on the ground and has to reach a certain position given by these 2D points. With the intention of proving the algorithms, we simulate uncertainty of the system changing two parameters separately in two simulations.

In the first simulation, at the second 10, the mass of the robot has been increased 50%. It can be seen that conventional PID controller is unable to keep the position. The results are shown in Figure 6.

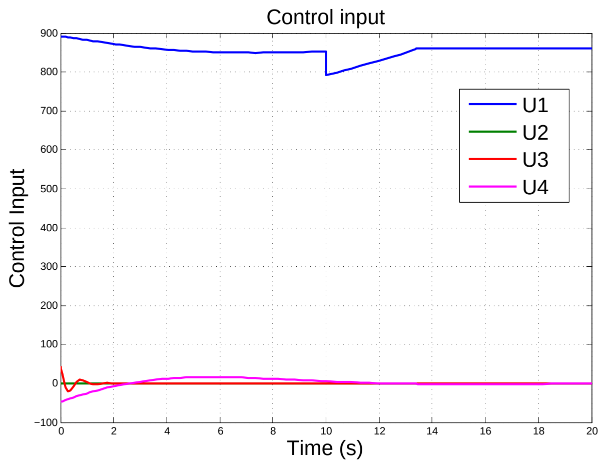
In the second simulation, the mass of the system remains constant, but the moment of inertia I_x is increased from $I_x = 0.0820$ to $I_x = 0.550$. Figure 7 shows the results of using conventional PID when the moment of inertia is increased. The results show that the control input is excessively high to the robot (Figure 7b), making the system unable to follow the reference (Figure 7c).

In the following simulations, the PID-ANN is now controlling the system under the same conditions. The mass has been incremented at second 10. It can be seen in Figure 8 that the controller can be adapted to this mass increment and keep the reference.

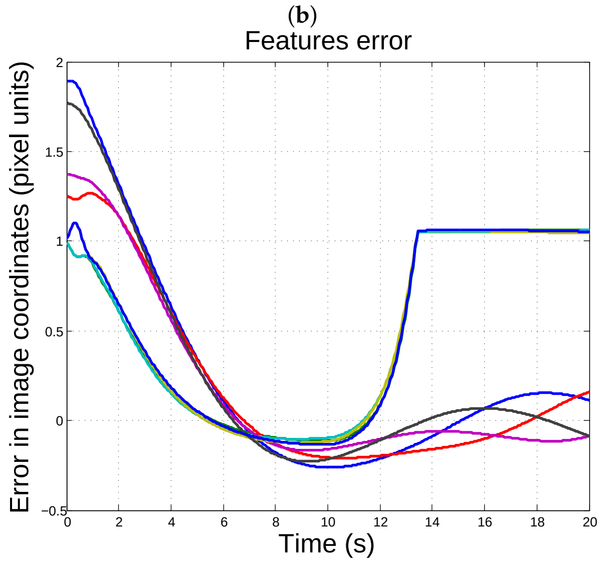
Finally, we show, in Figure 9 the results of changing the moment of inertia I_x while mass remains constant. In contrast with conventional PID, the PID-ANN is able to keep its position.



(**a**)

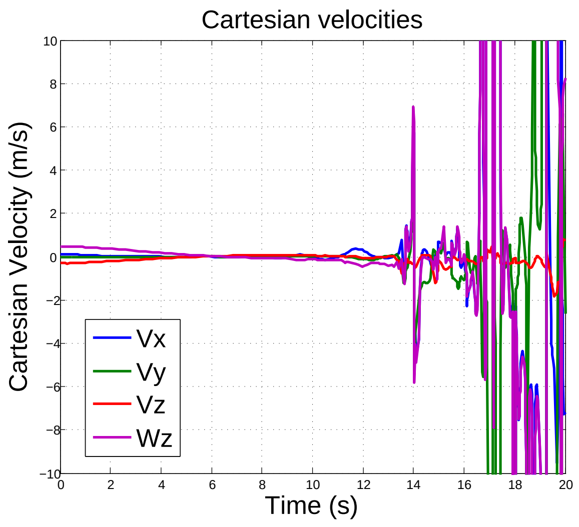


(**b**)

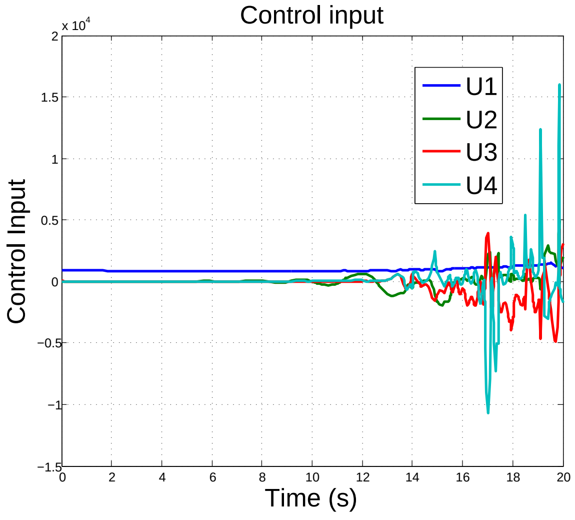


(**c**)

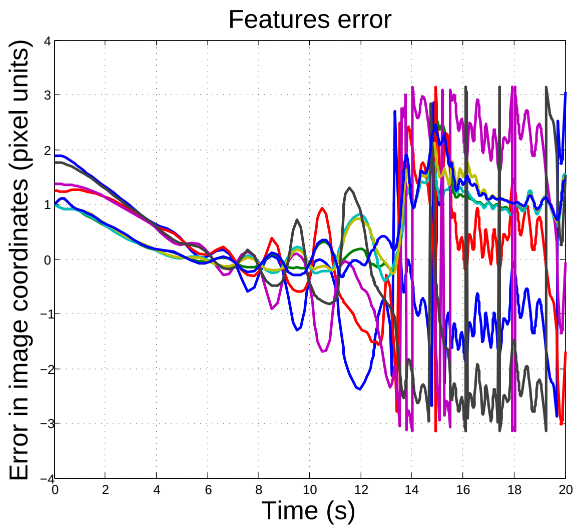
Figure 6. Simulation using conventional PID. At 10 s, the mass of the system is incremented 50% and $\lambda = 0.3$. (**a**) Cartesian velocities; (**b**) control input; (**c**) features error.



(**a**)

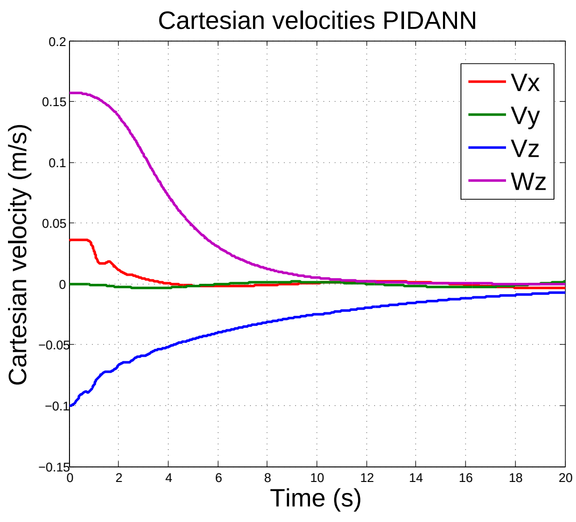


(**b**)

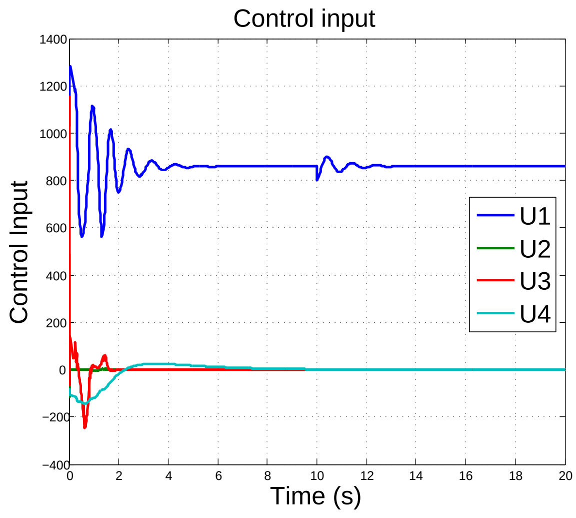


(**c**)

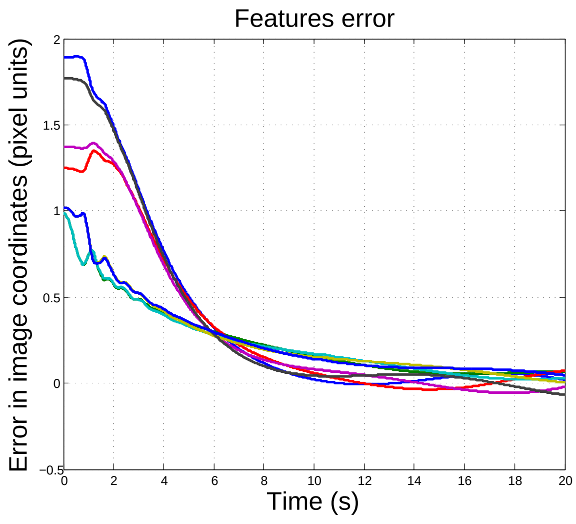
Figure 7. Simulation using conventional PID. Moment of inertia is increased. Mass remains constant. It can be seen that the robot is not stable when I_x changes. (**a**) Cartesian velocities; (**b**) control input; (**c**) features error.



(**a**)

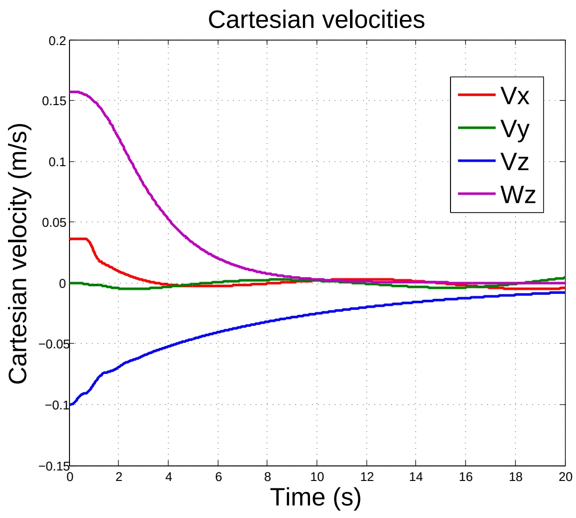


(**b**)

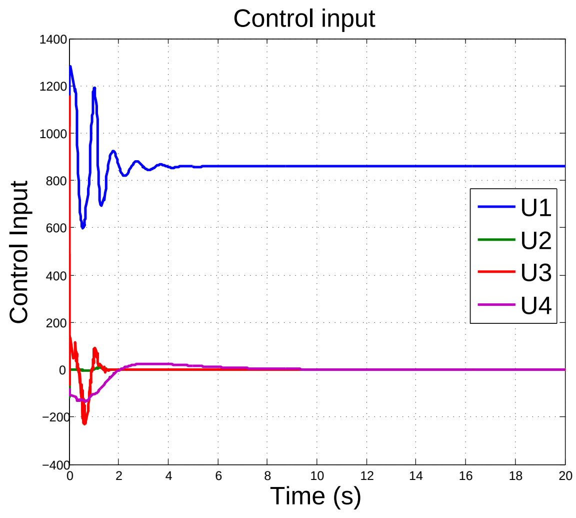


(**c**)

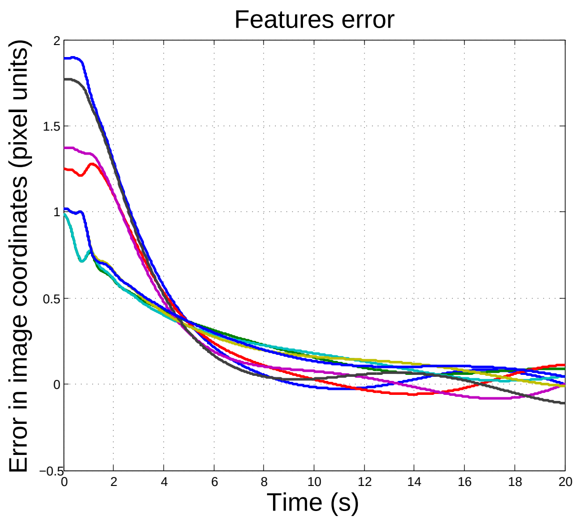
Figure 8. Simulation using conventional PID-ANN. At 10 s, the mass of the system is incremented 50% and $\lambda = 0.3$. The system remains at desired position. (**a**) Cartesian velocities; (**b**) control input; (**c**) features error.



(**a**)



(**b**)



(**c**)

Figure 9. Simulation using ANN based PID. Moment of inertia is increased. Mass remains constant. It can be seen that the robot remains stable at desired position when I_x changes. (**a**) Cartesian velocities; (**b**) control input; (**c**) features error.

7. Experimental Results

The hexarotor used in the experiments is the Asctec Firefly (Ascending Technologies, Krailling, Germany). The actual configuration of the experiment is shown in Figure 10. The vision sensor has been changed, we use the Intel RealSense R200 camera (Intel, Santa Clara, CA, USA) with RGB and infrared depth sensing features and an indoor range from 0.4 m to 2.8 m. This change in the vision sensor will modify robot mass and moment of inertia. This uncertainty can be absorbed by the neural network.

Vision information is highly noisy and presents a high computational cost even when working with low resolution images (in this case, 640 × 480). The more time the algorithm uses in image capture and processing phase, the more error will exist between what the robot sees and the actual position. Computer vision algorithms, such as optical flow approaches, requires tracking of a set of n features using some kind of descriptor. Other approaches use stereo vision but that requires a 3D reconstruction. We propose to use only four points to reduce this time.

It is important to note that coordination between vision sensors, neural network and model system working at different processing stages and its communication at their respective architectures is crucial to achieve real-time implementation. A QR-code is used, and we track it with a Zbar bar code reader library. This pattern is chosen because of its robustness to rotations and illumination changes. The algorithms are implemented on the onboard computer of the hexarotor.

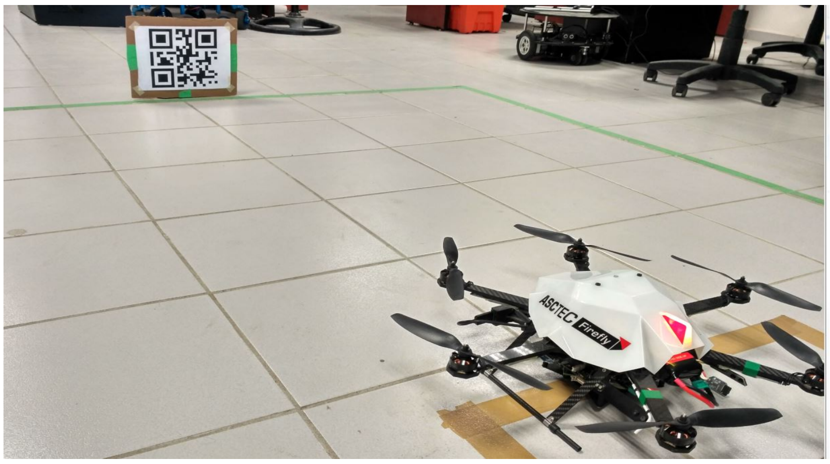
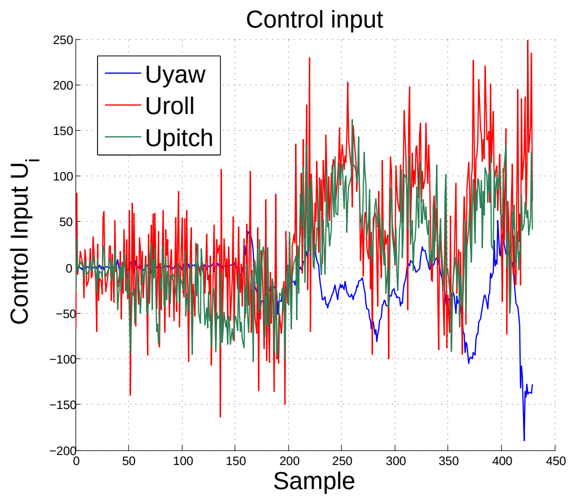


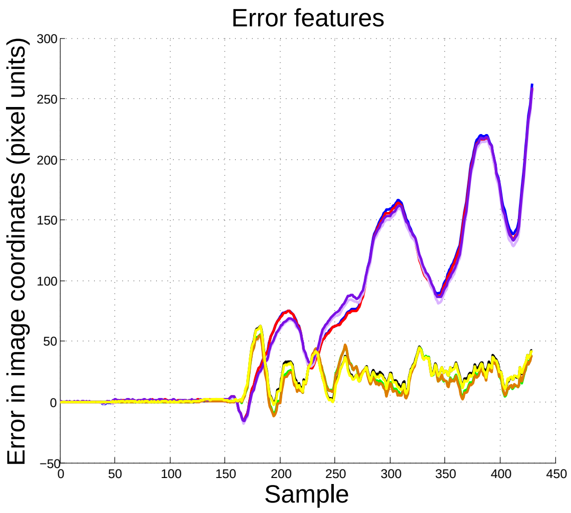
Figure 10. Actual experiment configuration. The corners of the Quick Response (QR) code represent the 3D features.

For a first experiment, the moment of inertia and mass of the system changed and a previously tuned conventional PID controller will be compared with the proposed algorithm. Figure 11 shows results when the pattern is fixed at a certain position and the hexarotor is at hover position. As can be seen in Figure 11b, the conventional controller cannot achieve system stabilization at a fixed position when the model changes. Table 1 shows the Root Mean Square Error (RMSE) and the Average Absolute Deviation (AAD) in pixel units. The pair (x_i, y_i) is the location of feature $(i = 1, 2, 3, 4)$ in image coordinates.

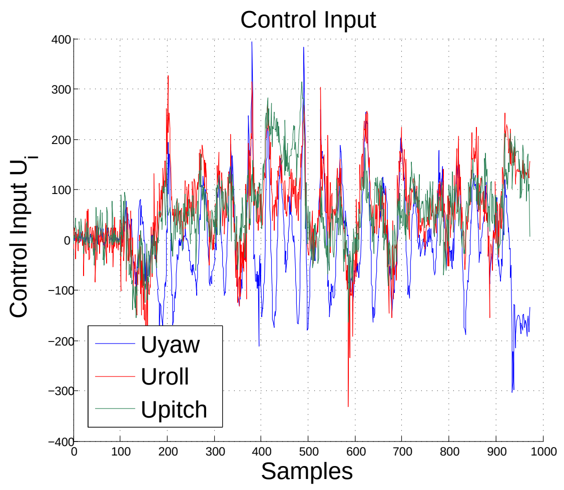
In Figure 11b, the solid increasing lines represent the x position of the four features in image coordinates (pixel units). As can be seen, if a conventional PID is not correctly tuned for this specific system, its position diverges. On the other hand, when the system is controlled by the PID-ANN, its position does not diverge (Figure 11d) even when the controller has not been previously tuned.



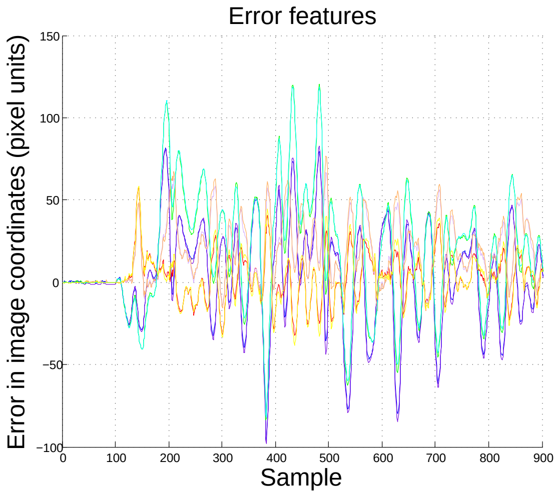
(**a**)



(**b**)



(**c**)



(**d**)

Figure 11. Experimental results when mass and moment of inertia changed. The pattern remains at the same position during the test. (**a**) control input conventional PID; (**b**) error conventional PID; (**c**) control input ANN-PID; (**d**) error ANN-PID.

Table 1. Controllers' comparison.

	Root Mean Square Error (Pixel Units)							
	x_1	y_1	x_2	y_2	x_3	y_3	x_4	y_4
PID	1741.9	393.1	1712.0	339.1	1677.5	329.2	1723.9	380.1
ANNPID	212.416	66.549	191.1447	57.4298	824.085	59.7654	861.8207	62.6225
	Average Absolute Deviation (Pixel Units)							
	x_1	y_1	x_2	y_2	x_3	y_3	x_4	y_4
PID	55.2177	9.6437	54.0636	10.5457	52.4827	10.704	53.069	9.6577
ANNPID	43.2281	7.4259	42.5686	11.3874	46.6832	11.484	47.5576	7.5503

Comparison Table between conventional Proportional Integral Derivative (PID) controller and the Artificial Neural Network based PID controller (ANNPID).

Once ANN-PID has demonstrated its effectiveness over the PID controller, the experiment is repeated, but now the QR pattern has movement. As shown in Figure 12, the hexarotor does not lose sight of the objective.

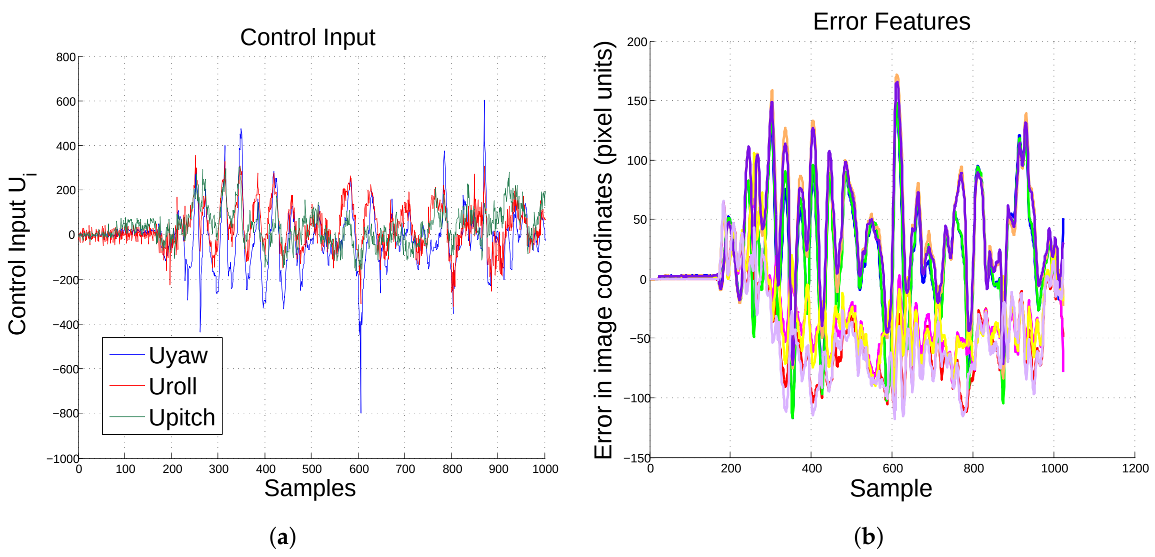


Figure 12. Experimental results at hover position when mass and moment of inertia changed. The pattern is moving during the test. (**a**) control input ANN-PID; (**b**) error ANN-PID.

8. Conclusions

In this paper, we propose a Neural Network based PID controller with visual feedback to control an hexarotor. The hexarotor is equipped with an RGB-D sensor that allows for estimating the feature error, this error has been used to compute the camera velocities. The proposed approach is able to deal with delays due to image processing, system uncertainties, noises and changes in the model since the ANN is continuously adapting the PID gains. In contrast with conventional PID controllers, where it is mandatory to tune it according to a specific system, the ANN can deal with nonlinearities and changes in the system.

Acknowledgments: The authors thank the support of CONACYT Mexico, through Projects CB256769 and CB258068 (Project supported by Fondo Sectorial de Investigacion para la Educacion).

Author Contributions: All of the experiments reported in this paper have been designed and performed by Javier Gomez-Avila and Carlos Villaseñor. The data and results presented on this work were analyzed and validated by Carlos Lopez-Franco, Alma Y. Alanis and Nancy Arana-Daniel. All authors are credited for their contribution on the writing and edition of the presented manuscript.

Conflicts of Interest: The authors declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Abbreviations

The following abbreviations are used in this manuscript:

UAV	Unmanned Aerial Vehicle
IBVS	Image Based Visual Servo
VTOL	Vertical Take-Off and Landing
PID	Proportional Integral Derivative
ANN	Artificial Neural Network
IMU	Inertial Measurement Unit
PD	Proportional Derivative
RMSE	Root Mean Square Error
AAD	Average Absolute Deviation

References

1. Bouabdallah, S.; Siegwart, R. Full control of a quadrotor. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, San Diego, CA, USA, 29 October–2 November 2007; pp. 153–158.
2. Achtelik, M.; Zhang, T.; Kuhnlenz, K.; Buss, M. Visual tracking and control of a quadcopter using a stereo camera system and inertial sensors. In Proceedings of the International Conference on Mechatronics and Automation, Changchun, China, 9–12 August 2009; pp. 2863–2869.
3. Klose, S.; Wang, J.; Achtelik, M.; Panin, G.; Holzapfel, F.; Knoll, A. Markerless, vision-assisted flight control of a quadrocopter. In Proceedings of the International Conference on Intelligent Robots and Systems (IROS), Taipei, Taiwan, 18–20 October 2010; pp. 5712–5717.
4. Zhang, T.; Kang, Y.; Achtelik, M.; Kuhnlenz, K.; Buss, M. Autonomous hovering of a vision/IMU guided quadrotor. In Proceedings of the International Conference on Mechatronics and Automation, Changchun, China, 9–12 August 2009; pp. 2870–2875.
5. Angeletti, G.; Valente, J.P.; Iocchi, L.; Nardi, D. Autonomous indoor hovering with a quadrotor. In Proceedings of the Workshop SIMPAR, Venice, Italy, 3–4 November 2008; pp. 472–481.
6. Stefanik, K.V.; Gassaway, J.C.; Kochersberger, K.; Abbott, A.L. UAV-based stereo vision for rapid aerial terrain mapping. *GISci. Remote Sens.* **2011**, *48*, 24–49.
7. Kim, J.H.; Kwon, J.W.; Seo, J. Multi-UAV-based stereo vision system without GPS for ground obstacle mapping to assist path planning of UGV. *Electron. Lett.* **2014**, *50*, 1431–1432.
8. Salazar, S.; Romero, H.; Gómez, J.; Lozano, R. Real-time stereo visual servoing control of an UAV having eight-rotors. In Proceedings of the 2009 6th International Conference on IEEE Electrical Engineering, Computing Science and Automatic Control, Toluca, Mexico, 10–13 October 2009; pp. 1–11.
9. Hrabar, S.; Sukhatme, G.S.; Corke, P.; Usher, K.; Roberts, J. Combined optic-flow and stereo-based navigation of urban canyons for a UAV. In Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005), Edmonton, AB, Canada, 2–6 August 2005; pp. 3309–3316.
10. Altug, E.; Ostrowski, J.P.; Mahony, R. Control of a quadrotor helicopter using visual feedback. In Proceedings of the IEEE International Conference on Robotics and Automation, Washington, DC, USA, 11–15 May 2002; Volume 1; pp. 72–77.
11. Romero, H.; Benosman, R.; Lozano, R. Stabilization and location of a four rotor helicopter applying vision. In Proceedings of the American Control Conference, Minneapolis, MN, USA, 14–16 June 2006.
12. Saripalli, S.; Montgomery, J.F.; Sukhatme, G.S. Visually guided landing of an unmanned aerial vehicle. *IEEE Trans. Robot. Autom.* **2003**, *19*, 371–380.
13. Wu, A.D.; Johnson, E.N.; Proctor, A.A. Vision-aided inertial navigation for flight control. *J. Aerosp. Comput. Inf. Commun.* **2005**, *2*, 348–360.
14. Azinheira, J.R.; Rives, P.; Carvalho, J.R.; Silveira, G.F.; De Paiva, E.C.; Bueno, S.S. Visual servo control for the hovering of all outdoor robotic airship. In Proceedings of the International Conference on Robotics and Automation, Washington, DC, USA, 11–15 May 2002, Volume 3; pp. 2787–2792.
15. Bourquardez, O.; Chaumette, F. Visual servoing of an airplane for auto-landing. In Proceedings of the International Conference on Intelligent Robots and Systems, San Diego, CA, USA, 29 October–2 November 2007; pp. 1314–1319.
16. Mejias, L.; Saripalli, S.; Campoy, P.; Sukhatme, G.S. Visual servoing of an autonomous helicopter in urban areas using feature tracking. *J. Field Robot.* **2006**, *23*, 185–199.
17. Serres, J.; Dray, D.; Ruffier, F.; Franceschini, N. A vision-based autopilot for a miniature air vehicle: Joint speed control and lateral obstacle avoidance. *Auton. Robots* **2008**, *25*, 103–122.
18. Hamel, T.; Mahony, R. Visual servoing of an under-actuated dynamic rigid-body system: An image-based approach. *IEEE Trans. Robot. Autom.* **2002**, *18*, 187–198.
19. Alaimo, A.; Artale, V.; Milazzo, C.L.R.; Ricciardello, A. PID controller applied to hexacopter flight. *J. Intell. Robot. Syst.* **2014**, *73*, 261–270.
20. Ceren, Z.; Altuğ, E. Vision-based servo control of a quadrotor air vehicle. In Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA), Daejeon, Korea, 15–18 December 2009; pp. 84–89.

21. Rivera-Mejía, J.; Léon-Rubio, A.; Arzabala-Contreras, E. PID based on a single artificial neural network algorithm for intelligent sensors. *J. Appl. Res. Technol.* **2012**, *10*, 262–282.
22. Yang, J.; Lu, W.; Liu, W. PID controller based on the artificial neural network. In Proceedings of the International Symposium on Neural Networks, Dalian, China, 19–21 August 2004; pp. 144–149.
23. Ge, S.S.; Zhang, J.; Lee, T.H. Adaptive neural network control for a class of MIMO nonlinear systems with disturbances in discrete-time. *IEEE Trans. Syst. Man Cybern. Part B Cybern.* **2004**, *34*, 1630–1645.
24. Freddi, A.; Longhi, S.; Monteriù, A.; Prist, M. Actuator fault detection and isolation system for an hexacopter. In Proceedings of the 2014 IEEE/ASME 10th International Conference on Mechatronic and Embedded Systems and Applications (MESA), Senigallia, Italy, 10–12 September 2014; pp. 1–6.
25. Sheng, Q.; Xianyi, Z.; Changhong, W.; Gao, X.; Zilong, L. Design and implementation of an adaptive PID controller using single neuron learning algorithm. In Proceedings of the 4th World Congress on Intelligent Control and Automation, Shanghai, China, 10–14 June 2002; Volume 3; pp. 2279–2283.
26. Moussid, M.; Sayouti, A.; Medromi, H. Dynamic modeling and control of a hexarotor using linear and nonlinear methods. *Int. J. Appl. Inf. Syst.* **2015**, *9*, doi:10.5120/ijais2015451411.
27. Bouabdallah, S.; Murrieri, P.; Siegwart, R. Design and control of an indoor micro quadrotor. In Proceedings of the International Conference on Robotics and Automation, New Orleans, LA, USA, 26 April–1 May 2004; Volume 5; pp. 4393–4398.
28. Chaumette, F.; Hutchinson, S. Visual servo control. I. Basic approaches. *IEEE Robot. Autom. Mag.* **2006**, *13*, 82–90.
29. Weiss, L.; Sanderson, A.; Neuman, C. Dynamic sensor-based control of robots with visual feedback. *IEEE J. Robot. Autom.* **1987**, *3*, 404–417.
30. Michel, H.; Rives, P. Singularities in the Determination of the Situation of a Robot Effector from the Perspective View of 3 Points. Ph.D. Thesis, INRIA, Valbonne, France, 1993.
31. Fischler, M.A.; Bolles, R.C. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM* **1981**, *24*, 381–395.
32. Chaumette, F. Potential problems of stability and convergence in image-based and position-based visual servoing. In *The Confluence of Vision and Control*; Springer: Berlin, Germany, 1998; pp. 66–78.
33. Wu, Z.; Sun, Y.; Jin, B.; Feng, L. An Approach to Identify Behavior Parameter in Image-based Visual Servo Control. *Inf. Technol. J.* **2012**, *11*, 217.
34. Ogata, K. *Discrete-Time Control Systems*; Prentice Hall: Englewood Cliffs, NJ, USA, 1995; Volume 2.
35. Corke, P.I. *Robotics, Vision & Control: Fundamental Algorithms in Matlab*; Springer: Berlin, Germany, 2011.