sensors

MDPI

Article

Error Reduction in Vision-Based Multirotor Landing System

Juan Pedro Llerena Caña *, Jesús García Herrero * and José Manuel Molina López

Applied Artificial Intelligence Group (GIAA), Carlos III University of Madrid, 28270 Madrid, Spain; molina@ia.uc3m.es

* Correspondence: jllerena@inf.uc3m.es (J.P.L.C.); jgherrer@inf.uc3m.es (J.G.H.)

Abstract: New applications are continuously appearing with drones as protagonists, but all of them share an essential critical maneuver—landing. New application requirements have led the study of novel landing strategies, in which vision systems have played and continue to play a key role. Generally, the new applications use the control and navigation systems embedded in the aircraft. However, the internal dynamics of these systems, initially focused on other tasks such as the smoothing trajectories between different waypoints, can trigger undesired behaviors. In this paper, we propose a landing system based on monocular vision and navigation information to estimate the helipad global position. In addition, the global estimation system includes a position error correction module by cylinder space transformation and a filtering system with a sliding window. To conclude, the landing system is evaluated with three quality metrics, showing how the proposed correction system together with stationary filtering improves the raw landing system.

Keywords: UAV; autonomous landing; filtering; computer vision; helipad context; global position; navigation system; SITL

check for updates

Citation: Llerena Caña, J.P.; García Herrero, J.; Molina López, J.M. Error Reduction in Vision-Based Multirotor Landing System. *Sensors* **2022**, *22*, 3625. https://doi.org/10.3390/s22103625

Academic Editor: Sindhuja Sankaran

Received: 30 March 2022
Accepted: 6 May 2022
Published: 10 May 2022

1. Introduction

The growing demand for drone applications motivates the study of the support technologies for this type of small and powerful unmanned aerial vehicle (UAV). However, all new applications share an essential and critical maneuver—landing.

Generally, work in the literature about landing maneuvers, both for fixed and rotary wing UAVs, focuses on control strategies [1–3]. All of them require access to the internal vehicle states, the actuators or specific modes of the control or navigation system.

Under the precision landing concept is included all the solutions that approach this maneuver in an autonomous or supervised way, independent of the techniques and sensors used to estimate the vehicle states such as position or orientation, as well as its corresponding velocities and accelerations. The landing maneuver can be included within the navigation system where two main groups can be distinguished, outdoor and indoor navigation. Generally, outdoor navigation is based on the Global Navigation Satellite System (GNSS), but practically all current systems use fusion techniques that allow the integration of different strategies for estimating one or more vehicles' states, which is necessary for the control and/or navigation system. Some of the most common cases in navigation systems are the use of barometers and/or sonars to improve the accuracy of the altitude provided by GNSS, or the use of small zenith cameras to determine small horizontal displacements [4]. Other specific navigation techniques such as visual odometry or visual Simultaneous Location And Mapping (SLAM) [5] are beginning to be used in indoor navigation.

Thus, other landing works focus on improving the accuracy of instrument systems such as [6–8] or even context information such as safe landing zones, as in work by Shah Alam, Md et al. shown in work [9]. Some commercial solutions focus on the use of beacons that indicate the landing region, as can be seen in the work of J. Janousek and P. Marcon [7] using a commercial infrared light beacon. This type of system includes an

external controller to minimize the pixel error between the region of interest (ROI), defined by the centroid of the infrared area and the reference in the image plane, generally located at the center of the image plane. These strategies need to access internal vehicle states such as velocity or acceleration to correct the error.

In terms of context information, vision systems proved to be efficient to identify ROIs. In addition, knowing the landing context, specifically where or how the helipad is where the UAV must land, can help to improve the landing maneuver.

Developing strategies to identify and understand the context information of the aircraft allows providing the systems with greater autonomy. In the survey of autonomous landing techniques for UAVs by Alvika Gautam et al. [1], the authors describe the relationship between sensors/navigation systems and aircraft control modules, paying particular attention to vision landing techniques, generally responsible for recognizing and estimate the helipad position.

Some civil and commercial UAVs, such as certain DJI models [10], are beginning to integrate vision-based precision landing systems. In the work of Yoakum and Carreta [11], the authors conduct a study of the precision landing system of a DJI Mavic Pro, proving the aircraft and the integrated landing system meet specific accuracy requirements to use a specific wireless charging station.

Generally, vision systems for landings focus on identifying the landing area, either by means of context information of the helipad pattern or by the terrain conditions. The work of Mittal et al. [12] is an example of the identification of landing area conditions, where terrain slope is estimated to verify the feasibility of a UAV to land in urban search and rescue.

Regarding pattern recognition landing systems, works such as [13,14], among others [15–18], focus on finding the position using known patterns by Perspective-n-Point (PnP) algorithms [19]. Patterns such as Aruco [20], charuco, or new fractal patterns such as [21] or the deep learning trend You Only Look Once (YOLO) [22] try to improve the pattern pose estimation and prove to be widespread systems in the literature. In the literature and throughout this paper, an object "pose" means a set of position and heading of a specific reference system.

On the other hand, the emergence of open-source flight controllers such as Pixhawk [23], together with specific communication protocols such as Micro Air Vehicle Link (MAVLink) [24] and multiplatform APIs such as MavSDK [25], help to develop new applications and research new landing strategies.

PX4 [26] autopilot set up as a rotary wing vehicle has a planification system that allows dynamically smoothing the trajectories between different positions [27]. Specifically, it smooths the trajectories between consecutive waypoints by rounding the turns with radii over the waypoints and increasing or decreasing the drone speed when approaching or moving away from a waypoint [28–30]. The guidance algorithm integrated in PX4 is the L_1 algorithm introduced by Park et al. [31] under the linear approach. When forcing a new target position while the vehicle is navigating between two positions, the system changes target and tends to reach the newly added location by smoothing its current trajectory, as shown in works such as Stateczny et al. [32]. This behavior, when repeated with a certain frequency to include new waypoints referring to the same position, but with a certain noise, produces a spin effect on the aircraft that we call "inter-waypoint noise spin effect".

In this work, we propose using the aircraft guidance system without downing the controller level for a widespread implementation of the precision landing system, contributing to UAV air safety and helping the emergence of new applications.

We propose a new contribution with respect to classical geolocation landing strategies based on global positioning by decoupling the landing in two phases, first reaching the target coordinates and then activating the landing mode, descending vertically with a constant descent speed α.

Our proposed landing strategy seeks to descend quickly when the target is found and to smooth its descent as the aircraft approaches, without having to adjust the controller

parameters. This idea attempts to improve the image resolution quickly to improve the position estimation. The difference with respect to other works is that simultaneously descending and adjusting the positioning relies on the variable waypoint altitude adjustment without accessing the controller, so that without changing the internal descent controller of the PX4 [26] or adding new external control laws, the strategy allows smoothing the descent when approaching the target. This strategy allows taking further steps in the final phase of the approximation, improving the final estimate, and ensuring the stability of the system provided by the manufacturer. For this, we propose a function to modulate the default behavior of the PX4 controller which seeks a stationary descent at a constant speed.

In addition, this paper models the error of a precision landing system using a monocular vision system, context information from the helipad pattern, and the internal navigation system of the PX4 flight controller.

The vision system error modeling allows a fine calibration of helipad localization. This correction, together with a stationary filtering of the estimates by sliding time window and variable adjustment of the descent height, allow to reduce the spin effect produced by the estimation error of a vision system when integrated with the L_1 navigation algorithm, without access to the internal controller parameters or relative states of the aircraft that may require re-programming of the on-board computer.

To sum up, this paper presents two main contributions in precision landing by vision-based global position: the continuous adjustment of the approach to descent trajectory, and the improvement of the position by vision through systematic error adjustment and filtering.

The proposal strategy is evaluated after including an error model correction as well as different sliding time window filters. The work is developed on a hyper-realistic software in the loop (SITL) [33] simulation system with the PX4 flight controller and the AirSim [34] simulator.

Finally, the results of the study show the estimation error analysis and filtering of the estimates with sliding time window filters, minimizing the inter-waypoint noise spin effect generated by noisy waypoint transitions and improving three quality metrics of landing time, trajectory landing length, and landing accuracy without additional control law, enabling the use of the aircraft's guidance system as an alternative for the deployment of precision landing technology.

This paper is organized as follows: Section 2 shows the problem formulation of helipad spatial estimation by monocular computer vision. Section 3 describes the landing strategy proposal and the global estimation module. The correction module design, the analysis of the complete landing system, and a description of the test environment can be found in Section 4. Finally, the conclusions are presented in Section 5.

2. Problem Formulation

We consider the problem of a UAV landing on a certain landing pad using its internal autopilot waypoint guidance system and a monocular vision system with gimbal integrated in the UAV.

Figure 1 shows the set of reference frames, where the superscripts $\{t, ph, c, z, g, b, n, e$ and $g\}$ correspond to the reference frames of the helipad (target), pinhole camera model, camera, gimbal socket for the camera, gimbal, body, North-East-Down (NED), Earth-Centered Earth-Fixed (ECEF), and global.

In this way, any given point p_t^t expressed in a flat pattern reference frame $\{t\}$ and the homogeneous transformation ${}^nT_t$ between the landing pad reference frame to the NED referent frame $\{n\}$ can be expressed in the global reference frame $\{g\}$ system p_t^g applying a set of transformations shown in Equations (1) and (2).

$$p_t^n = \overbrace{{}^nT_b \cdot {}^bT_g \cdot {}^gT_z \cdot {}^zT_c \cdot {}^cT_t}^{{}^nT_t} \cdot p_t^t \tag{1}$$

$$p_t^g = {}^gF_e\left\{ {}^eF_n\left[p_t^n, {}^eF_g(p_{Ref}^g) \right] \right\} \tag{2}$$

where superscript $\{j\}$ over point p_i^j means the reference frame system, and the subscript $i = \{t, Ref\}$ denotes the name of the point (target and reference). ${}^jT_i$ means the homogeneous transformation between reference frame i and j. On the other hand, ${}^jF_i$ refers to nonlinear transformations between reference frame i and j. p_{Ref}^g indicates the global position of the body (UAV) as a global reference point.

The set of reference frame systems involved in the transformations are shown in Figure 1 and denoted as: helipad (target) $\{t\}$, pinhole camera model $\{ph\}$, camera $\{c\}$, gimbal socket for the camera $\{z\}$, gimbal $\{g\}$, body $\{b\}$, North-East-Down (NED) $\{n\}$, Earth-Centered Earth-Fixed (ECEF) $\{e\}$, and global $\{g\}$.

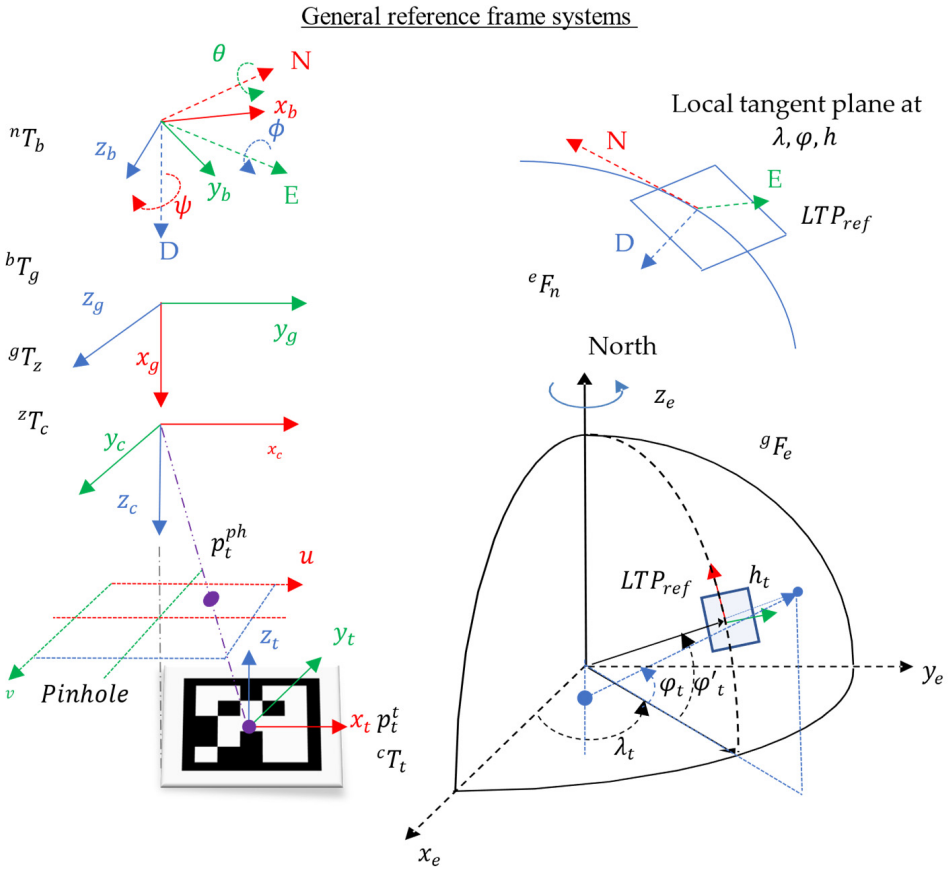


Figure 1. General reference frame systems. Reference frames bottom-left to up: helipad (target), pinhole camera model (image plane), camera, gimbal socket, body, NED. Reference frames right up to down: NED, ECEF, Global.

2.1. Pattern (Helipad) Detection

We consider as the helipad a reference pattern defined by an Aruco pattern [35] with a certain number of bits, as part of a library B. As shown in the paper [20], the system identifies candidate square regions as Aruco markers, then encodes these regions and compares them to the pattern dictionary as desired.

The full process can be divided into the following steps:

- **Image conversion:** Obtain an RGB image and transform it to grayscale.
- **Edge extraction**: We understand as edge an intensity change boundary, some classical algorithms are Canny [36] and Sobel [37].
- **Contour extraction**: We understand a contour as a curve of points without gaps or jumps. Therefore, the objective is to identify if the edges found represent contours. An example of simple contour extraction can be given by a binarized image of an

object whose outer contour can be extracted by subtracting the original binarized-dilated image from the original binarized image. To check if closed regions appear, a segmentation by connected components would provide us candidate regions of interest (ROI) as a result.

- **Contour filtering**: Only show rectangular regions.
- **Removing ROI perspective distortion**: For this it is necessary to find the general plane $\mathcal{P}^2$ projective transformation $h : \mathcal{P}^2 \rightarrow \mathcal{P}^2 \big| \, h(m) = m' = mH$, where m is a point in a plane. $H^{3\times3}$ is a non-singular matrix where $m\prime$ is the linear transformation H of m. The transformation H is biunivocal and homogeneous, in other words, a point over a plane is a unique point over another plane and $kH \big| \, k \in \mathbb{R}$ and $k \neq 0$ is also the solution. This condition allows dividing the matrix H by the element h_{33}, decreasing the dimension of terms to identify from 9 to 8. The correspondence between points $(x_i, y_i) \leftrightarrow (x_i', y_i')$ can be expressed in matrix form as $b_i = A_i h$, and their relationship is expressed as Equation (3) (more details in [38]). Knowing n pairs of points, the system of $2n$ equations and 8 unknowns is established as $b = Ah$, where $A = [A_1, A_2, \ldots, A_n]^T$, $b = [b_1, b_2, \ldots, b_n]^T$, and $h^{3\times3}$ matrix as $h_{33} = 1$. For $n = 4$, the direct solution $h = A^{-1}b$; if $n > 4$ the system is overdetermined and least squares can be applied, $h = \left[A^T A\right]^{-1} A^T b$. For cases where $h_{33} = 0$ refer to [38].

$$\begin{gathered} x' = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}} \\ y' = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}} \\ A = \begin{bmatrix} x & y & 1 & 0 & 0 & 0 & -x\prime x & -x\prime y \\ 0 & 0 & 0 & x & y & 1 & -y\prime x & -y\prime y \end{bmatrix}; b = \begin{bmatrix} x' \\ y' \end{bmatrix}; h = [h_{11}, h_{12}, h_{13}, h_{21}, \ldots h_{32}]^T \end{gathered} \tag{3}$$

- **Pattern library matching check**: The binary code of the ROI is extracted, superimposing on the binarized and perspective-corrected image a grid of the same cell size as the searched one. Each grid cell receives a binary value according to if the corresponding color is black (zero) or white (one). The Hamming coding algorithm (ref) is applied to the extracted code to eliminate false negatives. This resulting code is compared with the selected pattern dictionary, filtering the regions identified as markers and belonging to the pattern dictionary from other regions. In addition, this step provides information about the marker id if the ROI belongs to the library.

2.2. Helipad Pose Estimation

For pose estimation, the Perspective-n-Point (PnP) problem [35] is formulated where the objective is to minimize the reprojection error Equation (7) of 3D points in the image plane $\{ph\}$. This problem is closely linked to a calibrated system, since it requires a camera model, pinhole, and a pattern that allows to relate identified features of an image with features of the pattern.

Given a point $p_t^c \in \mathbb{R}^3$ belonging to the knowing pattern located in real-world 3D space and expressed in the camera reference frame $\{c\}$, it can be expressed in the image camera plane reference frame $\{ph\}$ as $p_t^{ph} \in \mathbb{R}^2$. The relationship between the two reference frames is provided by the pinhole camera model in Equation (4).

$$s p_t^{ph} = A p_t^c \tag{4}$$

where s is a scale factor and A intrinsic camera matrix [17]. The internal matrix A is composed of the focal distances (f_x, f_y) and the principal points (c_x, c_y). The pinhole model can be improved with radial, tangential, or prism distortion corrections, adding n set of k_i parameters to the model [39–41]. The set of internal parameters of the camera model can be expressed by the vector $\delta = (f_x, f_y, c_x, c_y, k_1, \ldots, k_n)$.

If the point is expressed in coordinates of the pattern reference frame p_t^t, there exists an extrinsic homogenous transformation ${}^cT_t$ to relate the reference frame of the pattern to the camera reference frame Equation (5) is used.

$$p_t^c = {}^cT_t p_t^t \tag{5}$$

where the transformation ${}^cT_t = [{}^cR_t | v_t^c]$ is a rototranslation composed of the pattern's orientation ${}^cR_t = R_x(\theta_1)R_y(\theta_2)R_z(\theta_3)$ to the camera and the pattern position vector to the camera $v_t^c \in \mathbb{R}^3$. Thus, the parameter vector to be identified to obtain the camera–pattern relationship is $\theta = (R, v) = (\theta_1, \theta_2, \dots, \theta_6) \in \mathbb{R}^6$.

Joining Equations (4) and (5) and adding distortion models, the camera model remains as a Function (6) that projects points $p_t^t \in \mathbb{R}^3$ to $p_t^{ph} \in \mathbb{R}^2$ points of the camera image plane.

$$p_t^{ph} = \Psi(\delta, \theta, p_t^t) \tag{6}$$

Then, helipad pose estimation is the problem of minimizing the reprojection error Equation (7) of the observed helipad pattern features. One of the classic features to identify by computer vision are the corners. If the pattern is known, we know a priori the 3D position of these corners in the reference frame of the pattern.

$$\hat{E} = arg\ min \sum_{p_i^t \in \mathbb{C}} \left[\Psi(\delta, \theta, p_i^t) - O(p_i^t)\right]^2 \tag{7}$$

where $p_i^t \in \mathbb{C}$ and $\mathbb{C}$ is a corner set of the pattern. $O(p_i^t) \in \mathbb{R}^2$ is the corners obtained in the camera plane by a specific computer vision algorithm such as the Harris or Susan algorithm [22,42].

Furthermore, since all points p_i^t belong to a pattern plane, the z-component of all corners in the pattern frame will always be 0. This quality allows solving Equation (7) using specific methods such as the Infinitesimal Plane-Based Pose Estimation (IPPE) [43].

The estimation of internal camera parameters requires a learning phase modeled in Equation (7) as an optimization problem. In addition, identifying the six parameters to define the transformation ${}^cT_t$ between the pattern calibration and the camera involves a similar process. Although both processes can be clustered as shown in Equation (7), the internal camera parameters δ will be constant for a particular vision system; however, the position of the pattern may change. For this reason, it is decoupled in two phases: on the one hand, a camera parameter learning (calibration) process, using a set of images of a known pattern to estimate internal camera parameters, and, on the other hand, the estimation of the helipad position for a certain image during flight.

2.3. Camera-Gimbal Frame

The camera is placed in a camera-gimbal socket, so it is necessary to include this referent frame $\{z\}$. As the camera-gimbal socket axis is equivalent with the general gimbal axes, but static, the ${}^zT_c$ transformation is shown in Equation (8).

$$\begin{gathered} {}^zT_c = \left[R_c^z \middle| 0^{3\times1}\right] \equiv \begin{pmatrix} R_c^z & 0^{3\times1} \\ 0^{1\times3} & 1 \end{pmatrix}; \\ R_c^z = R\left(x, \tfrac{\pi}{2}\right) R\left(z, \tfrac{\pi}{2}\right) \end{gathered} \tag{8}$$

where $R(x, \theta)$ and $R(z, \psi)$ represent θ and ψ rotations about the x and z axes of the camera reference system. As the axes of the gimbal and camera-gimbal socket are equivalent, the relationship between camera-gimbal socket and gimbal corresponds to the identity matrix ${}^gT_z = I^{4\times4}$.

2.4. Gimbal Body Frame

The gimbal's reference frame $\{g\}$ to the UAV's body gravity center frame is defined as the composition of a roto-translation in Equation (9).

$$\begin{gathered} {}^{b}T_{g} = \left[R_{g}^{b} \middle| p_{g}^{b}\right] \\ R_{g}^{b} = R_{g}(\theta,\ \phi, \psi); p_{g}^{b} = \left(x_{g}, y_{g}, z_{g}\right)^{T} \end{gathered} \tag{9}$$

For this work, we consider the gimbal is static but located at the p_{g}^{b} position with $R_{g}(\theta,\ \phi, \psi)$ rotation to the body axes.

In this paper we consider ${}^{b}T_{z} = \left[R_{b}\left(0, -\frac{\pi}{2}, 0\right) \middle| (0, 0, 0.1)^{T}\right]$, as shown in Section 4.1.

2.5. Body-NED Frame

The North-East-Down (NED) frame coordinates $\{n\}$ to the UAV body gravity center $\{b\}$, ${}^{n}T_{b}$ is equivalent to the body rotation at the angle defined by the yaw angle ψ to geographic north or azimuth, pitch attitude to horizon plane ϕ, and roll angle defined to gravity direction θ. These angles refer to the attitude and heading reference system (AHRS) frame of reference that groups magnetic, angular rate, and gravity (MARG) information. Generally, these systems usually include air data to provide altitude or wind speed information.

$$\begin{gathered} {}^{n}T_{b} = [R_{b}^{n} | 0^{3\times 1}]; \\ R_{b}^{n} = \begin{pmatrix} \cos\theta\cos\psi & \sin\psi\sin\theta\sin\phi - \sin\psi\cos\phi & \sin\psi\sin\theta\cos\phi + \cos\psi\sin\phi \\ \cos\theta\sin\psi & \cos\psi\cos\theta + \sin\psi\sin\theta\sin\phi & \sin\psi\sin\theta\cos\phi - \cos\psi\sin\phi \\ -\sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi \end{pmatrix} \end{gathered} \tag{10}$$

2.6. NED-ECEF-Global

The coordinate transformation between NED to the global reference frame $\{g\}$ requires the use of the Earth-Centered Earth-Fixed (ECEF) reference system $\{e\}$, which allows us to apply the corresponding geodetic transformations to the terrestrial model and finally obtain the coordinates in global terms. In our case, we use a WGS84 (World Geodetic System 84) [44] datum.

The constant parameters of the WGS84 datum in Figure 2 refer to: r_e semimajor axis (equatorial radius), r_p semiminor axis (polar axis radius), ε first eccentricity and $\varepsilon\prime$ second eccentricity of the ellipsoid. It is important to differentiate the geocentric coordinates, referred to as the ECEF system, from the geodetic coordinates, referred to as the geodetic model (WGS84). This difference is provided by the geodetic model (datum) and is represented in the diagram on the right of Figure 2, where $\varphi\prime$ refers to geocentric latitude and φ refers to geodetic latitude.

Given a point p_{t}^{n} expressed in NED reference frame $\{n\}$ of a local tangent plane (LTP) to a geodesic surface at a known point $p_{Ref}^{g} = (\lambda, \varphi, h)^{T}{}_{Ref}$, it can be expressed in ECEF coordinates $\{e\}$ applying Equation (12). This equation corresponds to a translation in ECEF reference frame. However, to obtain p_{Ref}^{e} coordinates of our reference point in ECEF frame it is necessary to transform the global coordinates to ECEF applying Equation (14). The transformation between local coordinates and ECEF is given by the transformation Equation (13). In this work, we consider $p_{Ref}^{g} = p_{UAV}^{g}$.

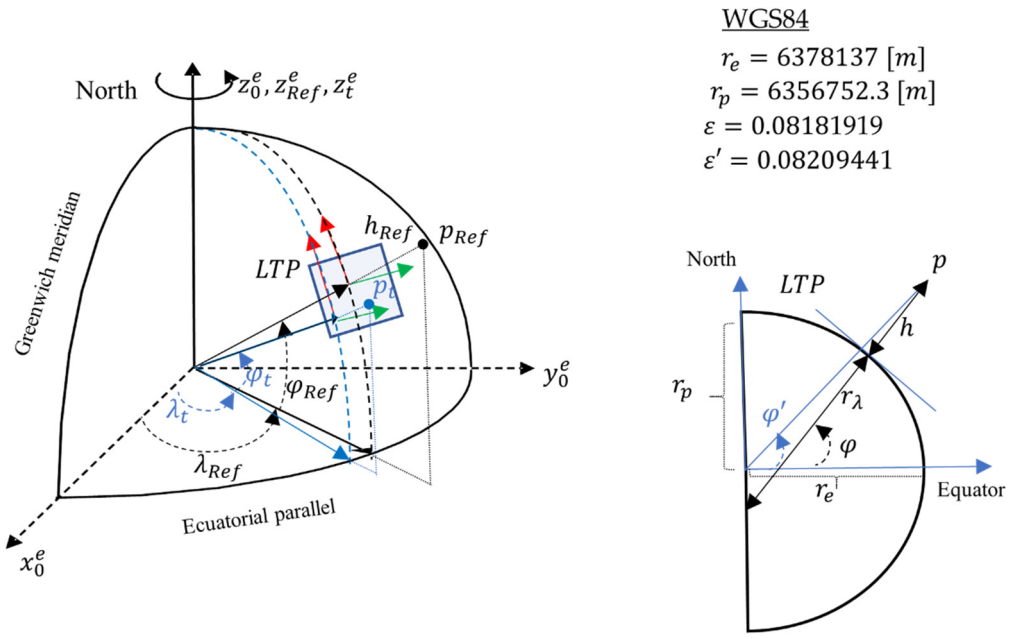


Figure 2. LTP, ECEF, and WGS84 reference systems and geometric relationships.

On the other hand, a given point p_t^e expressed in ECEF can be expressed in global coordinates p_t^g applying the transformation Equation (11).

$$p_t^g = \begin{pmatrix} \lambda \\ \varphi \\ h \end{pmatrix}_t = {}^gF_e(p_t^e) = \begin{pmatrix} tan^{-1}\left(\frac{y_t^e}{x_t^e}\right) \\ tan^{-1}\left(\frac{z_t^e + e'^2 Z_0}{r}\right) \\ U\left(1 - \frac{r_p^2}{r_e V}\right) \end{pmatrix} \tag{11}$$

where $(\lambda, \varphi, h)^T$ means longitude, latitude, and altitude in WGS84 datum. $(x_t^e, y_t^e, z_t^e)^T$ are the coordinates in the ECEF reference frame. The transformation Equation (11) corresponds to Jijie Zhu's algorithm [45] analyzed and compared in [46].

$$p_t^e = \begin{pmatrix} x_t^e \\ y_t^e \\ z_t^e \end{pmatrix} = {}^eF_n\left(p_t^n, p_{Ref}^e\right) = R_n^e \cdot p_t^n + p_{Ref}^e \tag{12}$$

$$R_n^e = \begin{pmatrix} -sin\varphi_{Ref}cos\lambda_{Ref} & -sin\lambda_{Ref} & -cos\varphi_{Ref}cos\lambda_{Ref} \\ -sin\varphi_{Ref}sin\lambda_{Ref} & cos\lambda_{Ref} & -cos\varphi_{Ref}cos\lambda_{Ref} \\ cos\varphi_{Ref} & 0 & -sin\varphi_{Ref} \end{pmatrix} \tag{13}$$

$$p_{Ref}^e = \begin{pmatrix} x_{Ref}^e \\ y_{Ref}^e \\ z_{Ref}^e \end{pmatrix} = \begin{pmatrix} \left(r_\lambda + h_{Ref}\right)cos\varphi_{Ref}cos\lambda_{Ref} \\ \left(r_\lambda + h_{Ref}\right)cos\varphi_{Ref}sin\lambda_{Ref} \\ \left((1-\varepsilon^2)r_\lambda + h_{Ref}\right)sin\varphi_{Ref} \end{pmatrix} \tag{14}$$

$$r_\lambda = \frac{r_e}{\sqrt{1 - \varepsilon^2 \sin^2 \varphi}} \tag{15}$$

3. Proposal

In this section, first the landing strategy is described, then the method to determine the helipad's global position is detailed, and finally the error analysis of the helipad's position estimation is given.

3.1. Landing Strategy

The landing strategy is responsible for telling the UAV navigation system the position to which it must go and the attitude it must have to align with the target (Algorithm 1). The position of the target is static, but the attitude and altitude to helipad vary overtime when the UAV attempts to land.

Algorithm 1 Landing Strategy

$\left[p_t^g, (\theta, \phi, \psi)_t^g, a\right]$ = helipad identificationh_{UAv}, ψ_{UAV} = UAV navigation sistem
if a = True
Buffer $\leftarrow [p,\ (\theta, \phi, \psi)]_t^g$
if $frequency = 1Hz$ & $Buffer \geq 10$
$p_t'^g,\ \psi_t'^g = Filter\ (Buffer)$
Buffer reset
Buffer (1) $\leftarrow [p',\ \psi']_t^g$
$\psi_{set} = \psi_{UAV}^n + \psi_t^b$
$[\lambda, \varphi]_{set} \leftarrow p_t'^g$
$h_{set} = h_{UAV}\left(1 - 0.1e^{\frac{-1}{h_t^b - 0.5h_0}}\right)$
UAV navigation planer $\leftarrow [\lambda_{set},\ \varphi_{set}, h_{set},\ \psi_{set}]$
if $h_{UAV} \leq h_0$
PX4 landing mode
break
end if
else
$goto \rightarrow 1$
end if
else
$goto \rightarrow 1$
end if

3.1.1. Helipad Azimuth

To align the drone to the marker, it is necessary to determine the azimuth of the marker ψ_t^n. For this, we use the azimuth of the drone ψ_{UAV}^n and the orientation of the marker to the drone ψ_t^b.

Figure 3 shows how the helipad azimuth can be obtained graphically by adding to the drone azimuth the orientation of the aircraft to the landing pad in Equation (16). When both systems are aligned, the marker azimuth will be equal to the drone azimuth.

$$\psi_t^n = \psi_{UAV}^n + \psi_t^b \tag{16}$$

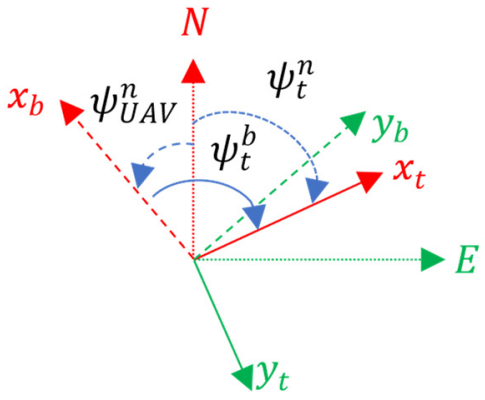


Figure 3. Helipad azimuth set formulation.

3.1.2. Altitude Setpoint Strategy

In order to change the default controller descent behavior, it is possible to use the behavior of the system in the transient state, i.e., before reaching the maximum velocity of stationary descent. Thus, if the new desired height is reached without having to reach the maximum descent speed, the behavior will be smooth, and if the destination point is far enough away, the controller saturates and descends with maximum constant speed behavior, without exceeding the internal controller parameters.

To define step points that allow a linear descent at constant speed α in an iterative loop, the new step point will correspond to the current height minus a certain parameter α.

$$h_{set} = h_{UAV} - \alpha \tag{17}$$

Considering this process is iterative (discrete) with a sample time of Δt, the previous equation can be expressed as follows:

$$h_{t+1} = h_t - \alpha \Delta t \tag{18}$$

where t subscript means instant time. Solving the α term, it is verified that alpha corresponds to a speed term.

$$\frac{(h_{k+1} - h_k)}{\Delta t} = \alpha = \frac{\Delta h}{\Delta t} = cte. \tag{19}$$

In our case, the aim is to design the $h_{set}(h_{UAV})$ function such that the aircraft approaches with a smooth behavior to h_0 and at that point lands automatically with internal autopilot.

To perform this, we propose Function (23).

$$h_{set} = h_{UAV}(1 - \beta_1 e^{\frac{-1}{h_t^b - \beta_0 h_0}}) \tag{20}$$

where β_1 is the weight of the exponential function and $\beta_0 < 1$, which allows slightly shifting the value of h_0 and to be able to switch to automatic landing mode. The $\beta_1 = 0.1$ value is set heuristically, while $\beta_0 = 0.5$ is set to shift 50% less than the switching height h_0. The system must consider the relative flight altitude h_{UAV} and the height of the UAV relative to the landing pad h_t^b.

Figure 4 shows an approximate representation of the altitude-set function behavior Equation (20) vs. constant decreasing Equation (17).

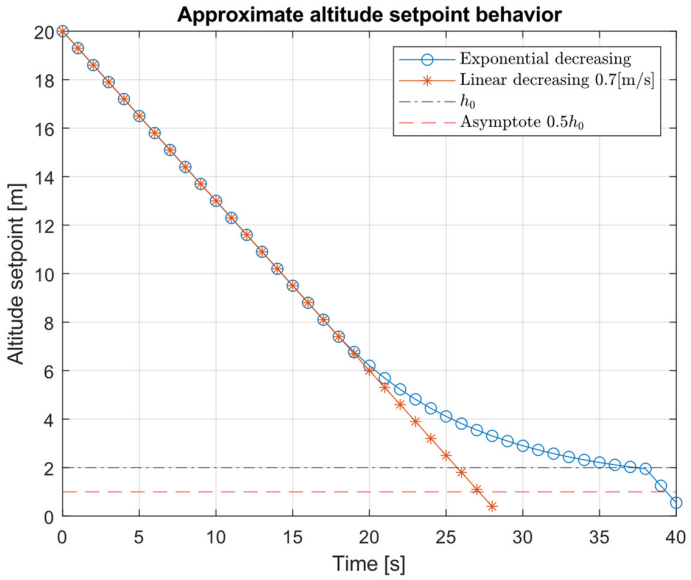


Figure 4. Approximate altitude setpoint evolution.

Figure 4 shows the approximate descent behavior of our proposal versus a constant speed descent. In the final phase of the approximation, the descent becomes smoother than in the linear behavior. The Figure 4 behavior should be taken as an illustrative example of the desired behavior, not as a realistic simulation. The final behavior can be seen in experimentation.

3.1.3. Filter

The states to be filtered are the global position of the helipad $p_t^g = (\lambda_t,\ \varphi_t)$ and its orientation to north or azimuth ψ_t^n. All these variables are static, since the landing pad is static; therefore, the filter model does not need to provide information for each new measurement, rather we need to know their stationary statistical values. For this we propose to generate a data buffer with memory. The size of the buffer defines the size of the filtering window L. The initialization saves L new measurements and then finds the mean or median of the buffered data. Finally, the buffer is reset.

To propagate the information over time, the sliding window does not overlap with previous values, but the value filtered at the previous instant is included as the first measurement in the clean buffer.

As for the filter memory, if the new values change substantially it will vanish in the long term, since the weight given to the past values is $\frac{1}{L}$ versus $\frac{L-1}{L}$ for each new data, so the window size can be critical for cases where the target is moving. Finally, the size of the buffer/window L is linked to time thanks to the 1 Hz system sampling time to provide new measurements.

3.2. Helipad Global Position Estimation

The helipad global position estimation system is responsible for integrating the vision system, the heliport context information, the gimbal, and the UAV navigation states, to provide the landing strategy with the helipad global position. In addition, this includes a spatial correction system for the NED frame, which is the objective of study of this work.

Figure 5 shows the diagram of the estimation system which is formulated in Equation (2). The system works as follows:

- Aircraft global position p_{UAV}^g and attitude $(\theta, \phi, \psi)_{UAV}^b$ is requested by the PX4 flight controller via MAVLink protocol [24] supported by the MAVSDK API [25].
- The gimbal position $p_g^b(x, y, z)$ and attitude $(\theta, \phi, \psi)_g^b$ is requested by the AirSim simulation environment via UDP protocol described in Section 4.1. This information composes the ${}^bT_z$ transform.
- The vision system receives I image of $W \times H$ size and 3 RGB channels. The image is received via UDP protocol from the simulation system. In addition, the vision system has as input the context information from the helipad, the library (*Lib*) of the marker, the marker's identification number ($Id \in Lib$), and the real marker's size (MS) in meters. The library is characterized by the number of horizontal and vertical bits (squares) that form the geometry of the marker and the number of elements that make up the library. The vision system output provides a Boolean variable $a \in \mathcal{B}$, that indicates if the landing pad has been detected or not. In addition, it provides the position of the landing pad to the camera p_t^c and the attitude $(\theta, \phi, \psi)_t^c$.
- The camera pose estimation (p_c^t and ${}^tT_c$) is gated by the PnP method integrated in the OpenCV Aruco library [47] from a previously pre-calibrated camera (Test environment).
- The aircraft, gimbal, and landing pad position are combined in the set of ${}^nT_b$, ${}^bT_z$, and ${}^zT_c$ transformations to obtain the positioning $p^n{}_t$ and attitude $(\theta, \phi, \psi)_t^n$ of the landing pad in NED frame.
- The correction module provides the $p\prime{}_t^n$ positioning and attitude $(\theta, \phi, \psi)\prime{}_t^n$, tuned in NED coordinates.
- Finally, the target position in NED frame $p\prime{}_t^n$, together with the drone global position p_{UAV}^g and ellipsoid WGS84 approximation, are used to obtain the helipad global position in Equation (11).

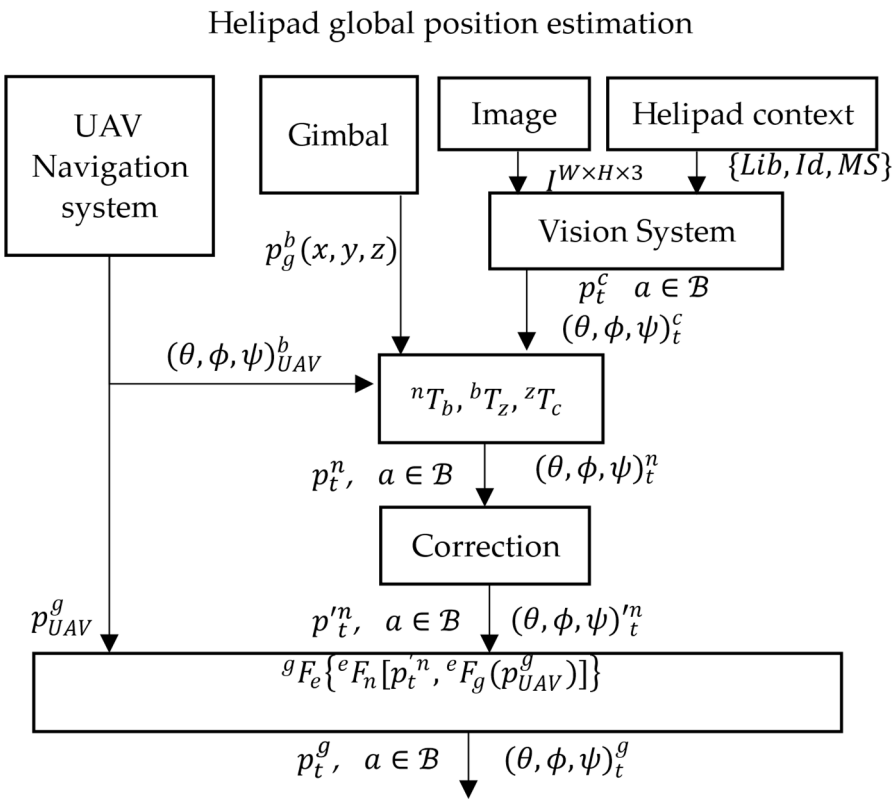


Figure 5. Helipad global position estimation system.

4. Landing System Analysis

The aim of this section is to evaluate the proposed estimation system and to identify the necessary corrections to be incorporated in the "correction" module of Figure 5. To achieve this, first the test environment and the necessary parameters are detailed in the subsection Test environment. Next, the system estimation error is modeled to provide the landing system a correction module. The quality of the correction is evaluated using the root mean square error (RMSE) together with the variation in the data distribution in terms of data distribution structure, mean, and standard deviation.

Finally, a full landing system and classical linear decreasing descent are compared and evaluated with four quality metrics, which quantify the trajectory length, the time to land, and the accuracy of landing on the helipad.

4.1. Test Environment

In this work, we use a hyper-realistic test environment based on Software in The Loop (SITL). SITL systems are simulation architectures where virtual world environments interact to simulate object, vehicle, and sensor together with external systems such as a flight controller or ground station, among others. These environments are powerful testing tools for earlier phases of system integration, as they allow realistic results to be obtained without potentially dangerous and expensive risks.

In our case, AirSim [48] is used as world environment and PX4 flight controller configured as a quadcopter. The simulated physical model corresponds to the Iris quadcopter and the set of sensors, and their specifications are detailed in Table 1. The models of the simulated sensors can be found detailed in [34].

Table 1. Sensor parameters simulated in AirSim.

Sensor	Parameters
Barometer IMU GPS Magnetometer Distance	Default AirSim settings [49]
Gimbal-Camera	Resolution $W \times H$: 640 × 480 Field of view (FOV) : 95 Depth of field focal distance : 100 Depth of field focal region : 100 Depth of field F − Stops : 2.8 Target gamma : 1.5 $p_c^b \equiv p_g^b = [0, 0, 0.1]$ $(\theta, \phi, \psi)_g^b = (0, -\frac{\pi}{2}, 0)$

Figure 6 shows an SITL communication diagram between the main system modules in SITL. The GCS module refers to the ground control station, in our case QGround control [50]. GCS is used to help to download the .log files generated in the test missions.

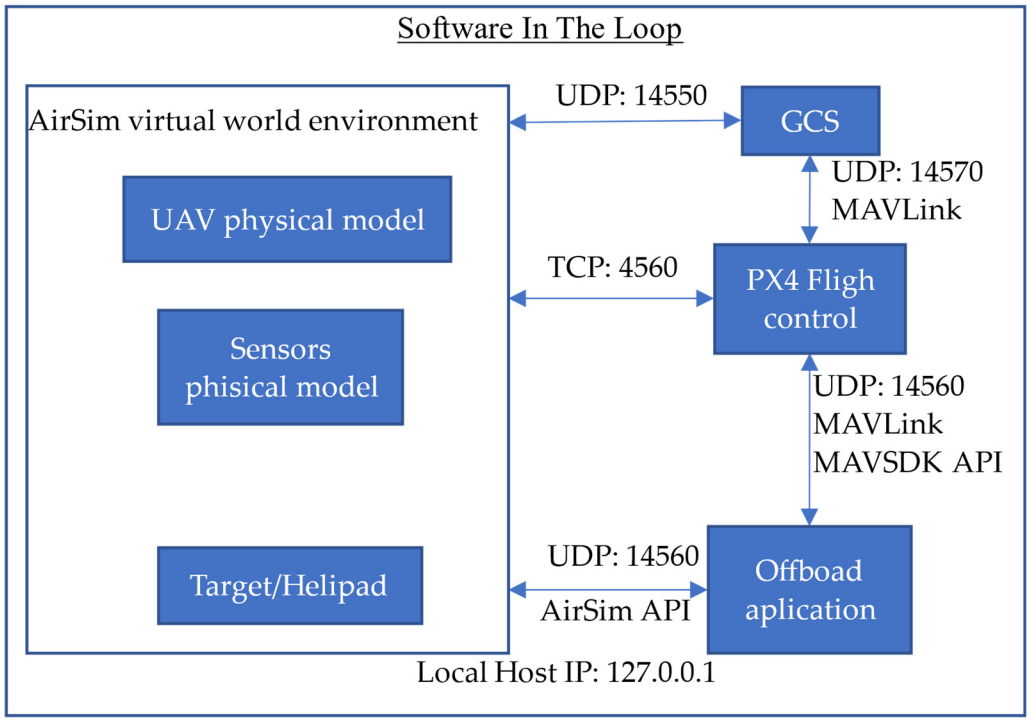
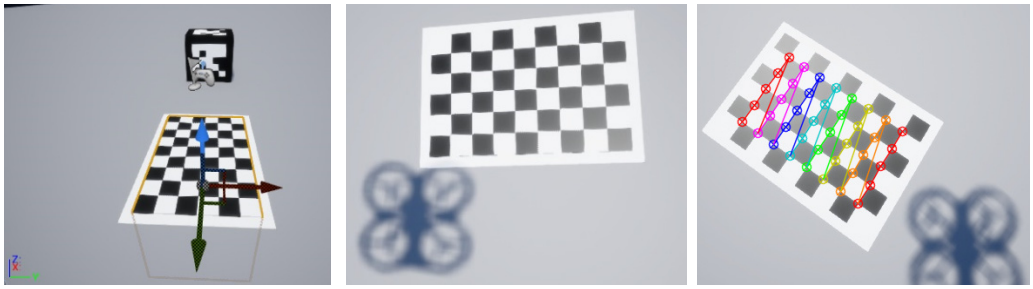


Figure 6. SITL Communication and protocol diagram.

The vision-based estimation system requires knowledge of the internal camera parameters $\{A, k_i\}$. These parameters are obtained by standard calibration [39] using a chess pattern with nine rows, six columns and 20 cm sides of the squares. This pattern is integrated into the AirSim environment as a texture over a rectangular prism with $1.8 \times 1.2 \times 1.8$ $[m]$ sides. To capture images, we implemented a system that automatically captures images while performing a spiral upward flight over the reference pattern. This allows obtaining a large set of images with the pattern from different positions.

Figure 7 shows the image of the calibration pattern in the AirSim reference frame, random image of the image registration process used for calibration, and a sample of the reprojection error.



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

Figure 7. Simulation environment in the calibration process: (**a**) Image of the calibration pattern in the AirSim reference frame; (**b**) Random image of the image registration process; (**c**) Example of reprojection error.

Finally, the internal camera parameters are shown in Table 2. The context information used for the experimentation is: $Lib = 5 \times 5 \times 1000$, $Id = 68$, and $MS = 1\ [m]$.

Table 2. Internal camera parameters.

Parameter	Value
f_x	293.35 [mm]
f_y	293.31 [mm]
c_x	319.64 [px]
c_y	239.64 [px]
Distortion coefficients k_i	$\{16.44,\ 35.89,\ 6.35, -6.35,\ 100.7\} \times 10^{-4}$

The landing system was developed in Python 3.6 with the AirSim [34] and MAVSDK [25] APIs. The experiments and the SITL environment were developed on a Windows Server 2019, 64 bits, hosted in AMD Ryzen 9 3900X 12-Core Processor CPU, 3.79 GHz with 64 GB RAM and 2×1TB SSD + 2×HDD 1.5TB of internal memory, graphic card Nvidia GeForce RTX 2060.

4.2. NED Error Modeling

To evaluate the estimation error, we propose to analyze the estimation data provided by the vision system over twenty static flights located at seventeen different positions at the same relative altitude above the ground, 10 meters.

The selected positions correspond to five different headings centered, 45° {northeast, southeast, southwest, and northwest} and four different distances $\{2, 3, 4, 5\}\sqrt{2}$ to the takeoff origin where the helipad is located. In each position is recorded a total of 1000 p^t_{UAV} samples. Looking at Figure 8, while the blue line maintains the desired directions of 45° (NE, SE, SO, NO), the centers of the positions recorded by the vision system, the red line, are decoupled, showing a constant angular deviation of the positions.

We consider the aircraft control system is asymptotically stable so that in steady state its position converges to the reference one. Thus, we consider as ground truth the reference positions for the steady flight.

The error position for each of the components is given by Equation (21).

$$e(x)^f_i = x^f_i - x^f_{GT_i} \tag{21}$$

where $e(x)^f_i$ means the position error of the component x of the system i in f referent frame. GT subscript means the ground truth in f reference frame.

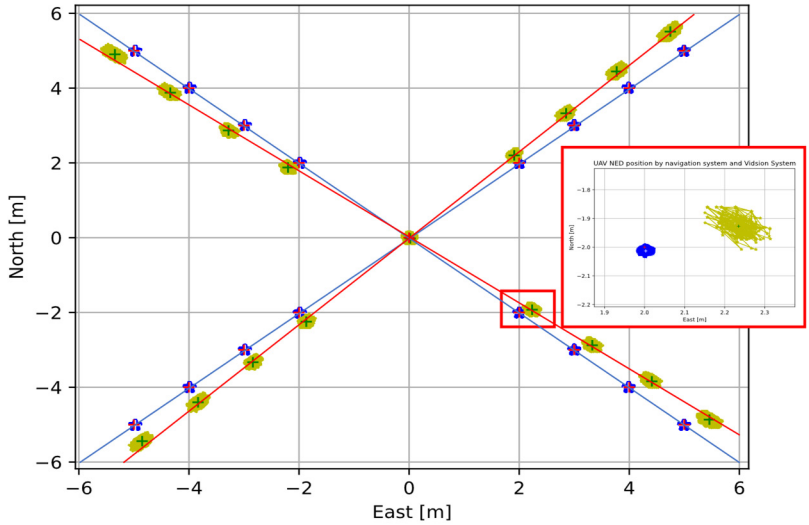


Figure 8. Data registration in twenty different flights. Yellow, UAV positions with vision system. Blue, UAV positions with navigation system. Blue and red line, linear approaches between navigation and vision system data centers, respectively. Red box, zoom in [−2, 2, 10] NED position.

Looking at the errors (Figure 9), the error distribution increases with increasing distance from the north-east plane origin (0, 0). This means the error position depends on the position in the NE plane.

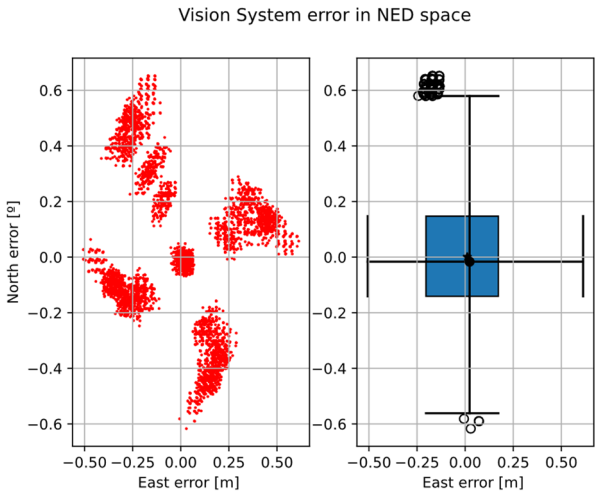


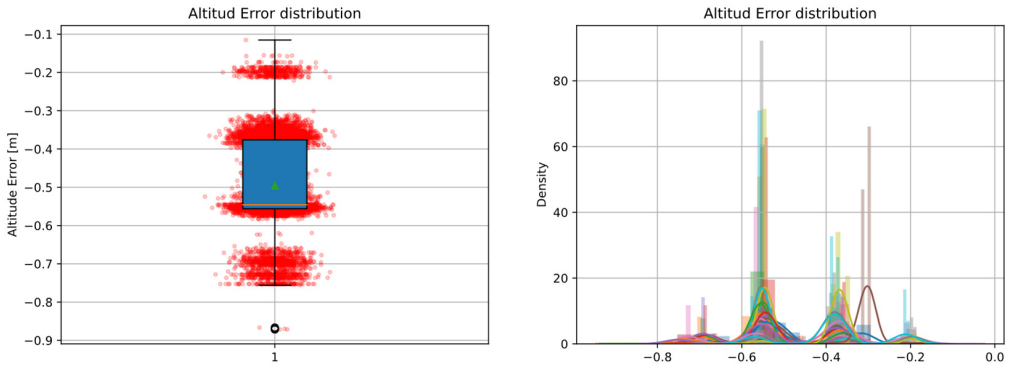
Figure 9. Vision system error in north-east plane coordinate. Left, scatter distribution; Right, north-east boxplot with 1.5 whis.

Regarding altitude error, Figure 10a,b show for each twenty register positions a distribution with four "modes". In Figure 10a these modes show as four scatter clusters and in Figure 10b as four peaks in each twenty distributions. In addition, the mean and median of the total error distribution, Figure 10, are displaced from the origin, showing a bias in altitude.

The different colors in Figure 10b show each of the twenty records, all of them showing four modes and centered on the same error terms. In this work, we focus on bias correction of mean and median; however, modeling the error in altitude is outside the scope of this paper.

The altitude error distribution may be a consequence of the internal discretizing of the simulator in the image render, so that the vision system, when segmenting the ROI of the helipad, extracts its contour with a size variation. This would be explainable according to

the pinhole model of Equation (4) and PnP Formulation (7), since its scale factor is constant, but the size of the ROI and corner positions would change.



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

Figure 10. Altitude error: (**a**) Scatter and boxplot. Green triangle: mean, orange line: median; (**b**) Altitude error distribution for twenty different register positions.

4.2.1. Polar Space Error Analysis

The visual results in Figures 8 and 9 show an apparent angular and radial bias of the helipad global position estimation system. We change the cartesian space to the cylindrical space defined by Equation (22), where the terms E, N, D are the coordinates in the NED referent frame.

$$\begin{aligned} r_0 &= \sqrt{E^2 + N^2} \\ \theta_0 &= atan\left(\frac{N}{E}\right) \\ D_0 &= D \end{aligned} \tag{22}$$

When plotting data in the new space in Figure 11, it can be seen how the data set is apparently clustered around a constant bias in the angle and radial error.

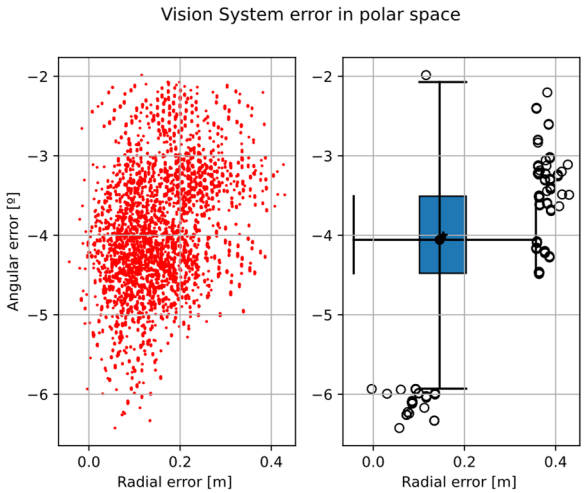


Figure 11. Vision system error in polar space. Left, scatter distribution; Right, 2D boxplot with 1.5 whis.

However, when showing the distance error (radial) behavior versus distance, Figure 12 shows a high linear correlation between distance and radial error. The angular error is also tested for linear dependence on distance, but the correlation does not exceed 35% of variance score r^2 in Equation (23), so it has been discarded.

$$r^2 = 1 - \frac{Var(x - \hat{x})}{Var(x)} \tag{23}$$

where $Var(x - \hat{x})$ indicates the variance of the error between $\hat{x}$ model estimation and x data.

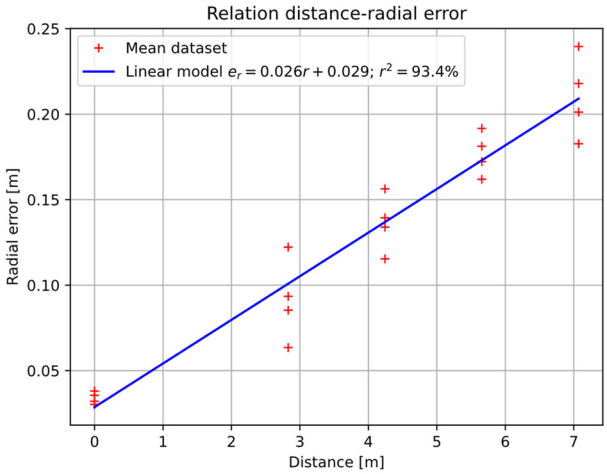


Figure 12. Relationship between distance and radial error. Red + symbol, centroid of each position. Blue line, linear model fitted by least squares.

Figure 12 shows with "+" the radial error centers of each of the measurement positions and the linear model fitted (blue line) by least squares to these points. This model has a slope $\alpha_r = 2.4923 \times 10^{-2}$ and independent term $\beta_r = 2.778497 \times 10^{-2}$ $[m]$. The linear model obtains around 94% of the variance score in Equation (23).

Given the results in Table 3 and Figure 12, the error bias in polar space can be tuned by the model of Equation (24), where the apostrophe over coordinates means corrected coordinate, subscript zero start value, and β_i the bias of coordinate i.

$$\begin{aligned} r' &= r_0(1 - \alpha_r) - \beta_r \\ \theta' &= \theta_0 - \beta_\theta \\ D' &= D_0 - \beta_D \end{aligned} \tag{24}$$

Table 3. Stationary error.

	β_θ [°]	Distances [m]	Altitude β_D [m]
Mean	−4.007078	0.153636	0.488412
Median	−4.056872	0.145540	0.545400
Var	0.512071	0.005182	0.014129

4.2.2. Error Correction in NED Space

Given N, E, and D coordinates of a point p_i^n in the NED frame and knowing the correction in a cylindrical space in Equation (24), the objective is to return to the NED space. For this purpose, we apply the transformation Equation (25).

$$\begin{aligned} \hat{N} &= \hat{r}.cos(\theta) \\ \hat{E} &= \hat{r}.sin(\theta) \\ \hat{D} &= D_0 - \beta_D \end{aligned} \tag{25}$$

where its terms $\hat{r}$, θ are taken as shown in Equation (26):

$$\begin{aligned} \hat{r} &= |v_{\beta_\theta}| - r' \\ r' &= \alpha_r r_0 + \beta_r \\ \theta &= atan\left(\frac{v_N}{v_E}\right) \\ v(\beta_\theta) &= \begin{cases} v_E = E \cdot \cos(\beta_\theta) - N \cdot \sin(\beta_\theta) \\ v_N = E \cdot \sin(\beta_\theta) + N \cdot \cos(\beta_\theta) \end{cases} \end{aligned} \tag{26}$$

where the hat over N, E, and D in Equation (26) means the NED coordinate with cylindrical corrections. $\beta_{\{\theta,r,D\}}$ are the biases of the radial, angular, and altitude terms, respectively. v_{β_θ} components and θ mean new position and new angular position after β_θ rotation correction.

Figure 13 shows the error distributions of the raw position estimation and error distribution after applying the correction Equation (26) with the parameters of Table 1.

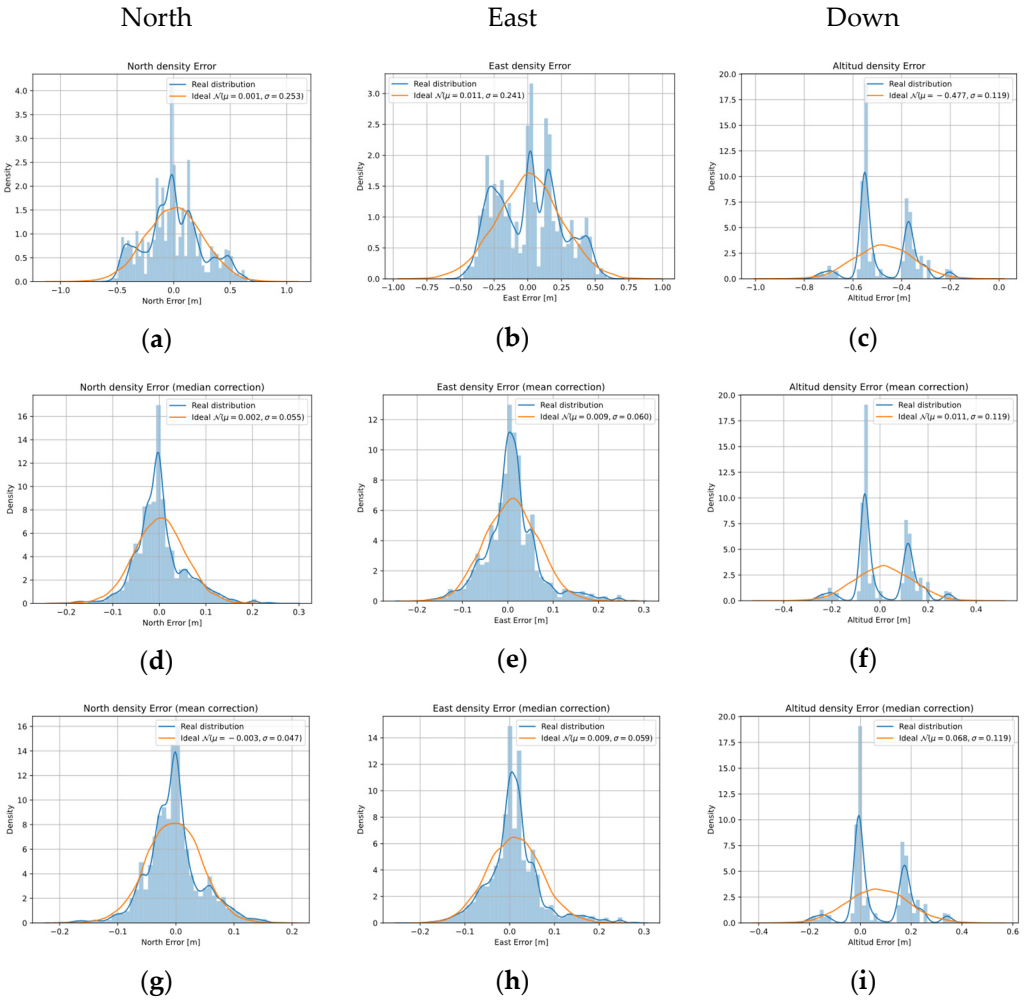


Figure 13. NED coordinates density error distribution. North-East-Down data without correction (**a**,**b**,**g**). Data with mean cylindrical correction (**c**,**d**,**h**). Data with median cylindrical correction (**e**,**f**,**i**).

For each drawing in Figure 13, the distributions of the real data are shown in blue and with a blue line their error distribution function [51]. The orange line shows a Gaussian distribution equivalent to the real data in Equation (27). For the NED coordinate, three different figures are represented: raw data Figure 13a–c, data after cylindrical correction using the mean of the raw data Figure 13d–f, and data after cylindrical correction according to the median of the raw data Figure 13g–i. In addition, the mean value and the standard deviation of each case represented by μ and σ, respectively, are indicated on each graph in their legend.

$$\mathcal{N}(\mu,\sigma,x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{27}$$

Figure 13 shows how correction Equation (25) modifies the structure of the error distribution for the cases of N and E coordinates, when comparing the blue with the orange lines. It can be seen in Figure 13a,b how the initial distribution is close to a Gaussian distribution Equation (27) and Figure 13d–h.

The effect on the Gaussian approximations in mean and standard deviation represented in Figure 13 is quantified in Table 4.

Table 4. Gaussian density distribution approximation.

	$\mathcal{N}(\mu_i,\sigma_i)$	Raw	Mean	Median
North	$\mu_N \ [\times 10^{-4}]$	7.930	20.200	17.120
	$\sigma_N \ [\times 10^{-2}]$	25.334	5.293	5.454
East	$\mu_E \ [\times 10^{-2}]$	1.087	0.851	0.882
	$\sigma_E \ [\times 10^{-1}]$	2.407	0.595	0.594
Altitude	$\mu_D \ [\times 10^{-1}]$	−4.774	0.110	0.680
	$\sigma_D \ [\times 10^{-1}]$	1.191	0.119	0.119

Table 4 shows the effect of mean and standard deviation on the data when applying the correction Equation (25) using the mean and median bias value indicated in Table 3.

The standard deviation rows of Table 4 decrease one order of magnitude in all coordinates when applying the correction Equation (25). The same effect can be seen in Table 5 when the RMSE Equation (28) of each NED coordinate is calculated. This metric can be considered as an indicator of accuracy.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\hat{x}_i^n - x_{GT_i}^n\right)^2} \tag{28}$$

Table 5. RMSE value.

Coor./RMSE	Raw	Mean	Median
North $[\times 10^{-2}]$	6.418	0.280	0.298
East $[\times 10^{-2}]$	5.804	0.361	0.361
Altitude $[\times 10^{-2}]$	2.421	0.143	0.188

4.3. Landing Evaluation

To evaluate the landing system, we propose to test twenty landing missions from the same position at (10, 10, −20) NED meters to the helipad. The twenty flights are divided into four groups corresponding to using the raw landing system without correction (*without*) Equation (2), applying the bias correction Equation (25) (*Bias*), with bias correction and a mean filter (Section 3.1.3) with a sliding time window (*Mean*&*Bias*) and a median filter together with the bias correction (*Median*&*Bias*). For each mission, we use as quality metrics the landing trajectory distance Equation (29), time to land Equation (30), and landing accuracy. In addition, the results are compared with classical linear descent setpoints with $\alpha = 0.7\ [m/s]$.

$$Dist = \sum_{k=1}^{K} \|p_{k+1}^n - p_k^n\| \tag{29}$$

$$Time = t_{start} - t_{land} \tag{30}$$

where the k term means temporal step starting in t_{start} instant and ending in t_{land} moment. p_k^n represents the global position at k instant in the NED reference frame.

For the flights' analysis, we use the information obtained from the logs recorded by the flight controller in each flight and unloaded with the ground control station (GCS). In particular, we focus on the global positioning of the UAV. This positioning is given by the EKF2 fusion system [52] integrated in PX4 and with the specific sensor parameters indicated in the SITL [33] (Section 4.1). To ensure that we evaluate exclusively the landing phase of the UAV, we study the trajectories from the instant t_{start} where the height gradient is detected to be negative, and the altitude is less than 99.8% of the altitude desired. The final instant t_{land} is obtained when the helipad is reached by the same method as t_{start}.

Finally, third quality metric, landing accuracy, is obtained with the RMSE of the last ten position samples of the NED coordinates (without altitude). In this way, we ensure

that we are on the ground with the same value plus a precision error. For this, we take as ground truth the control position of the marker.

Figure 14 shows the behavior of the aircraft when activating the landing system with the four modes corresponding to the landing system without correction (Figure 14a), landing with bias correction (Figure 14b), landing with bias correction and mean filter (Figure 14c,d), and landing with bias correction and median filter. The five trajectories in each of the figures show a different flight, using the corresponding landing mode in each case. The total number of flights is five for each landing mode, i.e., twenty flights.

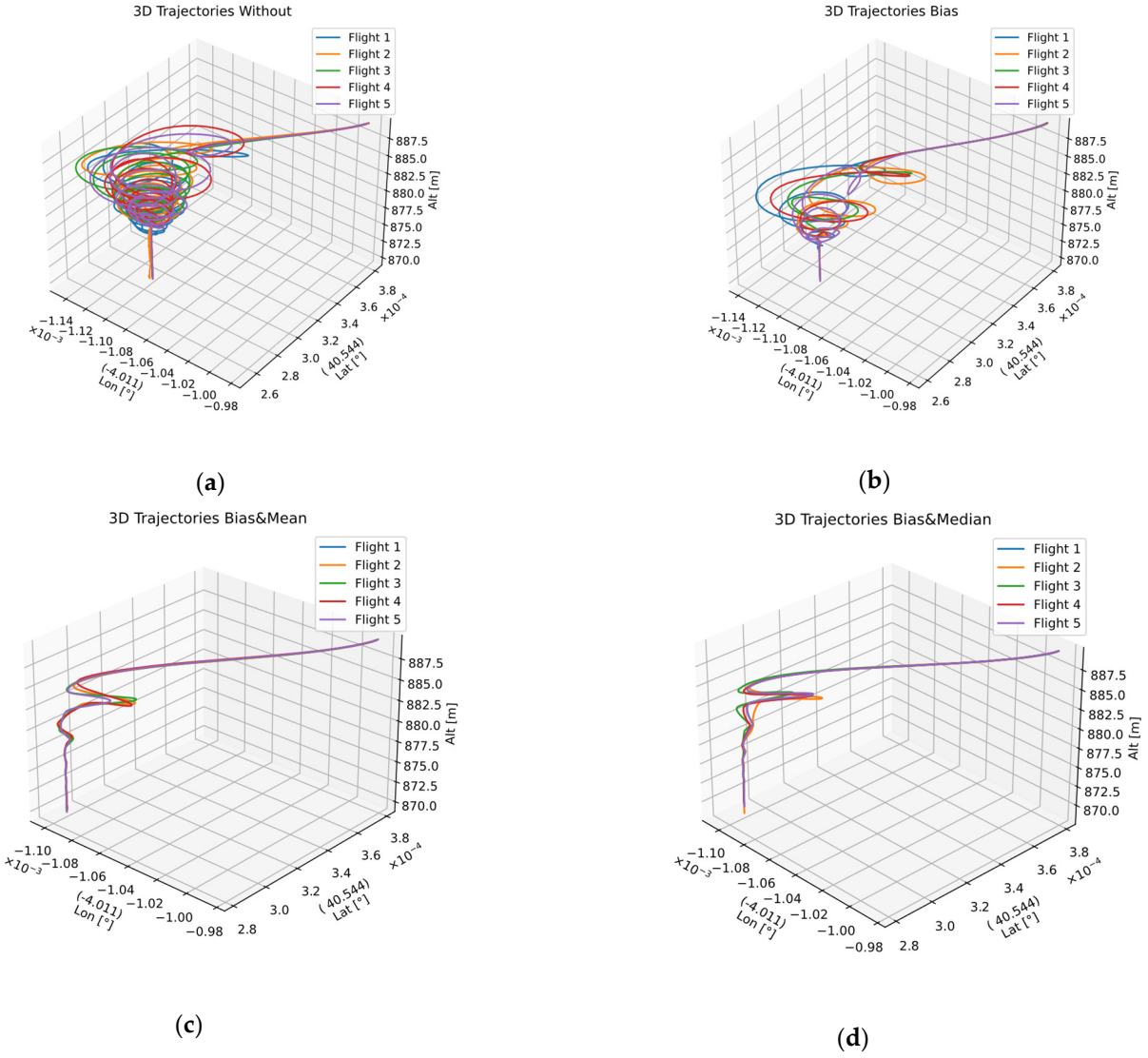


Figure 14. Five-flight 3D graphics for each of the four groups: (**a**) without correction; (**b**) bias correction; (**c**) bias and mean filter; (**d**) bias correction and median filter.

Figure 15 illustrates the temporal behavior of each analysis group, showing the three global position components: latitude, longitude, and relative altitude. In addition, our proposal is comparing with linear descent, paying special attention to altitude evolution Figure 15e,f. In this case, estimated altitude and setpoints are shown for the exponential proposal and linear descending.

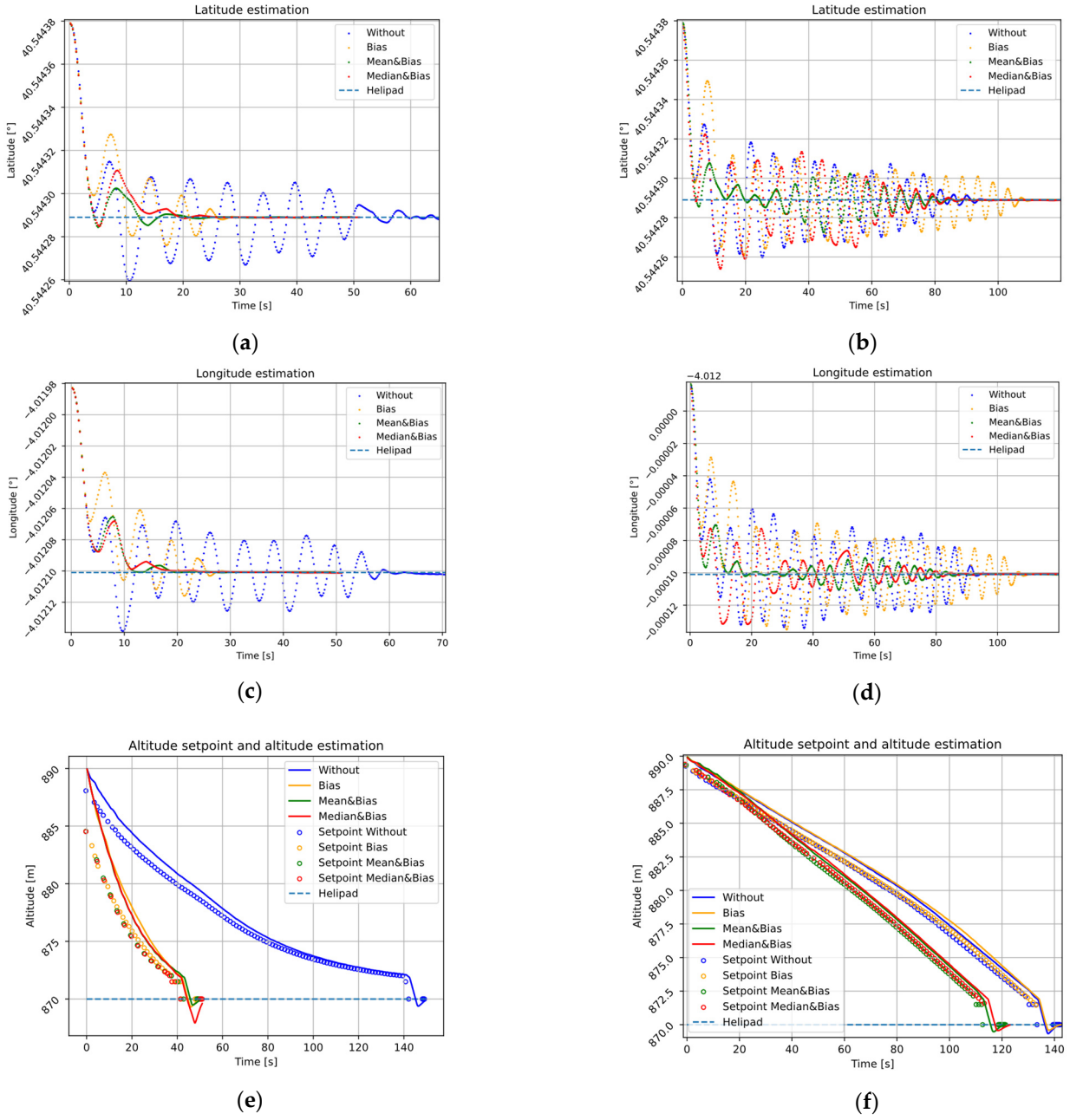


Figure 15. Time evolution of the latitude, longitude, and altitude of four flights with different correction modes in the landing phase: (**a**,**b**) latitude, (**c**,**d**) longitude, and (**e**,**f**) altitude. First column (**a**,**c**,**e**) exponential decrease, second column (**b**,**d**,**f**) linear decrease.

It can be seen in all cases in Figure 15 how the position of the landing pad, defined by a blue dashed horizontal line, is obtained in the stationary state. The effect of the error in precision is shown with an oscillating behavior (blue dotted line). However, when bias Equation (26) and filtering (Section 3.1.3) corrections are applied, the effect is damped.

The linear decrease approach converges in latitude and longitude (Figure 15b,d), but the system finds it difficult to dampen the spin effect. In the case of an exponential decrease, the inter-waypoint spin effect is considerably less than linear (Figure 15a–d). In both cases, the linear and exponential altitude decreasing approaches (Figure 15) show when error correction is applied and filtering estimation inter-waypoint noise spin effect is smoothed.

Figure 15e,f show a small break at the end of the trajectory that exceeds the reference and picks it up again. To identify the instant to obtain the landing surface, we keep the first term that satisfies the gradient and proximity conditions explained above.

This effect may be a consequence of a decoupling of the fusion system in the estimation of the height, for example for giving more weight to the estimation than to the measurements in the EKF2 filter. Therefore, when identifying the instant of reaching the pad, we are left with the first term that meets the conditions of gradient and proximity explained above.

The total of twenty flights with exponential decreasing approach and the other twenty flights with linear decreasing approaches are summarized in Tables 6 and 7. These tables are built with the mean and median values of all flights, so the quality metrics means the mean and median values of all flights.

Table 6. Mean value of quality metrics.

Exp. | Linear	Distance [m]	Time [s]	RMSE Landing $\times 10^{-7}$
Without	131.25 | 205.88	143.77 | 136.37	1.650 | 1.74
Bias	57.07 | 192.15	44.30 | 131.62	1.011 | 2.38
Mean&Bias	37.44 | 54.73	44.64 | 116.09	0.875 | 1.17
Median&Bias	37.91 | 67.25	43.62 | 131.07	1.119 | 2.68

Table 7. Median value of quality metrics.

Exp. | Linear	Distance [m]	Time [s]	RMSE Landing $\times 10^{-7}$
Without	130.14 | 205.95	144.93 | 136.37	2.885 | 2.12
Bias	57.25 | 195.46	44.54 | 131.81	1.193 | 2.43
Mean&Bias	34.89 | 60.68	44.95 | 116.09	1.037 | 1.18
Median&Bias	36.21 | 85.58	44.34 | 132.73	1.042 | 2.70

Tables 6 and 7 show the results of the exponential and linear decreasing approaches grouped for easy comparison. The previous tables show for all cases that the exponential descent proposal improves the results of the linear descent, emphasizing that in the best-case scenario of the linear approach (*Mena*&*Bias*), the trajectory distance is reduced by 32%, the time to land by 61%, and the RMSE of the precision landing by 12%.

In both tables, the mode that provides the minimum mean landing trajectory distance is the bias correction together with the median filter. The minimum mean time to land is provided by the bias correction together with the median filter and the minimum mean RMSE is again provided by the bias correction together with the mean filter. Finally, the similarity between the mean and median values in Tables 6 and 7 are a good indicator of normal distribution.

5. Conclusions

Through the study of the global position estimation system error of Section 4, an angular and radial bias was identified. In addition, it was shown how the error distribution increases its degree of dispersion with the distance to the origin. This position error was initially modeled in a cylindrical space in Equation (24) and transferred to the NED reference space under the transformation Equations (25) and (26). The corrections over cylindrical space produced a structural transformation of the position error distribution, approximating its distributions to the Gaussian error.

On the other hand, it was verified how using a system aimed at smoothing trajectories between waypoints can produce a spin effect if the new waypoints are updated with a frequency such that the UAV cannot obtain the previous target and these new waypoints correspond to the same position, but with high uncertainty. Therefore, the path planning system with path smoothing between waypoints can work as an error amplifier performing a circular trajectory.

Finally, we conclude that the combination of an exponential altitude decrease, together with the correction of systematic estimation error and a sliding time window filtering,

improves all three quality metrics proposed and reduces the effect of the inter-waypoint noise spin effect. These results facilitate the development of new applications that require a lightweight but robust precision landing strategy.

Author Contributions: Conceptualization, J.P.L.C., J.G.H. and J.M.M.L.; Formal Analysis, J.P.L.C., J.G.H. and J.M.M.L.; Funding Acquisition, J.G.H. and J.M.M.L.; Investigation, J.P.L.C.; Methodology, J.P.L.C.; Project Administration, J.G.H. and J.M.M.L.; Resources, J.P.L.C., J.G.H. and J.M.M.L.; Software, J.P.L.C.; Supervision, J.G.H. and J.M.M.L.; Validation, J.P.L.C., J.G.H. and J.M.M.L.; Visualization, J.P.L.C.; Writing—Original Draft Preparation, J.P.L.C.; Writing—Review and Editing, J.P.L.C., J.G.H. and J.M.M.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially funded by public research projects of Spanish Ministry of Science and Innovation, references PID2020-118249RB-C22 and PDC2021-121567-C22—AEI/10.13039/501100011033, and by the Madrid Government (Comunidad de Madrid, Spain) under the Multiannual Agreement with UC3M in the line of Excellence of University Professors, reference EPUC3M17.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: Thanks to Cristina Muntañola for her support to convert Figures 1 and 2 to .svg image format.

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

Nomenclature

Referent frames (RF)		p_i^j	Position of element i in j RF
$\{t\}$	Helipad (target)	${}^jT_i$	Linear translation i to j RF
$\{ph\}$	Pinhole model image plane	$[{}^jR_i \vert v_i^j]$	Rotation and translation between i and j RF
$\{c\}$	Camera	${}^jR_i$	Rotation between i and j RF
$\{z\}$	Camera socket	v_i^j	Translation between i and j RF
$\{g\}$	Gimbal	${}^jF_i$	Nonlinear RF transformation between i and j
$\{b\}$	Body gravity center	f_x	Focal
$\{n\}$	North-East-Down (NED)	f_y	length
$\{e\}$	Earth-Centerd Earth-Fixed (ECEF)	c_x	Principal point
$\{g\}$	Global WGS84 datum	c_y	
θ, ϕ, ψ	Euler angles	k_i	Distortion coefficients
$\lambda,\ \varphi,\ h$	Longitude, Latitude, altitude	A	Intrinsic camera matrix
$\mathbb{R}$	Real number space	δ	Intrinsic camera parameters
$\mathcal{B}$	Boolean number	Ψ	General camera model
$\mathcal{P}^2$	Planar projective space	$\hat{E}$	Reprojection error
$x, \hat{x}$	State and estimated state	O	Image feature function
$\|\|x\|\|$	Euclidian norm of x	$\mathcal{N}$	Normal distribution
L	Sliding window size	μ	Mean
β_i	Constant bias of coordinate i	σ	Standard deviation

References

1. Gautam, A.; Sujit, P.; Saripalli, S. A survey of autonomous landing techniques for UAVs. In Proceedings of the 2014 International Conference on Unmanned Aircraft Systems (ICUAS), Orlando, FL, USA, 27–30 May 2014; pp. 1210–1218.
2. Olivares-Mendez, M.A.; Mondragon, I.; Campoy, P.; Martinez, C. Fuzzy controller for UAV-landing task using 3D-position visual estimation. In Proceedings of the 2010 IEEE World International Conference on Fuzzy Systems, WCCI 2010, Barcelona, Spain, 18–23 July 2010.
3. Shi, G.; Shi, X.; O'Connell, M.; Yu, R.; Azizzadenesheli, K.; Anandkumar, A.; Yue, Y.; Chung, S.-J. Neural lander: Stable drone landing control using learned dynamics. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 9784–9790.

4. Jeong, H.-J.; Choi, J.D.; Ha, Y.-G. Vision Based Displacement Detection for Stabilized UAV Control on Cloud Server. *Mob. Inf. Syst.* **2016**, *2016*, 1–11. [CrossRef]
5. Chen, Y.; Zhou, Y.; Lv, Q.; Diversity, K.K. A review of V-SLAM. In Proceedings of the 2018 IEEE International Conference on Information and Automation, Wuyi Mountain, China, 11–13 August 2018; pp. 603–608.
6. Kang, Y.; Park, B.-J.; Cho, A.; Yoo, C.-S.; Kim, Y.; Choi, S.; Koo, S.-O.; Oh, S. A Precision Landing Test on Motion Platform and Shipboard of a Tilt-Rotor UAV Based on RTK-GNSS. *Int. J. Aeronaut. Space Sci.* **2018**, *19*, 994–1005. [CrossRef]
7. Janousek, J.; Marcon, P. Precision landing options in unmanned aerial vehicles. In Proceedings of the 2018 International Interdisciplinary PhD Workshop (IIPhDW 2018), Swinoujscie, Poland, 9–12 May 2018; pp. 58–60.
8. Aishwarya, C. The Instrument Landing System (ILS)—A Review. *Int. J. Progress. Res. Sci. Eng.* **2022**, *3*, 1–6.
9. Alam, S.; Oluoch, J. A survey of safe landing zone detection techniques for autonomous unmanned aerial vehicles (UAVs). *Expert Syst. Appl.* **2021**, *179*, 115091. [CrossRef]
10. DJI. Página Oficial. Available online: https://www.dji.com/es (accessed on 19 April 2022).
11. Yoakum, C.; Cerreta, J. A Review of DJI's Mavic Pro Precision Landing Accuracy. *Int. J. Aviat. Aeronaut. Aerosp.* **2020**, *7*, 1–19. [CrossRef]
12. Mittal, M.; Mohan, R.; Burgard, W.; Valada, A. Vision-Based Autonomous UAV Navigation and Landing for Urban Search and Rescue. *arXiv* **2022**, arXiv:1906.01304.
13. Wubben, J.; Fabra, F.; Calafate, C.T.; Krzeszowski, T.; Marquez-Barja, J.M.; Cano, J.-C.; Manzoni, P. Accurate Landing of Unmanned Aerial Vehicles Using Ground Pattern Recognition. *Electronics* **2019**, *8*, 1532. [CrossRef]
14. Zheng, Y.; Xie, H. Review on Neural Network Identification for Maneuvering UAVs. In Proceedings of the International Conference on Sensing, Diagnostics, Prognostics, and Control (SDPC 2018), Xi'an, China, 15–17 August 2018; pp. 339–346.
15. Mebarki, R.; Lippiello, V.; Siciliano, B. Autonomous landing of rotary-wing aerial vehicles by image-based visual servoing in GPS-denied environments. In Proceedings of the 2015 IEEE International Symposium on Safety, Security, and Rescue Robotics, West Lafayette, IN, USA, 18–20 October 2015.
16. Abujoub, S.; McPhee, J.; Westin, C.; Irani, R.A. Unmanned Aerial Vehicle Landing on Maritime Vessels using Signal Prediction of the Ship Motion. In Proceedings of the OCEANS 2018 MTS/IEEE, Charleston, CA, USA, 22–25 October 2018.
17. Mondragón, I.F.; Campoy, P.; Martínez, C.; Olivares-Méndez, M.A. 3D pose estimation based on planar object tracking for UAVs control. In Proceedings of the 2021 IEEE International Conference on Robotics, Automation and Artificial Intelligence, Hong Kong, China, 21–23 April 2021; pp. 35–41.
18. Lebedev, I.; Erashov, A.; Shabanova, A. Accurate Autonomous UAV Landing Using Vision-Based Detection of ArUco-Marker. In *International Conference on Interactive Collaborative Robotics*; Springer: Cham, Switzerland, 2020; Volume 12336, pp. 179–188.
19. Li, S.; Xu, C.; Xie, M. A Robust O(n) Solution to the Perspective-n-Point Problem. *IEEE Trans. Pattern Anal. Mach. Intell.* **2012**, *34*, 1444–1450. [CrossRef]
20. Garrido-Jurado, S.; Muñoz-Salinas, R.; Madrid-Cuevas, F.J.; Marín-Jiménez, M.J. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognit.* **2014**, *47*, 2280–2292. [CrossRef]
21. Romero-Ramirez, F.J.; Munoz-Salinas, R.; Medina-Carnicer, R. Fractal Markers: A New Approach for Long-Range Marker Pose Estimation Under Occlusion. *IEEE Access* **2019**, *7*, 169908–169919. [CrossRef]
22. Li, B.; Wang, B.; Tan, X.; Wu, J.; Wei, L. Corner location and recognition of single ArUco marker under occlusion based on YOLO algorithm. *J. Electron. Imaging* **2021**, *30*, 033012. [CrossRef]
23. Pixhawk. The Hardware Standard for Open-Source Autopilots. Available online: https://pixhawk.org/ (accessed on 13 March 2022).
24. Koubaa, A.; Allouch, A.; Alajlan, M.; Javed, Y.; Belghith, A.; Khalgui, M. Micro Air Vehicle Link (MAVlink) in a Nutshell: A Survey. *IEEE Access* **2019**, *7*, 87658–87680. [CrossRef]
25. Introduction MAVSDK Guide. Available online: https://mavsdk.mavlink.io/main/en/index.html (accessed on 13 March 2022).
26. Open Source Autopilot for Drones—PX4 Autopilot. Available online: https://px4.io/ (accessed on 22 February 2022).
27. Lizarraga, M.; Curry, R.; Elkaim, G.H. Flight test results for an improved line of sight guidance law for UAVs. In Proceedings of the American Control Conference, Washington, DC, USA, 17–19 June 2013; pp. 818–823.
28. Anderson, E.P.; Beard, R.W.; McLain, T.W. Real-time dynamic trajectory smoothing for unmanned air vehicles. *IEEE Trans. Control Syst. Technol.* **2005**, *13*, 471–477. [CrossRef]
29. Kikutis, R.; Stankūnas, J.; Rudinskas, D.; Masiulionis, T. Adaptation of Dubins Paths for UAV Ground Obstacle Avoidance When Using a Low Cost On-Board GNSS Sensor. *Sensors* **2017**, *17*, 2223. [CrossRef]
30. Iii, D.W.S.; Sanfelice, R.G. Autonomous Waypoint Transitioning and Loitering for Unmanned Aerial Vehicles via Hybrid Control. In Proceedings of the AIAA Guidance, Navigation, and Control Conference, San Diego, CA, USA, 4–8 January 2016.
31. Park, S.; Deyst, J.; How, J. A New Nonlinear Guidance Logic for Trajectory Tracking. In Proceedings of the AIAA Guidance, Navigation, and Control Conference an Exhibit, Providence, RI, USA, 16–19 August 2004.
32. Stateczny, A.; Burdziakowski, P.; Najdecka, K.; Domagalska-Stateczna, B. Accuracy of trajectory tracking based on nonlinear guidance logic for hydrographic unmanned surface vessels. *Sensors* **2020**, *20*, 832. [CrossRef]
33. Ma, C.; Zhou, Y.; Li, Z. A New Simulation Environment Based on Airsim, ROS, and PX4 for Quadcopter Aircrafts. In Proceedings of the 6th International Conference on Control, Automation and Robotics (ICCAR 2020), Singapore, 20–23 April 2020; pp. 486–490.

34. Shah, S.; Dey, D.; Lovett, C.; Kapoor, A. AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles. *Springer Proc. Adv. Robot.* **2018**, *5*, 621–635.
35. Adli, S.E.; Shoaran, M.; Noorani, S.M.S. GSPnP: Simple and geometric solution for PnP problem. *Vis. Comput.* **2019**, *36*, 1549–1557.
36. PLi, P. Quantum implementation of the classical Canny edge detector. *Multimed. Tools Appl.* **2022**, *81*, 11665–11694.
37. Luo, S.; Hou, J.; Zheng, B.; Zhong, X.; Liu, P. Research on edge detection algorithm of work piece defect in machine vision detection system. In Proceedings of the 2022 IEEE 6th Information Technology and Mechatronics Engineering Conference (ITOEC 2022), Chongqing, China, 4–6 March 2022; pp. 1231–1235.
38. Hartley, R.; Zisserman, A. *Multiple View Geometry in Computer Vision*, 2nd ed.; Cambridge University Press: Cambridge, UK, 2003; pp. 262–278.
39. Datta, A.; Kim, J.S.; Kanade, T. Accurate camera calibration using iterative refinement of control points. In Proceedings of the 2009 IEEE 12th International Conference on Computer Vision (ICCV 2009), Kyoto, Japan, 29 September–2 October 2009; pp. 1201–1208.
40. Home—OpenCV. Available online: https://opencv.org/ (accessed on 21 July 2021).
41. Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library—Adrian Kaehler, Gary Bradski—Google Libros. Available online: https://books.google.es/books?hl=es&lr=&id=LPm3DQAAQBAJ&oi=fnd&pg=PP1&dq=G.+Bradski+and+A.+Kaehler,+Learning+OpenCV3:+ComputerVision+in+C%2B%2B+With+the+OpenCV+Library,+2nd+ed.+Newton,+MA,+USA:+O\T1\textquoterightReilly+Media,+2013&ots=2wLqQga9C7&sig=nzLIWPd4uyeVkH93pJkiN7b3hbA&redir_esc=y#v=onepage&q&f=false (accessed on 22 March 2022).
42. Patel, T.P.; Panchal, S.R.; Student, P.G. Corner Detection Techniques: An Introductory Survey. *IJEDR* **2014**, *2*, 2321–9939.
43. Collins, T.; Bartoli, A. Infinitesimal Plane-Based Pose Estimation. *Int. J. Comput. Vis.* **2014**, *109*, 252–286. [CrossRef]
44. Valavanis, K.P.; Vachtsevanos, G.J. *Handbook of Unmanned Aerial Vehicles*; Springer: Dordrecht, The Netherlands, 2015.
45. Zhu, J. Conversion of Earth-centered Earth-fixed coordinates to geodetic coordinates. *IEEE Trans. Aerosp. Electron. Syst.* **1994**, *30*, 957–961. [CrossRef]
46. Osen, K. Accurate Conversion of Earth-Fixed Earth-Centered Coordinates to Geodetic Coordinates. Ph.D. Thesis, Norwegian University of Science and Technology, Trondheim, Norway, 2017.
47. OpenCV: Detection of ArUco Markers. Available online: https://docs.opencv.org/4.x/d5/dae/tutorial_aruco_detection.html (accessed on 14 March 2022).
48. Home—AirSim. Available online: https://microsoft.github.io/AirSim/ (accessed on 22 February 2022).
49. Settings—AirSim. Available online: https://microsoft.github.io/AirSim/settings/ (accessed on 24 March 2022).
50. QGC—QGroundControl—Drone Control. Available online: http://qgroundcontrol.com/ (accessed on 29 March 2022).
51. Chen, Y.-C. A tutorial on kernel density estimation and recent advances. *Biostat. Epidemiol.* **2017**, *1*, 161–187. [CrossRef]
52. García, J.; Molina, J.M.; Trincado, J. Real evaluation for designing sensor fusion in UAV platforms. *Inf. Fusion* **2020**, *63*, 136–152. [CrossRef]