Received: Added at production | Revised: Added at production | Accepted: Added at production

DOI: xxx/xxxx

ARTICLE TYPE

Optimized outdoor navigation with obstacles detection, collision avoidance and path planning for an autonomous quadcopter

Orlando García[1] | Omar-Jacobo Santos-Sánchez*[3] | Sergio Salazar[2] | Hugo Romero[3] | Rogelio Lozano[2]

[1]Desarrollo Tecnológico DOI, SAS, Ciudad de México, México
[2]UMI-LAFMIA, CINVESTAV-IPN, Ciudad de México, México
[3]CITIS-ICBI, Autonomous Hidalgo State University, Hidalgo, México

Correspondence
*Omar-Jacobo Santos-Sánchez Email: omarj@uaeh.edu.mx

Summary

This paper presents a navigation method in outdoor environments when trees are present in the workspace, in order to detect the obstacles, a laser scanning sensor is used. When an obstacle is sensed, the path planning task for avoiding collisions is carried out with a geometrical approach. The trajectory tracking of the path found is performed with a suboptimal nonlinear control of finite horizon in discrete domain; as it was demonstrated in previous results, the sub-optimal nonlinear sequence allows to optimize energy. Satisfactory experimental results are obtained in outdoors using a four-rotor vehicle with a GPS sensor.

KEYWORDS:
Collision avoidance, path planning, suboptimal nonlinear control, quadrotor.

1 | INTRODUCTION

The problem of autonomous navigation using an aerial vehicle is a complex task because it involves many challenges: environmental recognition, obstacles detection, collision avoidance, decision making, path planning and updating, and also the aircraft control strategy to be applied. There are many works that address the navigation problem for aerial vehicles in the presence of obstacles, the differences between them mainly lie in the type of sensor used to identify the obstacles. For example, Ait-Jellal et al.[1] implemented a hybrid algorithm combining SLAM and stereo vision. The union of these algorithms allowed them to obtain a 3D map. Finally, for the planning of safe trajectories, the rapidly exploring random tree algorithm (RRT) was used in a three-dimensional space. The algorithm was implemented in a quadcopter, data was collected in outdoors and finally these data were analyzed in the Kitti odometry benchmark. Another similar work, where stereoscopic vision was also used, was developed by Barry et al.[2], where Open CV tools were used for the disparity analysis between characteristic points. This analysis was done only for a depth which allowed them to analyze images at high frame rate, this algorithm was implemented in a small Unmanned Aerial Vehicle (UAV) which was able to detect obstacles flying at a speed of $20m/s$. Other works include optimization methods to find an obstacle-free path, as it was proposed by Campos-Macías et al.[3], in that work, a method for planning trajectories in known environments was presented, the proposed algorithm was a fusion of sampling-based techniques and model-based optimization via quadratic programming; experimental results were obtained by using the Optitrack motion capture system in order to sense the positioning, while the control algorithm was computed in a ground station. There are also works that use a combination of one or more sensors to detect obstacles, as the results presented by Anis et. al[4], where an image sensor was used to recognize the objects position and an ultrasonic sensor was used to detect the distance to the obstacles. All the scheme was also implemented in a Quadcopter using a PID control. Bugayong et al.[5] also applied stereoscopic vision: this algorithm provides a disparity map which was used to obtain the actual depth measurements of the obstacles, the proposed algorithm was executed in a Raspberry Pi embedded computer and implemented on a quadcopter, experimental results were obtained in an indoor environment where a PID control was used to drive the trajectory tracking task, however this task was only executed in the rotational dynamics of the vehicle. Rahman et al.[6] reported the use of rotating ultrasonic sensor to detect obstacles, in order to characterize the behavior of the vehicle depending on the distance to obstacle, several flight states or modes were introduced. Experimental tests were conducted using a four-rotor vehicle. Mohanta et. al[7] have used an ultrasonic sensor for the detection of obstacles. These measurements were combined with data from an infrared sensor having a detection range of 1 meter. Several flight scenarios were proposed in order to track trajectories, the algorithm was tested in a four-rotor vehicle, which was controlled by a PID controller. Simulation and experimental results were carried out. Thanh et. al[8] synthesized a robust nonlinear control to avoid collisions on the aircraft, with a geometric approach, the obstacles were represented as a circumference to facilitate the algorithm calculations, the evasion process was carried out by changing the direction of the aircraft, the algorithm was implemented in a quadcopter vehicle to avoid one or multiple obstacles, numerical simulation results were obtained. Yang et. al[9] proposed a reactive obstacle avoidance system with a monocular camera, which used Convolutional Neural Networks to progressively estimate the depth of the obstacles. Simulation and experimental results were obtained, using a Quadcopter Parrot. This prototype sends the images to a server, where the analysis of obstacle avoidance action and control signals were computed. Other works are focused on the control synthesis based on optimization to track trajectories and obstacle avoidance problem. In fact, based on prior knowledge of the obstacles position, Mendoza-Soto et al.[10] reported an evasion collision of the considered aerial vehicle. However, the experiments were made in a controlled environment, the control strategy used was a constrained generalized predictive control which was implemented in a quadcopter. Experimental results were obtained using an artificial vision system to estimate the position of vehicle. A similar method for the detection and obstacle avoidance for multiple robots in unknown environments was presented by Yu et. al[11], a laser scanning sensor was used, and an optimization procedure was performed in order to choose the best obstacle-free path. Only simulation results of the proposed algorithm were presented.

According to the results exposed above, most of the algorithms for the obstacles avoidance problem are based on the artificial vision, and they could be affected by the illumination levels in outdoor environments. This is a common problem in most navigation systems based on computer vision, see Máthé[12]. Additionally, the computational cost could be relatively high if one aims to implement the algorithm in a low cost autopilot. The other option is the use of the laser scanning sensor. However, in the best knowledge of the authors only simulation results are presented or experimental results under controlled environments, and not in outdoor environments. **Additionally, the optimized nonlinear control used here, are not tested yet for detecting and obstacle avoidance tasks**. So, the main contributions of this paper could be summarized as follows:

- Implementation of a real time algorithm for obstacle detection and avoidance, on an aerial vehicle.
- Real-time planning and updating of the UAV trajectory, based on the analysis of the navigation environment, based on a laser scanning sensor.
- The real time implementation on the autopilot of a non-linear control algorithm based on energy optimization for trajectory tracking task.

This paper is organized as follows: In section 1, a brief introduction of the navigation process using an autonomous aerial vehicle is presented. Section 2 describes the mathematical model of the used platform. The applied control technique to track the trajectories is shown in section 3. Section 4 presents the navigation process which involves the process of data acquisition and processing, the process of detection and collision avoidance, and the trajectory planning and updating. Section 5 shows the description of the experimental platform used for the experimental tests. Section 6 presents the experimental results of the navigation process. In Section 7 the conclusions of the work are discussed. Finally the acknowledgments are given in Section 8.

2 | QUADCOPTER MODEL

The Quadcopter is a vehicle with four rotors, whose mathematical model is already well studied, for example see Lozano[13], Castillo[14] and Das[15]. In this paper we use the model proposed by Lozano[13], which is obtained from the Euler-Lagrange formalism and considering the following assumptions:

- The aircraft has a rigid and a symmetrical structure.
- The center of gravity of the vehicle coincides with the origin in the body frame.

- The propellers are rigid with a fixed pitch.
- At low velocities, aerodynamic effects can be neglected.
- The rotor dynamics is approximately equal to 1
- The angles are restricted: $-\pi/4 < \phi, \theta < \pi/4$ and $-\pi < \psi < \pi$.

.

So, the movement equations for the aerial vehicle are given in the following way:

$$\begin{aligned} m\ddot{x} &= -u\sin\theta \\ m\ddot{y} &= \sin\phi\cos\theta \\ m\ddot{z} &= \cos\phi\cos\theta - mg \end{aligned} \tag{1}$$

$$\begin{aligned} \ddot{\phi} &= \tau_\phi \\ \ddot{\theta} &= \tau_\theta \\ \ddot{\psi} &= \tau_\psi \end{aligned} \tag{2}$$

where $m = 1.4\,kg$ is the mass of the vehicle, (ϕ, θ, ψ) are the angles $roll$, $pitch$ and yaw which represent the attitude of the aircraft. The variables $(x,\ y,\ z)$ are the Cartesian coordinates and they represent the relative position of the center of mass with respect to an inertial frame, $g = 9.81\frac{m}{s^2}$ is the gravitational acceleration. The variables u, τ_ϕ, τ_θ and τ_ψ represent the control inputs. For the control synthesis of the vehicle, the mathematical model is divided into four subsystems which will be given in next section. The control strategy used was proposed originally by Santos[16], without obstacles detection, collision avoidance and path planning.

3 | CONTROL STRATEGY

In this section the control technique used to do the navigation task is presented. It is a sub-optimal discrete nonlinear sequence control of finite horizon, the continuous mathematical model is discretized by using the Euler's approximation. The synthesis of this control, originally presented by Santos[16], is used here to perform the trajectory tracking task. The main motivation to use a sub-optimal discrete nonlinear control (SDNLC) is to penalize the energy consumption, and at the same time the state has to track the reference with the fastest possible convergence rate. **In fact, the optimal control could be a good option to optimize energy and to penalize the convergence of the closed loop response of the plant. However, notice that the optimal control problem for nonlinear discrete and continuous systems is an open problem on the Dynamic Programming Approach, due to the difficulty to solve the Hamilton Jacobi Bellman equation for the general case, see for example Komaee[17] and Santos[16]. So numerical approximations (sub-optimal control) could be proposed, see Wang[18]. Nevertheless, as Komaee[17] states, an acceptable approximation that meets all the control requirements can be still difficult to obtain. In fact, for UAV's, if a nonlinear mathematical model is considered, a SDNLC could be used to satisfy the requirement of the energy saving. It will increase the flight time of the UAV, due to the battery usage optimization of the vehicle, but at the same time, the structure of the chosen SDNLC and its tuning, have to be reasonably simple to implement them on the UAV autopilot. The SDNLC is summarized briefly in the next section.**

3.1 | Suboptimal discrete nonlinear control for affine systems

Consider the nonlinear discrete affine system as follows

$$\bar{x}(k+1) = f_0(\bar{x}(k)) + f_1(\bar{x}(k))\,u(k), \tag{3}$$

where $k = 0, 1, .., N$, $\bar{x}(k), f_0(\bar{x}(k)) \in R^n$, $f_1(\bar{x}(k)) \in R^{n\times m}$ and $u(k) \in R^m$, with sampling time T. Define the following performance index

$$J = \frac{1}{2}\bar{x}^T(N) H\bar{x}(N) + \frac{1}{2}\sum_{k=0}^{N-1}\left\{\bar{x}^T(k) Q\bar{x}(k) + u^T(k)Ru(k)\right\}, \tag{4}$$

where the horizon is $t_f = TN$ and $H, Q \geq 0$ and $R > 0$ are matrices of appropriate dimensions. We want to find a suboptimal control sequence which minimizes the performance index (4) subject to the system given by (3). The essence of the following dynamic programming approach in discrete time was proposed originally by Bellman[19],[20],[16]. First, define the next notation:

$$J^*_{N,N} = \frac{1}{2}\bar{x}^T(N) H\bar{x}(N).$$

Notice that this term does not depend of the control law $u(N)$, and it could be called the optimal value of J in the discrete time N. Then, the next step is defined as follows

$$\begin{aligned} J^*_{N-1,N} &= \min_{u(N-1)}\left\{\frac{1}{2}\bar{x}^T(N) H\bar{x}(N) + \frac{1}{2}\bar{x}^T(N-1) Q\bar{x}(N-1) + \frac{1}{2}u^T(N-1)Ru(N-1)\right\} \\ &= \min_{u(N-1)}\left\{J^*_{N,N} + \frac{1}{2}\bar{x}^T(N-1) Q\bar{x}(N-1) + \frac{1}{2}u^T(N-1)Ru(N-1)\right\}. \end{aligned}$$

The value of the state $\bar{x}(N)$ could be calculated using the state space equation given by (3), then:

$$\begin{aligned} J^*_{N-1,N}(\bar{x}(N-1), u(N-1)) = \min_{u(N-1)}&\left\{\frac{1}{2}\left[f_0(\bar{x}(N-1)) + f_1(\bar{x}(N-1))u(N-1)\right]^T H\times\right. \\ &\left[f_0(\bar{x}(N-1)) + f_1(\bar{x}(N-1))u(N-1)\right] \\ &\left.+\frac{1}{2}\bar{x}^T(N-1) Q\bar{x}(N-1) + \frac{1}{2}u^T(N-1)Ru(N-1)\right\}, \end{aligned} \tag{5}$$

in this step, necessary conditions for the optimality of the variational calculus are used to find the minimum of $J_{N-1,N}$. It follows that

$$u^*(N-1) = -\left[f_1^T(\bar{x}(N-1)) Hf_1(\bar{x}(N-1)) + R\right]^{-1} f_1^T(\bar{x}(N-1)) Hf_0(\bar{x}(N-1)), \tag{6}$$

the positive definiteness of the matrix R guarantees that $\left[f_1^T(\bar{x}(N-1)) Hf_1(\bar{x}(N-1)) + R\right]^{-1}$ exists. Observe that the control given by (6) is the local optimal control, because

$$\frac{\partial^2 J_{N-1,N}(\bar{x}(N-1), u(N-1))}{\partial^2 u(N-1)} = R > 0. \tag{7}$$

The existence of a local minimum is guaranteed, because right side in the equation (5) is strongly convex with respect to $u(N-1)$. For the following step $N-2$, note that the value of $u^*(N-1)$ is the optimal value in the step $N-1$, and according with the Bellman optimality principle, this control generates the optimal value for $\bar{x}(N-1)$. Then for this step we have that:

$$\begin{aligned} \widetilde{J}_{N-2,N}(\bar{x}(N-2), u(N-1), u(N-2)) = \min_{u(N-1),u(N-2)}&\left\{\frac{1}{2}\bar{x}(N)^T H\bar{x}(N) + \frac{1}{2}\bar{x}^T(N-1) Q\bar{x}(N-1)\right. \\ &+\frac{1}{2}\bar{x}^T(N-2) Q\bar{x}(N-2) + \frac{1}{2}u^T(N-1)Ru(N-1) \\ &\left.+\frac{1}{2}u^T(N-2)Ru(N-2)\right\}, \end{aligned} \tag{8}$$

Notice that, the term which involves $\bar{x}(N)$, depends of the control $u(N-1)$ and the term with $\bar{x}(N-2)$ depends of the control $u(N-3)$, only the term with $\bar{x}(N-1)$ depends of the control $u(N-2)$. But, the control $u(N-1)$ founded in previous step is optimal and $\bar{x}(N-1)$ is given by the state equation (3), it follows that

$$\begin{aligned} \widetilde{J}_{N-2,N}(\bar{x}(N-2), u(N-2)) = \min_{u(N-2)}&\left\{\frac{1}{2}\bar{x}(N)^T H\bar{x}(N) + \frac{1}{2}\left[f_0(\bar{x}(N-2)) + f_1(\bar{x}(N-2))u(N-2)\right]^T Q\times\right. \\ &\left[f_0(\bar{x}(N-2)) + f_1(\bar{x}(N-2))u(N-2)\right] \\ &+\frac{1}{2}\bar{x}^T(N-2) Q\bar{x}(N-2) + \frac{1}{2}u^T(N-1)Ru(N-1) \\ &\left.+\frac{1}{2}u^T(N-2)Ru(N-2)\right\}. \end{aligned} \tag{9}$$

Now, we use this equation to obtain the suboptimal control $u\left(N-2\right)$ in this step, with this we avoid the obtaining of the type of Riccati equations in discrete domain, which is a very complex problem. However, notice that the equation given by (9) is strongly convex with respect to $u\left(N-2\right)$ and this fact guarantees the existence of a local minimum, however it is only an approximation to optimal value of $u\left(N-2\right)$. We proceed in the usual manner in order to find the suboptimal control $u\left(N-2\right)$:

$$\widetilde{u}(N-2)=-\left[f_1^T\left(\bar{x}\left(N-2\right)\right)Qf_1\left(\bar{x}\left(N-1\right)\right)+R\right]^{-1}f_1^T\left(\bar{x}\left(N-1\right)\right)Qf_0\left(\bar{x}\left(N-1\right)\right).$$

For the general case:

$$\begin{aligned}\widetilde{u}(N-k)&=-\left[f_1^T\left(\bar{x}\left(N-k\right)\right)Ef_1\left(\bar{x}\left(N-k\right)\right)+R\right]^{-1}f_1^T\left(\bar{x}\left(N-k\right)\right)Ef_0\left(\bar{x}\left(N-k\right)\right),\\ \widetilde{J}_{N-k,N}\left(\bar{x}\left(N-k\right),u\left(N-k\right)\right)&=\widetilde{J}_{N-k+1,N}+\frac{1}{2}\left\{\bar{x}^T\left(N-k\right)Q\bar{x}(N-k)+\widetilde{u}^T(N-k)R\widetilde{u}(N-k)\right\},\\ \textbf{for all } k&=2,..,N.\end{aligned}$$

where $E=H$ for $k=1$ and $E=Q$, for $k=2,3,...,N$. Notice that a particular choise for matrices H and Q could give different structures of the suboptimal controller. This suboptimal sequence obtained here guarantees that an approximation for the minimal value (in the local sense) of the performance index (4) is reached, see equation (7).
Now, the SDNLC proposed here, required 513 KB of flash memory of the Pixhawk autopilot of the UAV (it includes the navigation algorithm exposed in Section 4), as the Pixhawk used in the UAV has 2 MB of flash memory, we can conclude that both algorithms (control strategy and navigation routines) are feasible. It is important to say that the SDNLC used here, does not use previous values of control parameters, as the optimal linear control, please see Santos[16], so it reduces the use of flash memory of the autopilot. Respect to the performance of the SDNLC, in Santos[16] a comparative study was conducted in outdoor environment: three control strategies (suboptimal nonlinear control, optimal linear control and Proportional Derivative Control) were compared respect to two factors: energy consumption and error tracking (when circles are considering as reference), see Tables 4 and 5 of Santos[16]. In general, the SDNLC presents a better performance than the others controllers: less energy consumption, less error tracking and less error mean tracking, when 10 experiments were conducted in outdoor environment. Additionally, other advantages of the suboptimal nonlinear control used in this paper, are related to the tuning and its digital implementation of the controller: a preliminary selection of two penalty matrices is required, just like in the Linear Quadratic Regulator (LQR) approach, but avoiding the curse of dimensionality[19] which implies the storage of weight parameters of the controller. In contrast, another kind of nonlinear controllers for UAV, needs a complex tuning of their parameters, see for example, García[21]. As the SDNLC depends of the plant model, then it could affect the closed loop performance of the plant. However, when is it compared with linear controllers, according with the experimental evidence presented in Santos[16], the SDNLC resulted less affected in this point when an outdoor environment was considered. Of course, the SDNLC only represents a local optimal, but as it was mentioned above, the optimal control problem for nonlinear systems for Dynamic Programming approach is a still open problem.

3.2 | Subsystem z -ψ

The model of this 4-rotor vehicle, given by equations (1) and (2), can be divided into small subsystems to simplify the implementation of the control algorithm previously exposed. Firstly, the equations describing the yaw angle ψ and height z are considered; which correspond to vehicle rotation and the translation movement on the z axis. This subsystem is described as follows:

$$\begin{aligned}\ddot{z}\left(t\right)&=u\cos\theta\left(t\right)\cos\phi\left(t\right)-mg\\ \ddot{\psi}\left(t\right)&=\tau_{\psi},\end{aligned}$$

The next step is to discretize these dynamic equations by using of the Euler's approximation as follows:

$$x_{z,\psi}\left(k+1\right)=f_{0,z\psi}\left(x_{z,\psi}\left(k\right)\right)+f_{1,z\psi}\left(\theta\left(k\right),\phi\left(k\right)\right)U\left(k\right), \tag{10}$$

where

$$f_{0,z\psi}(x_{z,\psi}(k)) = \begin{bmatrix} x_{2,z}(k)t_s + x_{1,z}(k) \\ x_{2,z}(k) - gt_s \\ t_s x_{4,\psi}(k) + x_{3,\psi}(k) \\ x_{4,\psi}(k) \end{bmatrix},$$

$$f_{1,z\psi}(k) = \begin{bmatrix} 0 & 0 \\ \frac{t_s}{m}\cos(\theta(k))\cos(\phi(k)) & 0 \\ 0 & 0 \\ 0 & t_s \end{bmatrix},$$

$U(k) = [u(k) \;\; \tau_\psi(k)]^T$ and t_s is the sample time. This model has the form given in (3), and by the algorithm exposed in the previous subsection, we can obtain the suboptimal controllers $\tilde{u}(N_1 - k)$ and $\tau_\psi{}^*\left(N_1 - k\right)$ in the following form:

$$\tilde{u}(N_1 - k) = -F_1\left(N_1 - k\right)\begin{bmatrix} x_{1,z}\left(N_1 - k\right) \\ x_{2,z}\left(N_1 - k\right) \end{bmatrix} + F_2\left(N_1 - k\right),$$

$$\tau_\psi{}^*\left(N_1 - k\right) = -G_1\begin{bmatrix} x_{3,\psi}\left(N_1 - k\right) \\ x_{4,\psi}\left(N_1 - k\right) \end{bmatrix},$$

where

$$F_1\left(N_1 - k\right) = \left[\frac{mt_s\cos\left(\theta\left(N_1 - k\right)\right)\cos\left(\phi\left(N_1 - k\right)\right)}{r_{11}m^2 + E_{22}t_s^2\cos^2\theta\left(N_1 - k\right)\cos^2\phi\left(N_1 - k\right)}\right] \left[E_{21}\left(E_{21}t_s + E_{22}\right)\right],$$

$$F_2\left(N_1 - k\right) = \left[\frac{mt_s\cos\left(\theta\left(N_1 - k\right)\right)\cos\left(\phi\left(N_1 - k\right)\right)}{r_{11}m^2 + E_{22}t_s^2\cos^2\theta\left(N_1 - k\right)\cos^2\phi\left(N_1 - k\right)}\right] E_{22}gt_s,$$

$$G_1 = \left(\frac{t_s}{E_{44}t_s^2 + r_{22}}\right)\left[E_{43}\;\left(E_{43}t_s + E_{44}\right)\right].$$

and

$$E_{ij} = \begin{cases} h_{ij}, & \text{for } k = 1, \\ q_{ij}, & \text{for } k > 1, \end{cases}$$

with

$$E_{z,\psi} = \begin{bmatrix} E_{11} & E_{12} & 0 & 0 \\ E_{21} & E_{22} & 0 & 0 \\ 0 & 0 & E_{33} & E_{34} \\ 0 & 0 & E_{43} & E_{44} \end{bmatrix}.$$

where, h_{ij} and q_{ij} are the elements of the matrices $H_{z\psi} \geq 0$ and $Q_{z\psi} \geq 0$ respectively, which have appropriate dimensions. **That particular choice of matrices H and Q guarantees that the position and velocity of the subsystem z-ψ are involved in the control law.**

3.3 | y and x trajectory control

The analysis of $y - \phi$ subsystem is developed below, that is, the dynamic equations describing the rotation about the x-axis and displacement along the y axis. These related dynamic equation are:

$$\begin{aligned} m\ddot{y} &= u\cos\theta\sin\phi \\ \ddot{\phi} &= \tau_\phi. \end{aligned} \tag{11}$$

The subsystem is discretized as was stated above, then the synthesized discrete sequence control is:

$$\tau_\phi^*(N - k) = -F_\phi x_{y,\phi}^*(N - k),$$

where

$$F_\phi = \frac{1}{e_{44}t_s^2 + R_\phi}\left[e_{41}t_s \;\; e_{41}t_s^2 \;\; e_{43}t_s \;\; e_{43}t_s^2 + e_{44}t_s \right].$$

and $e_{i,j}$ are the elements of,

$$E_{y,\phi} = \begin{bmatrix} e_{11} & 0 & 0 & e_{14} \\ 0 & e_{22} & 0 & 0 \\ 0 & 0 & e_{33} & e_{34} \\ e_{41} & 0 & e_{43} & e_{44} \end{bmatrix}.$$

This matrix includes the elements of $H \geq 0$ and $Q \geq 0$ (as in the previous subsection), which are the parameters used for the optimization technique. Similar steps could made to synthesize the control law τ_θ. For more details see Santos[16].
Lets now with the following subsystem $x - \theta$ described by:

$$\begin{aligned} m\ddot{x} &= -u\sin\theta \\ \ddot{\theta} &= \tau_\theta, \end{aligned}$$

then the corresponding control input τ_θ is given as follows:

$$\tau_\theta^*(N-k) = -F_\theta x_{x,\theta}^*(N-k),$$

where

$$F_\theta = \frac{1}{\tilde{e}_{44}t_s^2 + R_\theta}\left[\tilde{e}_{41}t_s \;\; \tilde{e}_{41}t_s^2 \;\; \tilde{e}_{43}t_s \;\; \tilde{e}_{43}t_s^2 + \tilde{e}_{44}t_s \right].$$

and $\tilde{e}_{i,j}$ are defined as in the previous subsystem, considering a matrix $E_{x,\theta}$ as follows:

$$E_{x,\theta} = \begin{bmatrix} \tilde{e}_{11} & 0 & 0 & \tilde{e}_{14} \\ 0 & \tilde{e}_{22} & 0 & 0 \\ 0 & 0 & \tilde{e}_{33} & \tilde{e}_{43} \\ \tilde{e}_{41} & 0 & \tilde{e}_{43} & \tilde{e}_{44} \end{bmatrix}, E_{x,\theta}(i,j) = \begin{cases} h_{2,ij}, & \text{for } k = 1, \\ q_{x,\theta,ij}, & \text{for } k > 1. \end{cases}$$

.

Observe that an appropriate selection of matrices Q and H in each one of performance index corresponding to each specific subsystem gives the possibility that the control inputs τ_θ^*, τ_ϕ^* and τ_ψ^* are linear. Then, we choose this option in order to obtain simpler controllers, but as it is showed in the next section, also another choice of the penalization matrices, gives a different control structure.

4 | OUTDOOR NAVIGATION

Although Santos[16] presented some experimental results for outdoor navigation of an UAV by using of the SDNLC, the algorithm was programmed without considering an avoiding obstacles task. We think that it is important to evaluate the controller and autopilot performances, when optimizer control strategy and obstacle avoidance routines are together programmed. So, in this section we present the method used to navigate in the presence of obstacles, specifically "trees". Using a laser scanning sensor for the detection of them. The data obtained with this sensor, are analyzed to make the recognition of the environment and the evasion of obstacles if necessary. Table 1 displays the parameters used by the evasion task, planning and repulsion algorithm.

The navigation procedure by using the laser scanning sensor in the presence of trees is as follows:

1. At the beginning the initial position (x_0, y_0, z_0) of the vehicle is known. We also know the final position $(x_{goal}, y_{goal}, z_0)$ where the vehicle must arrive.

2. The takeoff phase is executed

3. The vehicle begins to navigate following a straight line while the path is free of obstacles.

4. If an obstacle is detected at a distance smaller than l_m, an identification procedure is initiated; where the dimensions and the position of the obstacle are estimated.

TABLE 1 Parameters used in the evasion, planning and repulsion algorithm

Parameters	Value
Vehicle mass, (m)	1.5 kg
Measurement health, (H_s)	150 u
Minimum distance to analyze, (l_m)	0.4 m
Maximum distance to analyze, (l_M)	5m
Minimum safety distance, (l_s)	3 m
Maximum distance between, (d_m) consecutive points	0.15 m
Sample time, (T)	0.01 s

5. The data of the obstacle will serve to initiate the process of “repulsion" which consists of moving away in the opposite direction to the position of the tree, at a previously defined speed.
6. When the repulsion phase is over, the evasion process begins, which consists of surrounding the obstacle following a circular path. Note that here the vehicle must decide whether to make the evasion on the right or left side according to the analysis made in step 3.
7. The vehicle continues moving towards the final coordinates and the procedure begins to repeat itself again and again until $x \approx x_{goal}$ and $y \approx y_{goal}$.
8. Finally the landing phase is executed

This procedure can be seen graphically in Figure 1.

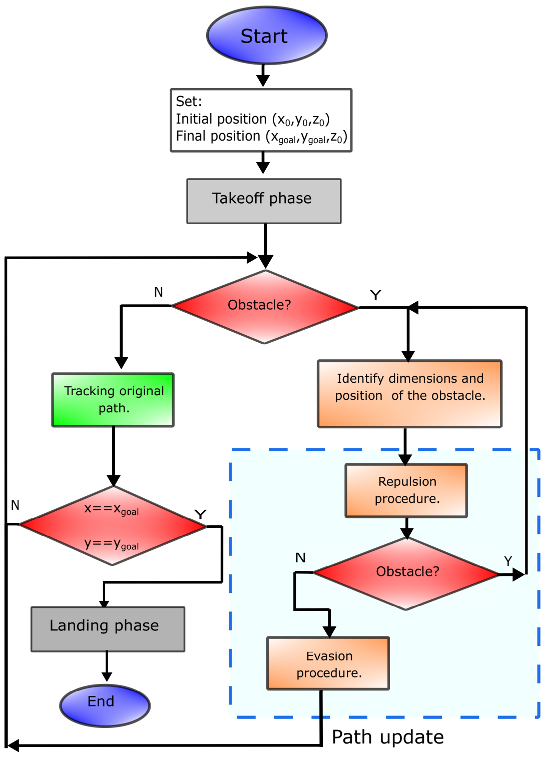

FIGURE 1 Proposed method for the planning of trajectories, detection and evasion of obstacles.

4.1 | Acquisition and processing of data

A relative low cost Light Detection and Ranging (LiDAR) sensor has been chosen to detect the obstacles. Here, we consider necessary to clarify that a vision sensor PixyCAM was tested too, some experimental results (without avoidance obstacles) was reported by Garcia[22]. However, we noticed some disadvantages (respect to the LiDAR) of this vision sensor in the avoidance obstacles, some of them are:

- **High computational complexity. This disadvantage was mentioned by Zuo[23], see Table 2.**
- **In order to improve the performance of the vision sensor is necessary to include more sensors, see for example Courbon[24] or the sensor Intel RealSense Depth Camera D455[25].**
- **Vulnerability to the influence of light environment, this was mentioned by Qiping[26]. For the case of vision sensor PixyCAM, only identifies regular shapes: circles, squares, etc., with high contrast colors. It represents a serious disadvantage, due to the main task here is to avoid trees, and they do not have regular shapes.**
- **A calibration procedure of vision sensor always is necessary. It is necessary, for example, to compensate the light variations, see for example Sala[27], or Gaspar[28], however, an offline training phase is required: images are collected at known discrete points in pose space. For our main task: to detect trees and to avoidance them, it is not feasibility.**

Platform	SensorType	Advantage	Disadvantage	Field
Quadrotor UAV	RGB-D Camera	Low Cost	Poor robustness	Indoor
Helicopter	LADAR	High Precision	High power consumption	Indoor+Outdoor
Quadrotor UAV	Monocular camera +IMU	Light weight	Low Reliability	Indoor+Outdoor
Quadrotor UAV	Binocular camera +IMU	Light weight +High Precision	High computational complexity	Indoor+Outdoor

TABLE 2 Comparison of sensors in UAV obstacle avoidance. Source: Zuo[23].

Table 2 shows a comparison, given by Zuo[23], of vision sensors and LAser Detection And Ranging (LADAR) sensors for aerial vehicles. The LADAR sensor is recommended for use on conventional helicopters (with tail rotor), due to the weight of the sensor, which also implies increasing the energy consumption of the vehicle. However, currently some laser sensors are light, just like the Sweep v1.0 sensor (120 g), so it can be used on a quadrotor UAV.

Additionally, in contrast with the vision sensors, the LiDAR sensors are not easy to be affected by light changes and they can work for 24 hours, see Qiping[26]. As it is explained below, the algorithm to process the data given by the Sweep v1.0, is relative simple when it is compared with the Binocular camera plus IMU, see Tabla 2. Only an Arduino Mega board is needed as signal adapter before to send the detected obstacles to the autopilot, and no extra sensor and calibration procedure are required.

The laser sensor used here (sweep v1.0), rotates at a frequency of 5 Hz, it means that, in one second, it spins 5 times and gets a point cloud representing the near environment. As the resolution of the sensor is 1 degree, follows that the maximum amount of points that can be obtained is 360 per lap. These data are read by the Arduino Mega board, for each data, a three parameters vector is obtained, which contains: distance, angle, and a reliability measurement parameter, or "measurement health". In fact, the read data set $W = \{M_i | i = 1, 2, 3, ..., N\}$ for each completed spin, where $M_i = (l_i, \bar{\theta}_i, H_i)$ is an ordered tuple formed by a distance, an angle and a reliability parameter; $l_i \in \Re^+$ is the distance to the object detected in meters, the angle $\bar{\theta}$ represents the direction in which the measurement was made and it can take values within $(0, 2\pi]$, and H_i corresponds to the health of the measurement i, this parameter lies between $[0, 254]$, a value of H_i closer to 254 indicates a more reliable measurement.

In the figure 2 a coordinate system $b(x_b, y_b, z_b)$ linked to the body is established, note that z_b is perpendicular to x_b and y_b. So, the installation of the laser sensor is done such that it is located in the center of mass of the quadrotor, with its angular deviation value $\bar{\theta} = 0$ aligned with the x_b-axis. The sensor rotates periodically counterclockwise from the axis x_b^+, and its rotation axis coincides with z_b-axis.

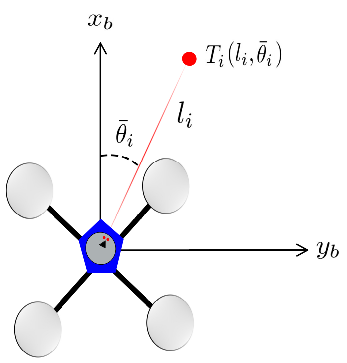

FIGURE 2 Mounting the laser scanning sensor on the experimental platform 8.

The data collected were subject to a filtering procedure in order to reduce the number of operations to be performed by the Arduino Mega and also to remove measurements with large noise levels (taking reference to H_i). **This filtering procedure consists in a thresholding process, which eliminates those measurements that do not meet a health threshold previously fixed and those that are outside of the minimum and maximum limits previously chosen.** Figure 3 shows the distances measured by the sensor in a time of 5 seconds, which is equivalent to just over 3 complete sweeps.

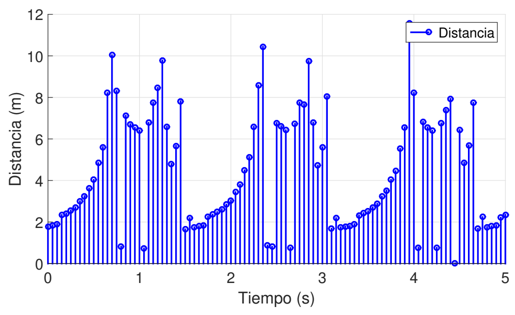

FIGURE 3 Measurements of the laser scanning sensor without any post-treatment

For the thresholding process, define p_k as a pair that belongs to W, which represents the location $(l_k, \bar{\theta}_k)$, of the object identified in the k_{th} reading where $k = \{1, 2, 3, 4, \cdots, n\}$, $n \leq N$ is the number of data that complies with:

$$p_k = \{M_i | H_i > H_s \text{ and } l_m < l_i < l_M\}. \tag{12}$$

In Figure 4 we have discarded all M_i that do not comply with a reliability parameter $H_i > H_s$. The discarded values are those that can cause problems in the estimation of the position and dimensions of the obstacle, due to the noise presents in those measurements.

Finally, Figure 5 shows the measurements that comply with condition given by (12), that is, the distance measurements were subject to a thresholding process in order to reduce the number of data to be processed (similar to the one presented by Yu[11]), this data set p will be used to estimate the dimensions and location of the obstacles.

The filtered measurements are sent via serial communication from Arduino Mega board to the Pixhawk autopilot. But the Pixhawk has also several additional tasks. In this way, communication with the Arduino Mega is done at a frequency

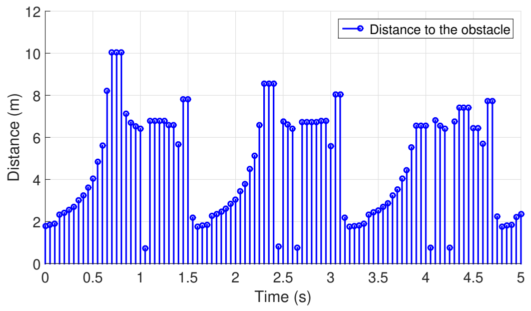

FIGURE 4 Measurements obtained with the laser sensor when $H_i > H_s$.

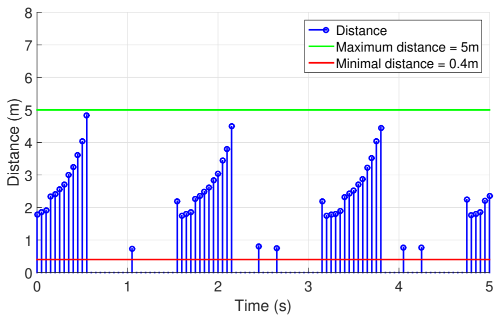

FIGURE 5 Measurements obtained from the laser sensor that were selected for the analysis of the obstacles.

of 50Hz with a maximum duration of 50 microseconds. Exceed this communication rate can produce that the Pixhawk board not be able to read all the data coming from the Arduino, it leads a time delay in the whole system (acquisition, processing, communication and navigation). On average, the estimated delay is about 1.5 seconds. Of course, if there are more obstacles in the environment, the amount of information increases, then more data must be sent to the Pixhawk provoking that this time delay also experience an increase, since that for each detected point tow parameters must be sent to Pixhawk: distance and angle, with a 6-byte format. In view of the recently discussed, experimental tests are a essential issue, and a great challenge, because there is no previous information about the navigation environment. Although we show that there are many advantages in the use of this type of sensor, but we also consider very convenient the integration of this sensor with the Pixhawk autopilot in a single electronic board in order to remove the problem of delay in data transmission. However, it is not a trivial problem.

4.2 | Obstacle detection and collision avoidance

This section addresses the design of the collision avoidance algorithm, which will identify obstacles and generate new positions for the UAV in order to avoid a collision. Define an “obstacle" as one or several laser scan sensor measurements with a magnitude smaller than a safe distance (l_s), with respect to the vehicle.

In order to discriminate between several obstacles measurements during a single sweep, a maximum distance d_m has been established between points read from the same obstacle, ie; $\overline{P_k P_{k-1}}$ is the distance between 2 consecutive measurements and is given by:

$$\overline{P_k P_{k-1}} = \sqrt{(x_{k-1} - x_k)^2 + (y_{k-1} - y_k)^2}, \tag{13}$$

where x_k and y_k are the coordinates of each point identified as an obstacle. Notice that a single detected point is considered as an obstacle.

A point p_k belongs to an obstacle if $\overline{P_k P_{k-1}} < d_m$, where d_m was defined above, otherwise a new set of points is generated and the parameters of the previous obstacle are estimated.

To simplify the data processing in the autopilot, the obstacles were considered as circles of radius r_{obs} centered on (x_{obs}, y_{obs}) with respect to the inertial frame. r_{obs} is estimated for an obstacle with the following expression:

$$r_{obs} = \frac{\sqrt{(x_k - x_1)^2 + (y_k - y_1)^2}}{2}. \tag{14}$$

The coordinates of the center of the obstacle are estimated by projecting l_{obs} at an angle θ_{obs} with respect to the inertial frame, where:

$$l_{obs} = \frac{\sum_1^k l_n}{k} + r_{obs} \tag{15}$$

and

$$\theta_{obs} = \frac{\sum_1^k \bar{\theta}_n}{k}. \tag{16}$$

where k is the number of measurements included in an obstacle. The new position to be obtained by the UAV to keep a safe distance from the object, and it is related to the distance between the UAV and the object when it last one invades the safe area. This new coordinates are define as:

$$x_{new} = x_{pos} + (l_s - l_{obs} - \Delta r)\sin(\theta_{obs} + \pi) \tag{17}$$

and

$$y_{new} = y_{pos} + (l_s - l_{obs} - \Delta r)\cos(\theta_{obs} + \pi). \tag{18}$$

where x_{pos} and y_{pos} are the coordinates of the vehicle relative to the inertial system and Δr is an extra radius to avoid repeating the repulsion process continuously.

4.3 | Path planning

In this section the evasion process is explained, and it is started once the identification and repulsion procedures have been carried out satisfactorily. According to Figure 6 the location of the obstacle can be estimated with:

$$x_{obs} = x_{pos} + l_{obs}\sin\theta_{obs} \tag{19}$$

and

$$y_{obs} = y_{pos} + l_{obs}\cos\theta_{obs}. \tag{20}$$

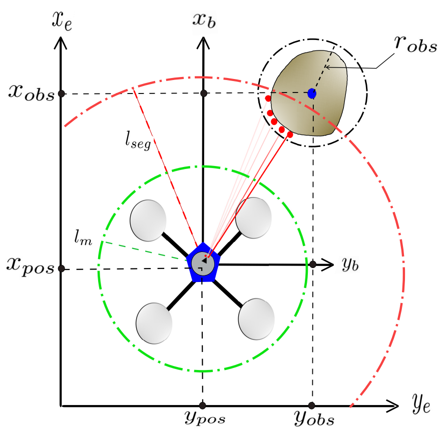

FIGURE 6 Estimating position and radius of an obstacle.

The knowledge of the location of the obstacle, together with the position and attitude of the vehicle allow to compute a new path that leads the UAV to achieve a desired location avoiding collisions.

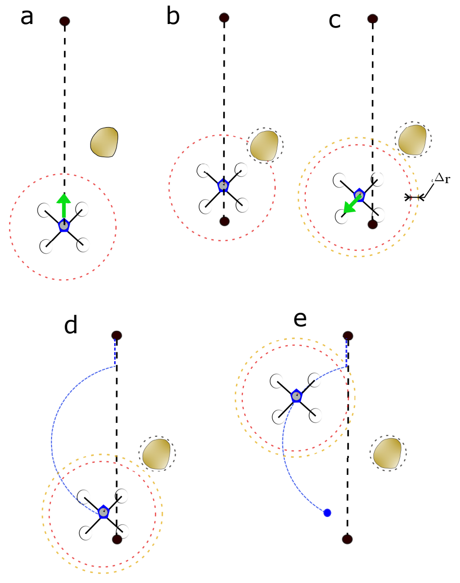

FIGURE 7 Collision avoidance process.

- In Figure 7a, the vehicle starts from (x_0, y_0) towards the defined end point (x_{goal}, y_{goal}), *i.e.*, it begins with the tracking of the original path (dotted black line), with a constant altitude while the sweep sensor performs periodic readings looking for obstacles.
- Figure 7b, if an obstacle is identified during the original path, the UAV computes a new trajectory towards a safe position.
- Figure 7c, the vehicle navigates to the position computed with equations 17 and 18.
- Figure 7d, once the UAV arrived to the position (x_{new}, y_{new}), an arc segment trajectory is computed (blue dotted line) which will surround the obstacle at a $d_s + \Delta r$ distance from the (x_{obs}, y_{obs}) point.
- Figure 7e, the vehicle tracks the new circular path and it will conclude until the intersection with the original path occurs.

While the obstacle is evaded, the repulsion algorithm continues active to avoid collisions with objects that are on the new trajectory. The above procedure is shown in the flow diagram in the Figure 1.

5 | HARDWARE

The experimental platform is shown in Figure 8, it was built using a 550-millimeter carbon fiber quadcopter kit, the quadcopter has an “X" configuration, the motors used are 920 kV. The flight controller is a Pixhawk of 3D Robotics. This autopilot has a high performance, some important features are: it has a 32-bit processor STM32F427 with FPU, its main clock works at 168 Mhz, it has 256kb of RAM and 2mb of flash memory. It has two gyroscopes and two accelerometers, it has a barometer to estimate the height. It also has different interfaces, such as 5 serial ports, 2 CAN ports, Futaba and Spektrum ports, PPM signal input, PWM signal output ports for controlling motors, I2C ports, SPI and two ADC ports[29]. An external Ublox NEO-M8N GPS, has been installed which has an accuracy in the range of 0.6m and 0.9m, and has internally build magnetometer to estimate the guidance of the aircraft.

The batteries used are LiPo technology with a capacity of 4000mA and a discharge rate of 35C. For remote communication with the aircraft, a 7-channel Futaba radio was used.

The laser sensor used to detect obstacles is the “Sweep $v1$ 360- Laser Scanner" wich is a scanning LiDAR sensor designed to bring powerful 360 degrees sensing capabilities. Other important features are: It has a range of 40m, 360 degree field of view, 2 to $10Hz$ of rotation frequency. $5v$ DC input, and 400 mA of power consumption[30]. An Arduino Mega was used to communicate this sensor with the Pixhawk autopilot through serial communication configured at a speed of 115200 Baud. The hardware setup can be observed more clearly in Figure 9.

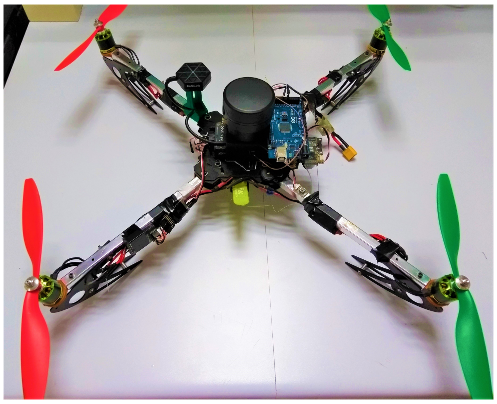

FIGURE 8 Experimental platform used for navigation in outdoor environments.

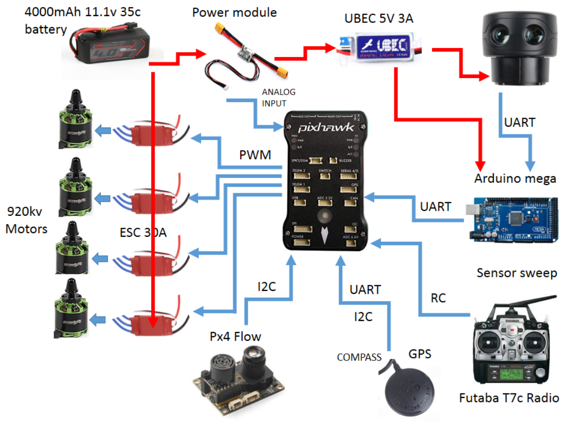

FIGURE 9 Overview of the proposed hardware

6 | EXPERIMENTAL RESULTS

This section presents the results obtained by using the algorithm presented above. This algorithm was implemented in the experimental platform displayed in the Figure 8. Translational position measurements were obtained by using a GPS.

The control technique used to track the path was carried out with the sub-optimal finite-horizon non-linear control shown in Section 3. The aerial mission performed by the vehicle is divided into 3 phases:

1. Takeoff: This phase was carried out manually, that is, the pilot sent the desired position z_p using the radio control.
2. Navigation: The vehicle was settled completely autonomous with the fixed altitude z_p and with the task of moving in a straight line on the x-axis, it means, moving up to $(x_{goal} = 20, y_{goal} = 0)$. The algorithm previously described allows to evade the trees that the UAV detected during its path.

3. Landing: The pilot take control again of the aircraft and lands.

Figure 10, shows the first 3 state variables x, y and z which correspond to the position of the vehicle with respect to the earth coordinates. It also shows the desired values which were modified by the algorithm of planning and collision avoidance since initially the vehicle only had to track a straight line on the x-axis.

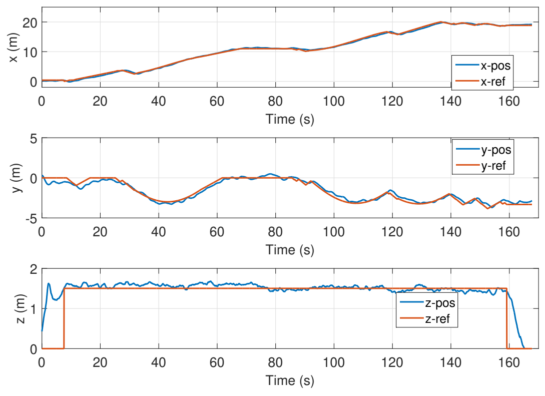

FIGURE 10 UAV position on the x, y and z axes

Figure 11 shows the variables ϕ, θ and ψ which represent the angular position of the vehicle. These variables show the attitude of the vehicle during navigation phase, notice that the yaw angle (ψ) remained constant close to zero.

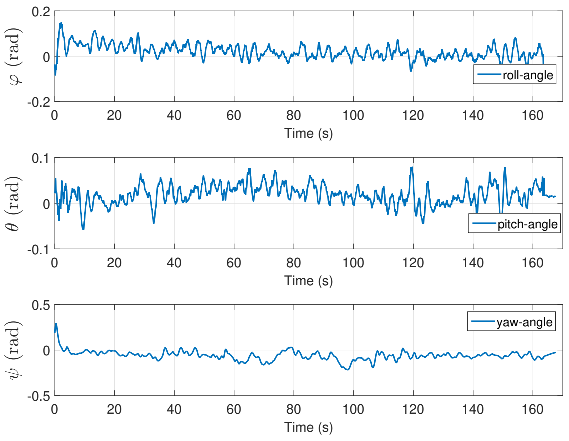

FIGURE 11 Attitude signal, roll pitch and yaw

Figure 12 displays the translational velocity $\dot{x}$, $\dot{y}$ and $\dot{z}$ experienced by the flying robot, we observe that these velocities are smaller than $1m/s$. The position in z had to remain constant, which means that the speed in altitude is very close to zero. Furthermore, is easy to notice that the repulsion maneuvers were executed at $t = 30\,s$ and $t = 120\,s$ in $\dot{x}$, when the vehicle was moving in a straight line forward. However, when detecting an obstacle the UAV had to move back immediately, and a change in the direction of speed occurred.

Figure 13 gives the tracking errors during navigation phase. These variables show the difference between the trajectory reference and the UAV position measured by the GPS. The average error for the x-position is $0.2330\,m$, while for y-position is

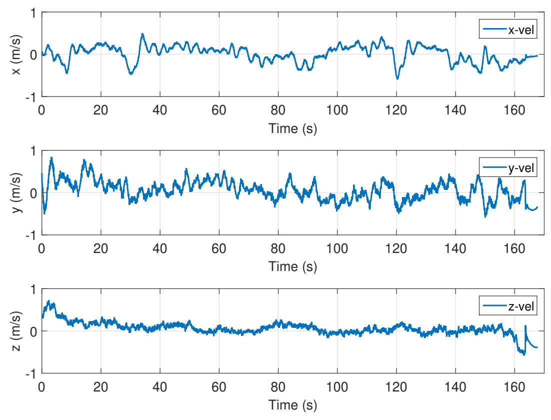

FIGURE 12 Speed responses in the vehicle during navigation

0.2612 m, which we consider to be acceptable with respect to the inherent measurement error of the GPS presented during the experiments. The error in z is even lower and it is just 0.062 m, due to the fact that the altitude reference is constant. These averages were obtained with the following formula: $Average\ error_{x,y,z} = \frac{\sum_1^N |error_{x,y,z}(k)|}{N}$, where k is the current sample and N is the total number of samples of the signal. **Signal errors of the trajectory tracking task illustrate the effectiveness of the scheme combining the optimal non linear control strategy, the laser sensor by detecting obstacles and the obstacle avoidance algorithm. In fact, the bounds of trajectory tracking error related with the real-time experiments are inside of the established limits of the GPS precision (0.6 m to 0.9 m), consequently this experiment could be consider successfully.**

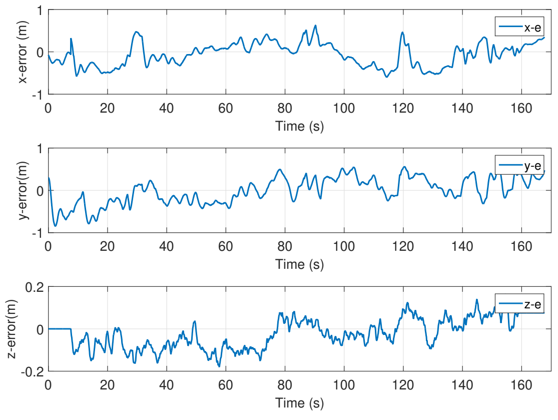

FIGURE 13 Tracking error signals on the x, y and z axes, during navigation

Figure 14 depicts the control signals computed during the aerial mission, these signals were presented in values of %PWM levels in order to see how large or small is the signal that the speed controllers are sending to 3-phase motors. It can be see that the control signal u_z which corresponds to collective throttle, increases little by little. The input τ_ψ remains close to zero, because the set-point of this angle was set to zero.

Finally, Figure 15 displays the 3 dimensional view of the flying robot performing the navigation and collision avoidance tasks, which gives a more clear idea of the behavior of the vehicle throughout the experiment. The original desired trajectory is represented by the green dotted line, which serves as a flight path reference for the aerial vehicle. However, as it detected some obstacles, this reference was updated automatically. In cyan and blue colors we can see the updated reference and the

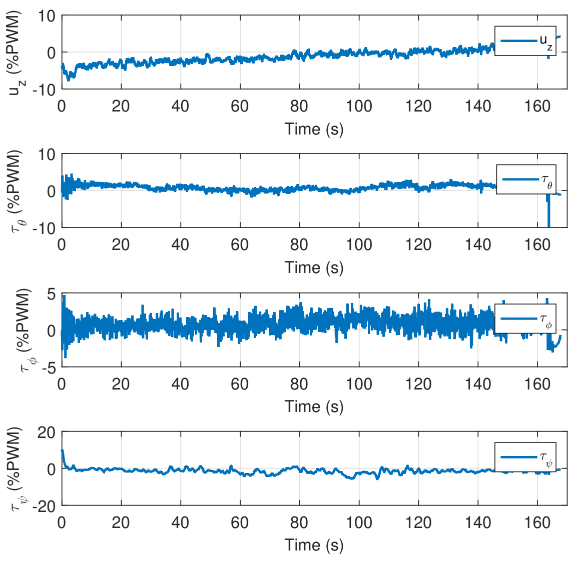

FIGURE 14 Controllers signals during the tracking trajectory

measured real-time position respectively. It is also possible to observe that the vehicle reaches approximately the final reference $x_{goal} = 20\,m$ but not in the y-axis, this was due to the fact that in the final position $[x_{goal}, y_{goal}]$ there were obstacles and it can be seen how the vehicle tries to reach the this final position but the avoidance collision algorithm does not allow it.

According with Figure 15, the first detected obstacle by the laser sensor was in the coordinates (x,y,z)=(5.6, 1.7, 1), observe that firstly, the vehicle done a repulsive action and then the obstacle avoidance algorithm plans the trajectory (a circle) in order to avoid that first tree. However, analyzing Figures 11, 12 and 13, where the attitude, speeds $\dot{x}, \dot{y}, \dot{z}$ and tracking error signals are respectively represented, we can observe that those signal do not experienced any abrupt change during these phases of repulsion and evasion and this performance feature is caused by the proposed control law, and the same behavior is obtained for any detected tree. Consequently, the four control signals, do not present sudden changes in the PWM levels, despite the time delays present in the control loop, see the Figure 14. A video of the complete task can be seen in https://youtu.be/MvyI2OuHZEY.

7 | CONCLUSIONS AND FUTURE WORK

The main contributions of this paper are, the outdoor navigation of a Quadrotor helicopter in presence of trees using a laser scanning sensor, which was mounted on the aerial vehicle, trajectory planning and collision avoidance algorithms were proposed with a geometric approach which allow to successfully perform the tasks of repulsion and evasion. The proposal is based on a synthesized suboptimal nonlinear control sequence for finite horizon, allowing to penalize the energy consumption for the vehicle. The obtained results were satisfactory in comparison to the few works reported in the literature, because most of them, show only simulation results. In contrast, our results were presented in real-time, using an experimental platform, see Figure 8. This technological challenge is non trivial to solve, since the system becomes complex as more sensors are added. **Please notice that Zuo[23] exposed that one tendency in obstacles avoidance in UAVs is to optimize the memory occupation for process in real time. In our proposal, this tendency was achieved by implementing a control strategy that optimize energy, together with avoidance obstacles and trajectory planning algorithms, all them with the philosophy of be simple, whit reduced memory usage, and using a single sensor to detect obstacles. Emphasizing that the proposed algorithm was tested for the avoidance of trees at their lower part, nevertheless, as a future work, the algorithm will be applied for any obstacle that can be detected by the Sweep laser sensor**.

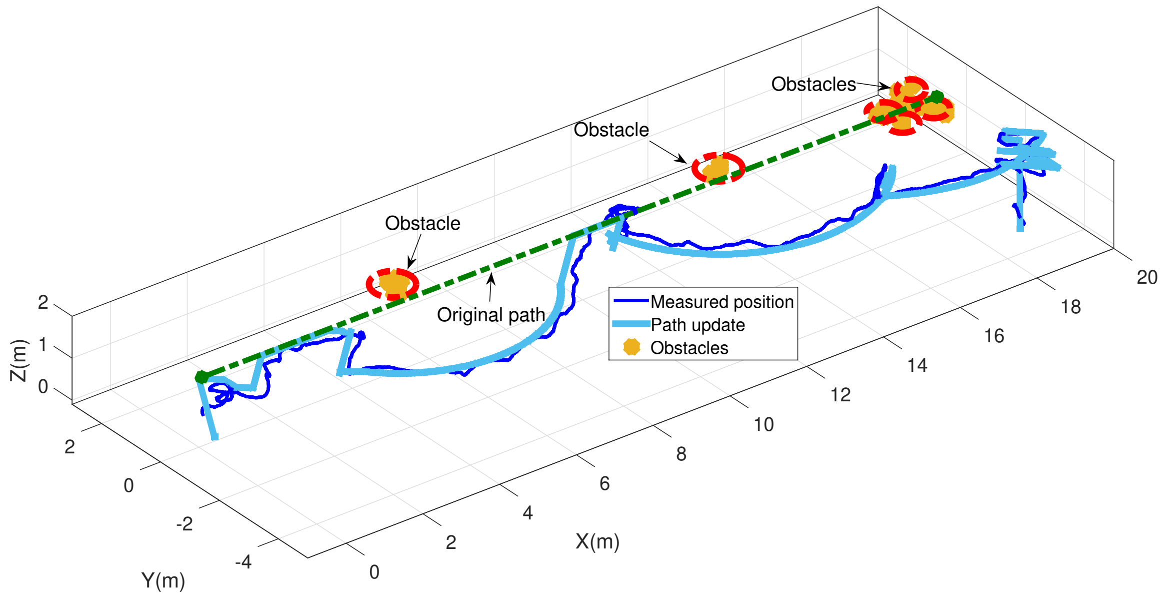

FIGURE 15 Planning of trajectories in outdoor environments and collision avoidance

8 | ACKNOWLEDGMENT

The authors would like to thank José Manuel López Rivera, a student at the National Polytechnic Institute, for his great support during the experimental tests.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available from the corresponding author upon reasonable request.

References

1. Ait-Jellal R, Zell A. Outdoor obstacle avoidance based on hybrid visual stereo SLAM for an autonomous quadrotor MAV. In: IEEE. ; 2017: 1–8.

2. Barry AJ, Tedrake R. Pushbroom stereo for high-speed navigation in cluttered environments. In: IEEE. ; 2015: 3046–3052.

3. Campos-Macías GGDALRdlGR, Parra-Vilchis JI. A Hybrid Method for Online Trajectory Planning of Mobile Robots in Cluttered Environments. *IEEE Robotics and Automation Letters* 2017; 2(2): 935–942.

4. Anis H, Fadhillah AHI, Darma S, Soekirno S. Automatic Quadcopter Control Avoiding Obstacle Using Camera with Integrated Ultrasonic Sensor. In: . 1011. IOP Publishing. ; 2018: 012046.

5. Bugayong A, Ramos Jr M. Obstacle Detection using Binocular Stereo Vision in Trajectory Planning for Quadcopter Navigation. In: . 976. IOP Publishing. ; 2018: 012008.

6. Rahman MF, Sasongko RA. Obstacle avoidance for quadcopter using ultrasonic sensor. In: . 1005. IOP Publishing. ; 2018: 012037.

7. Mohanta J, Parhi DR, Mohanty S, Keshari A. A control scheme for navigation and obstacle avoidance of autonomous flying agent. *Arabian Journal for Science and Engineering* 2018; 43(3): 1395–1407.

8. Thanh HLNN, Phi NN, Hong SK. Simple nonlinear control of quadcopter for collision avoidance based on geometric approach in static environment. *International Journal of Advanced Robotic Systems* 2018; 15(2): 1729881418767575.

9. Yang X, Luo H, Wu Y, Gao Y, Liao C, Cheng KT. Reactive obstacle avoidance of monocular quadrotors with online adapted depth prediction network. *Neurocomputing* 2019; 325: 142–158.

10. Mendoza-Soto JL, Alvarez-Icaza L, Rodríguez-Cortés H. Constrained generalized predictive control for obstacle avoidance in a quadcopter. *Robotica* 2018; 36(9): 1363–1385.

11. Yu Y, Wu Z, Cao Z, Pang L, Ren L, Zhou C. A laser-based multi-robot collision avoidance approach in unknown environments. *International Journal of Advanced Robotic Systems* 2018; 15(1): 1729881418759107.

12. Máthé K, Buşoniu L. Vision and control for UAVs: A survey of general methods and of inexpensive platforms for infrastructure inspection. *Sensors* 2015; 15(7): 14887–14916.

13. Lozano R. *Unmanned aerial vehicles: Embedded control*. John Wiley & Sons . 2013.

14. Castillo P, García P, Lozano R, Albertos P. Modelado y estabilización de un helicóptero con cuatro rotores. *Revista Iberoamericana de Automática e Informática Industrial RIAI* 2007; 4(1): 41–57.

15. Das A, Subbarao K, Lewis F. Dynamic inversion with zero-dynamics stabilisation for quadrotor control. *IET control theory & applications* 2009; 3(3): 303–314.

16. Santos-Sánchez O, García O, Romero H, Salazar S, Lozano R. Finite horizon nonlinear optimal control for a quadrotor: Experimental results. *Optimal Control Applications and Methods* 2021; 42(1): 54–80.

17. Komaee A. An inverse optimal approach to design of feedback control: Exploring analytical solutions for the Hamilton-Jacobi-Bellman equation. *Optimal Control Applications and Methods* 2021; 42(2): 469–485.

18. Wang FY, Zhang H, Liu D. Adaptive dynamic programming: An introduction. *IEEE computational intelligence magazine* 2009; 4(2): 39–47.

19. Bellman R, Kalaba RE. *Dynamic programming and modern control theory*. 81. Citeseer . 1965.

20. Kirk DE. *Optimal control theory: an introduction*. Courier Corporation . 2004.

21. García O, Ordaz P, Santos-Sánchez OJ, Salazar S, Lozano R. Backstepping and robust control for a quadrotor in outdoors environments: An experimental approach. *IEEE Access* 2019; 7: 40636–40648.

22. Garcia O, Flores D, Santos O, Romero H, Salazar S, Lozano R. Autonomous take-off and landing on a colored platform. In: IEEE. ; 2017: 877–884.

23. Zuo K, Cheng X, Zhang H. Overview of Obstacle Avoidance Algorithms for UAV Environment Awareness. In: . 1865. IOP Publishing. ; 2021: 042002.

24. Courbon J, Mezouar Y, Guénard N, Martinet P. Vision-based navigation of unmanned aerial vehicles. *Control Engineering Practice* 2010; 18(7): 789–799.

25. FRAMOS . `https://www.framos.com/en/products-solutions`. December 2021.

26. Chen Q, Xie Y, Guo S, Bai J, Shu Q. Sensing system of environmental perception technologies for driverless vehicle: A review of state of the art and challenges. *Sensors and Actuators A: Physical* 2021: 112566.

27. Sala P, Sim R, Shokoufandeh A, Dickinson S. Landmark selection for vision-based navigation. *IEEE Transactions on robotics* 2006; 22(2): 334–349.

28. Gaspar J, Winters N, Santos-Victor J. Vision-based navigation and environmental representations with an omnidirectional camera. *IEEE Transactions on robotics and automation* 2000; 16(6): 890–898.

29. Pixhawk . `https://docs.px4.io/en/ flight_controller/pixhawk.html`. Updated: 2018-07-05 23:26:14.

30. Sweep V1 . `https:https://www.kickstarter. com/projects/scanse/sweep-scanning-li dar ?lang=es`. Updated: 2019.

□