

DATA COLLECTION USING UNMANNED AERIAL VEHICLES AND A DELAY-TOLERANT NETWORK

EDISSON PAUL CABRERA, JEFFERSON SANTIAGO AGILA
FABIAN ASTUDILLO-SALINAS*, ANDRES VAZQUEZ-RODAS
SANTIAGO PATRICIO TORRES AND ISMAEL MINCHALA-AVILA

Department of Electrical, Electronics and Telecommunications
University of Cuenca

Av. 12 de Abril S/N and Calle Agustin, Cuenca 010207, Ecuador

{edison.cabrerat; santiago.agila17; andres.vazquezr; santiago.torres; ismael.minchala}@ucuenca.edu.ec

*Corresponding author: fabian.astudillos@ucuenca.edu.ec

Received November 2022; revised March 2023

ABSTRACT. *Wireless sensor networks (WSNs) are widely used in low-energy consumption and dispersed monitoring applications. In this context, a common issue in WSNs corresponds to connectivity in low-density networks. An unmanned aerial vehicle (UAV)-based data collection is a promising solution for data transfer to tackle this challenge. This work proposes and evaluates a delay-tolerant network (DTN) for data transmission, where the transport mule is based on a UAV. The transmission throughput between the sensor node and the collector node (UAV) is evaluated in an environment with three different distances between the sensor node and the UAV: 0 m (land), 3 m, and 6 m. The results show that the transmission takes less time when the UAV is on land, while the transmission time increases with greater distance.*

Keywords: WSN, UAV, OpenCV, DTN, Drone, Ardupilot, Raspberry Pi, MAVLink

1. Introduction. Data transfer between information management nodes in large geographic areas using WSNs is a challenging task, mainly due to the low scalability and the limited coverage area of traditional WSN [1]. On the other hand, the deployment of a wired transmission infrastructure represents a significant investment and is limited by the type of area and its accessibility conditions. Additionally, this infrastructure can imply a waste of resources according to the data flow generated by the nodes. WSNs are capable of overcoming issues associated with scalability and coverage. In this scenario, a WSN is a good option for obtaining, processing and transmitting information.

Data collection in WSNs is typically performed using one of the following three methods:

- **Static collection:** In this method, the sink node is fixed, and ordinary nodes upload data through one-hop or multi-hop routing;
- **Ground-based collection:** This method involves using a vehicle, typically equipped with a sink node, to collect data by visiting the network nodes;
- **Aerial mobile collection:** This method uses aerial vehicles to collect data from deployed nodes on the ground.

The processing and transmission of information in WSNs can become increasingly problematic in scenarios with low-density networks spread over a large area; in such scenarios, the static collection method may not be feasible, whereas ground-based mobile sink or mobile aerial methods could be deployed. However, the ground-based mobile sink method

is limited by ground transport. On the other hand, the aerial mobile method enables deployment in various environments with lower latency and higher bandwidth [2].

UAVs provide an information transport option for low-density networks spread over a large area. The incorporation of UAVs in various tasks and systems is due to their great capacity for customizing and adapting components to perform multiple functions [3]. An example of a UAV operating as a relay is found in [4], whose promising simulation results encourage the use of UAVs in low-density networks. According to [5], the data transmission in an aerial mobile method can be performed using store-and-forward, real-time, and hybrid data collection transfer models. In all cases, a UAV moves to positions within the range of a sensor node to collect data. The real-time data transfer model requires a way to send a message announcing a triggering event; this model is used in emergencies. The hybrid model combines the store-and-forward and the real-time data transfer models and is used depending on the situation and the type of data to be delivered.

The existing Internet protocols do not work well in certain environments. A DTN allows intermittent connection between nodes, and the data transfer is done using the store-and-forward model [6]. A DTN can carry data in environments where interruptions and high error rates are common [7], due to extreme conditions or node mobility. WSNs using DTN for covering vast geographic areas are emerging as a viable solution [8]. The movement of DTN nodes or mules (aerial mobile nodes) allows data to be transported between points in the network. Generally, the mules are adapted to existing services that partially communicate the area of interest.

We conducted this study for future implementation in precision agriculture and inspection of photovoltaic systems. Precision agriculture is based on the observation, measurement, and response to inter- and intra-parcel variability [9]. The purpose is to optimize the yield of agricultural supplies based on information collected by various means [10]. The University of Cuenca, Ecuador has a farm where we have some projects about transmitting variables used in agriculture. These variables are transmitted using LoRaWAN; however, there are some places where there is no coverage. We have two approaches. The first is to modify the LoRaWAN protocol to allow forwarder nodes and tree topology. The second approach is to use a UAV to transmit the data. On the other hand, the University has a microgrid laboratory. One of the projects aims to automate the inspection process in photovoltaic systems; several papers focus on this subject; one of them is [11]; it presents a system to address and detect the faults in a photovoltaic system by providing an inspection system in real-time using a UAV.

According to [2], the entire aerial data collection process can be divided into five steps: (i) the deployment of networks, (ii) node positioning, (iii) anchor point searching, (iv) fast path planning, and (v) data collection. This paper contributes to the last step. We analyze three network performance metrics in a UAV-based data collection system: delivery time, throughput, and delivery probability. We design and implement a system to analyze network performance metrics. These metrics evaluate the impact of the separation distance between nodes and the vehicle's flight on packet transmission and delivery in the DTN network. The first separation distance is 0 m, which means the UAV is on the ground. A heliport with a diameter of 1 m was built due to irregularities where the tests were performed. Consequently, an autonomous precision landing system is also required. The proposed autonomous landing system uses a camera, an ArUco marker, and an image processing system based on the OpenCV library. The distance measured between the center of the vehicle camera and the landing platform was defined to evaluate the performance of the landing system.

The main contributions of this paper are as follows.

- The evaluation of the proposed platform for aerial data collection with an autonomous landing system using OpenCV and ArUco marker. The source code and docker scripts are available at <https://github.com/fabianastudillo/uav-dtn>.
- Performance analysis based on node distance of a DTN network implemented in UAV-based data collection.

The remainder of the document is organized as follows. Section 2 discusses related works found in the literature. Section 3 illustrates the system deployment. Section 4 discusses the results. Finally, in Section 5, we conclude this paper.

2. Related Works.

2.1. Autonomous landing. The current literature reports several solutions related to UAV landing. In [12], a *WiFi* connection is established between the UAV and a computer responsible for edge detection in images captured by an onboard camera. A robot operating system (ROS) toolkit is used to recognize augmented reality markers (AR, *Augmented Reality*) to determine the position error in two axes and control the descent of the UAV using a proportional integral derivative (PID) control. In [13], the authors report the use of a *Raspberry Pi 3*; the ROS and OpenCV libraries are combined to perform edge detection (using the Canny Edge algorithm) and calculate the invariant moments of Hu. The latter detects the landing pad (marker) and exchanges flight parameters with the *Pixhawk* controller.

Similarly, the authors of [14] used a front camera to obtain two estimates of the UAV's position. The visual estimation detects the position of an AR marker using the ArUco library based on OpenCV. The sensor estimation uses a Kalman filter to improve the measurements obtained by the UAV's speed, acceleration, and angle sensors. The landing process is carried out using feedback from the marker-vertical camera assembly. Precision landing of UAVs can be used to accomplish tasks such as automatic battery replacement [15].

The characteristics of the landing platform are used to control the vehicle's approach to the ground. In [16], a semi-autonomous search algorithm is implemented to detect key points and compute descriptors to define landing marks. The algorithms used by the authors search for descriptor matches in the frames captured by an onboard camera, and speed signals are generated for the flight PID controller. In [17], an infrared camera (IR) detects a specially designed landing platform marked with IR diodes. The reading from the onboard camera is processed to establish speed signals for the flight controller. Similarly, in [18], authors use a camera for object detection based on HSV color threshold and a PID controller. However, a complementary filter has been introduced to determine the landing platform's altitude and position and mitigate the channel's effects.

The paper by Xu et al. [19] primarily focuses on detailing the methods employed for the autonomous landing of UAVs on moving targets. It utilizes both GPS and vision-based navigation throughout various stages of the landing process within a simulation environment. The main step involves estimating uncertain markers using techniques such as convex hull transformation, interference preclusion, ellipse fitting, and specific feature matching. Additionally, the paper introduces a comprehensive visual measurement program and guidance strategy specifically designed for UAV autonomous landing. Extensive experiments have been conducted, demonstrating the significance and feasibility of this vision-based approach for autonomous landing on moving targets.

In another review paper by Liu et al. [20], the research on UAV autonomous landing utilizing visual processing is thoroughly described. The paper covers various aspects, such as

image processing, target tracking, position estimation, and autonomous control technologies employed during the landing process. Existing limitations are identified, and research ideas are proposed to address these limitations, considering practical considerations.

Similarly, in a different review paper by Xin et al. [21], the authors provide a comprehensive summary of research findings in the field of vision-based autonomous landing for UAVs. The research is categorized into static, dynamic, and complex scenarios based on the specific landing destinations. Within the static scenario, cooperative targets and natural landmarks are the main categories, while the dynamic scenario encompasses vehicle-based and ship-based autonomous landing. The paper summarizes, compares, and analyzes key technologies while identifying future development trends. It serves as a valuable reference for further research in the domain of vision-based autonomous landing for UAVs.

2.2. UAV-based data collection system. UAVs typically perform flight planning using ground control stations (GCS) while maintaining intermittent or uninterrupted communications. This architecture is beneficial for enabling the deployment of a DTN. For instance, in [22], a UAV is integrated into a DTN scheme for relaying information from sensor nodes. In [23], UAV and DTNs are utilized for node searching in disaster scenarios. Furthermore, [24] explores the application of UAVs in search and rescue operations during hiking incidents.

WSNs present a suitable scenario for developing and implementing efficient data collection algorithms, which involves optimizing some process or resource in a system or function. The ultimate goal is to use UAVs for data collection in networks covering wide geographic areas by determining an efficient path regarding energy consumption and the number of nodes reached. In [25], a method based on heuristic algorithms is presented to solve the problem of aerial data collection. A testing scenario is proposed, where a network of UAVs is used, which have two types of communication: air-to-air and air-to-ground, targeting mobiles and a base station. The mobility of the targets has been used to minimize the hop route calculated to the base station using the *Dijkstra* algorithm. Thus, air-to-air communication is guaranteed by keeping the distance between pairs of UAVs within an established communication range. Likewise, air-to-ground communication is guaranteed by minimizing the altitude of the vehicle.

Wang et al. proposed a framework in [26], which includes network deployment, node positioning, anchor point search, fast route planning, and data collection. The network consists of two types of nodes: beacon and unknown. The guide node (beacon) is equipped with a GPS module, while the location of the unknown nodes is determined using the signal strength criterion for single-hop transmission. The fastest route is calculated using planning based on grid division, where a primary route is defined and the distance that passes through all guide nodes is minimized. Data collection is carried out through the leader nodes, considering one-hop routing.

In [27], WSN operation and an unmanned aerial system are used to improve data collection performance. The sensor network is organized based on the principle of groups, where node clusters are defined, and each cluster assigns a cluster head (CH) to transmit information to the UAV. The remaining nodes in the cluster are named cluster members. It operates on the premise that each node can adjust its transmission power, increasing or reducing its communication range. The network operation efficiency is achieved through the network partition and rotation process. Network partitioning allows the network to organize itself autonomously. The CH node is selected by using a weighting scheme based on the battery percentage and a ratio between the transmitted signal power and the received signal power between pairs of nodes within the cluster. The candidate CH node

with the highest weight is selected as the central node of the cluster. UAV flight planning through the network is defined as the traveling salesman problem, where the obtained path reduces the distance on a defined path with a starting and ending cluster. The stream is updated when a CH node triggers a CH rotation. This occurs when the CH node's battery is below a set threshold. The selection evaluates the distance the vehicle's current path traverses in the coverage area of the candidate node. A greater distance implies minor changes in the flight plan.

In [28], the authors discuss the challenge of effectively providing data-driven services supported by the Internet of Things (IoT) due to the lack of network infrastructure in large areas. It proposes a solution using UAVs as mobile networks to collect data from IoT devices. The UAVs follow a planned flight path based on the Hilbert Curve algorithm. The approach is validated through experiments, showing advantages over baseline methods. The research highlights the potential for cost-effective IoT applications in smart agriculture and public safety.

In [29], the authors discuss a proposed mechanism for autonomous data gathering in wireless sensor networks (WSNs) using unmanned aerial vehicles (UAVs). The objective is to train a UAV to collect data from ground sensor nodes distributed in a specific area. The approach combines deep deterministic gradient descent (DDPG) and Q-learning algorithms to optimize the UAV's trajectory and node visit order. Customized reward functions and simulations are used to evaluate the training performance. The proposed approach demonstrates effective data collection, and a comparison with a deterministic optimal solution validates its performance.

3. System Design. The monitoring system is a hybrid UAV-WSN multilevel system (see Figure 1). The ground level consists of WSN subsystems. Level 2 comprises sensor nodes, which collect data using specific sensor boards of Level 3. The UAV supports the ground

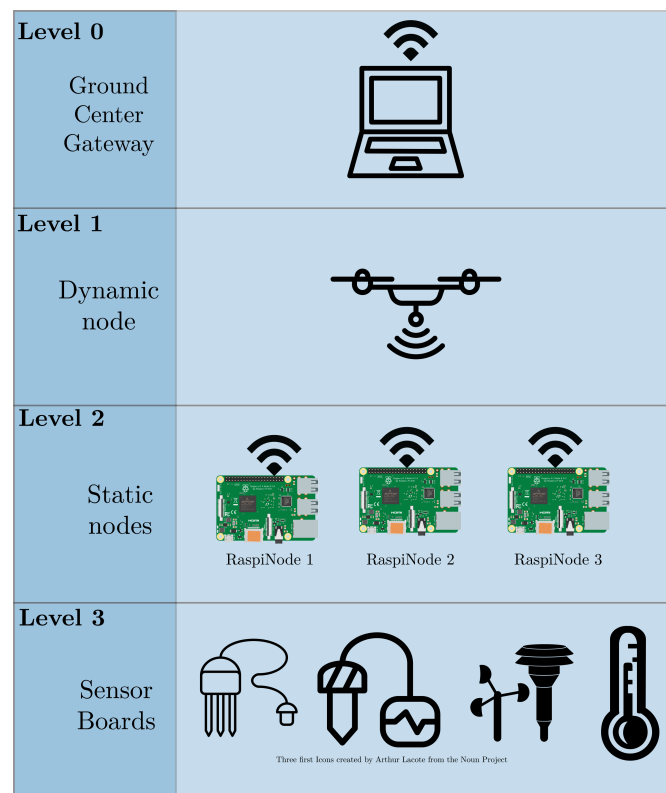


FIGURE 1. UAV-assisted WSN data collection

data collection from Level 1 and relays it to the ground center gateway (Level 0). In the implementation test, Level 3 is simulated using random variables. This representation is based on Dragana et al. [30].

The UAV described in Section 3.1 (UAV configuration) has been used to implement the platform, which requires an autonomous landing system (Section 3.2); it is needed to analyze different network metrics when the UAV lands at each waypoint. Section 3.3 presents the implementation of different algorithms to perform the flight process, while Section 3.4 shows the simplified network topology.

3.1. UAV configuration. The UAV used in this study is a buildable drone with a controller compatible with Ardupilot and PX4. The vehicle is a quadcopter in an X configuration. The flight controller used is PixRaptor, which is a generic version of the open hardware controller Pixhawk 1. A Raspberry Pi 3B is used to control the drone. Table 1 presents the main components of the assembled drone.

TABLE 1. Drone component specifications

Elements	Product
Frame	Carbon HJ-H4 reptile
Flight controller	PixRaptor
GPS+Compass	Ublox M8N
Gimbal	Tarot T-2D
Drone camera	Turnigy HD Wifi
Raspberry Pi 3	Model B, 1 GB RAM
Raspberry Pi Camera	Module version 2

After the UAV has been prepared for flight, the dronekit platform was used to develop the application for the drone, which includes two processes: the flight process (Section 3.3) and the autonomous landing process (Section 3.2). The autonomous landing is achieved using the OpenCV Library.

3.2. Autonomous landing process. The single board computer (SBC) is used to acquire, process, and analyze digital images to extract information and produce objective data, which provides positioning feedback when the UAV descends.

The first capture device considered was the version 2 camera module of the Raspberry Pi board, as it is optimized for exclusive use with this type of board. The second option was the camera Turnigy HD Wifi Action camera due to its compatibility with the drone's gimbal. The Raspberry Pi camera module has a fixed lens that does not generate distortion in the captures, while the camera Turnigy HD camera has a wide-angle lens that can distort the captures.

Radial distortion causes straight lines to appear curved as they move away from the central focus of the image, while tangential distortion causes certain areas of the image to appear closer than others due to misalignment of the image captured with the capture plane.

The pinhole camera model, which uses intrinsic and extrinsic parameters to map a 3D scene onto a 2D image [31, 32], is represented by Equation (1).

$$\vec{m}' = \bar{A} [\bar{R}|\vec{t}] \vec{M}' \quad (1)$$

where $\bar{A}$ represents the matrix of intrinsic parameters, $\bar{R}|\vec{t}$ represents the matrix of extrinsic parameters or rotation-translation matrix, and $\vec{M}'$ represents a point in the scene.

The intrinsic parameter matrix in Equation (2) does not depend on the scene and can be evaluated once and can be reused for different scenes. The parameters that make up this matrix are

- (f_x, f_y) : focal lengths expressed in pixels
- (c_x, c_y) : center point of image

$$\bar{A} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \quad (2)$$

The extrinsic parameter matrix describes camera movement in a static scene or the movement of an object in front of a fixed camera. As a result, it acts as a dictionary that translates the coordinates of a point M in a scene (X, Y, Z) to a point m in the camera's coordinate system (x, y) .

OpenCV provides functions that determine the intrinsic parameter matrix and the vector of radial and tangential distortion coefficients for any camera compatible with the library. This process requires several test images to extract the camera's characteristics, which are captured from different locations and orientations with the static camera. At least ten images are required to obtain a reliable intrinsic coefficient matrix [33].

The images used in this calibration process require a reference element that facilitates the extraction of characteristics. Checkerboard patterns are commonly used for this purpose. Placing the calibration board in different arrangements makes it possible to determine the points with greater distortion associated with a specific objective. Figure 2 shows some reference positions for capturing calibration images. These poses describe the location of the marker within the captured frame (position) relative to the capture device (depth) and relative to the capture plane (tilt).

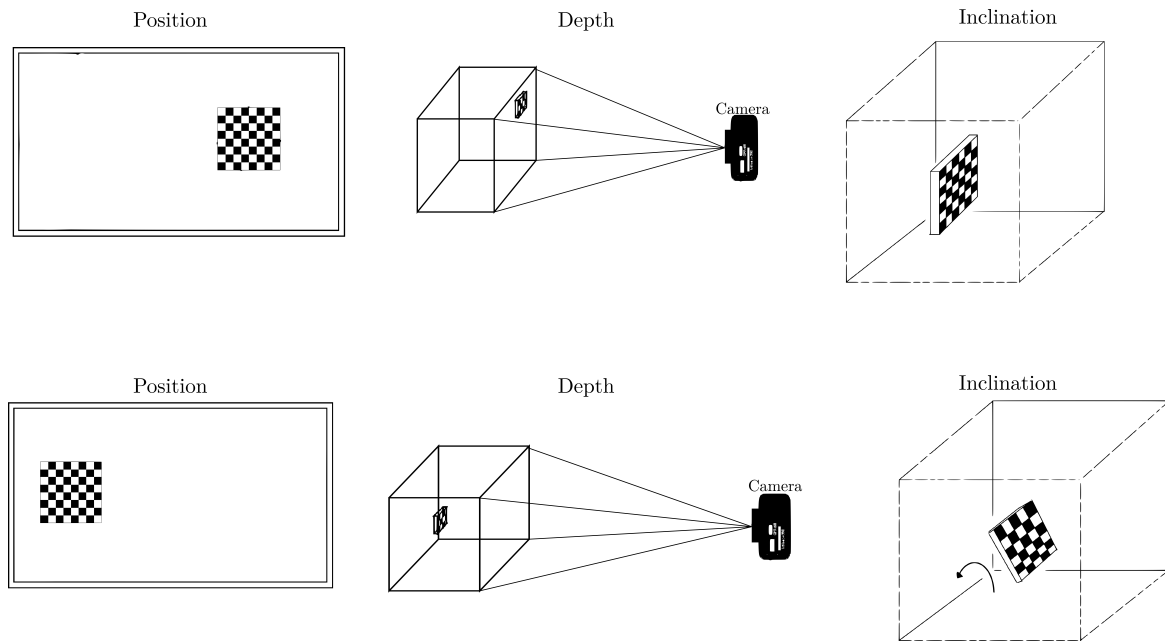


FIGURE 2. Reference postures for the capture of calibration images

Figure 3 shows some captures of the calibration board, with modifications in its position, depth, and inclination. The vertices of the checkerboard are obtained and repositioned to approximate the shape of the reference pattern and calibrate the distorted image. This allows for an estimate of the distortion present in a specific area to be obtained. OpenCV

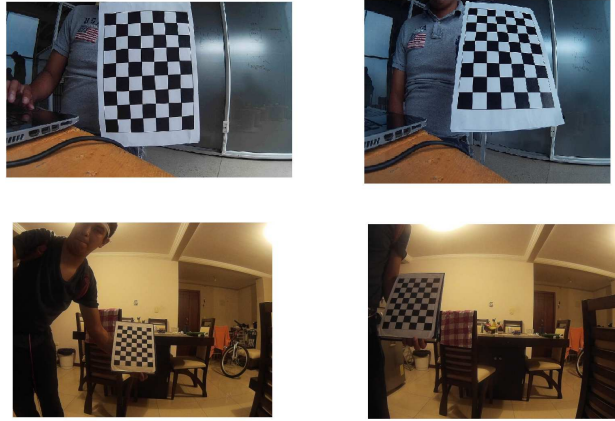
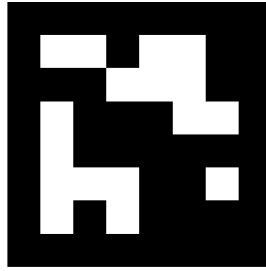


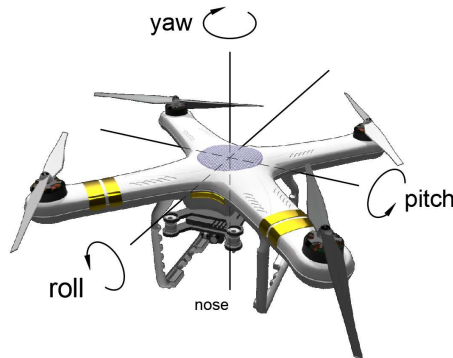
FIGURE 3. Calibration board

uses a model based on the work reported in [32] to obtain the intrinsic matrix and the distortion vector.

The landing pad chosen for this work is an ArUco marker. OpenCV uses functions developed by the Garrido y Jurado library [34] to detect these markers. Figure 4 shows a 6×6 ArUco marker with the identifier (ID) 5.

FIGURE 4. *ArUco* marker 6×6 , $N = 5$

3.3. Flight process. The natural consequence of modifying the UAV's speed is its spatial movement. The orientation of a UAV relative to its center of mass is defined using three angles: pitch, roll, and yaw. Figure 5 illustrates the location of these angles on a quadcopter in *X* configuration, with the front of the vehicle being the nose.

FIGURE 5. *Pitch*, *roll* and *yaw* angles in a *quadcopter*

Two coordinate systems can be used to describe the position and movement of the UAV in space.

- **XYZ:** The body coordinate system of the quadcopter is defined with respect to the vehicle. The X axis is aligned with the longitudinal axis of the drone and points towards its nose, the Y axis is aligned with the lateral axis of the drone and points towards its right side, and the Z axis is aligned with the vertical axis of the drone and points downwards. This coordinate system moves and rotates together with the vehicle.
- **North-East-Down (NED):** The ground coordinate system is defined with respect to a fixed reference point on the ground [35]. This coordinate system is used as a reference frame to describe the position and movement of the UAV with respect to the ground.

Figure 6 illustrates the two coordinate systems.

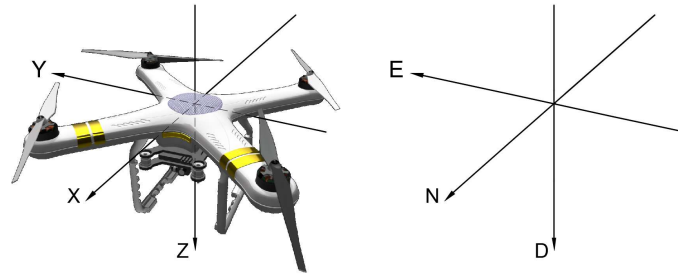


FIGURE 6. Coordinate system XYZ and NED

In this work, the flight process consists of a series of algorithms that are designed to perform specific procedures, including the deployment of a flight plan that can traverse the entire sensor network, visit each node, and return to the take-off place. The flight plan includes landing at each node to collect data, and the algorithms ensure the safety of the UAV during the flight.

First, a data collection algorithm is defined to establish the vehicle's flight plan through the network. The nodes within the network can be identified using a network identifier or by recognizing the associated ArUco marker. However, these markers must be defined and programmed into the flight controller firmware in advance.

The vehicle uses the information provided by the compass and the GPS module to navigate towards the set waypoints. A waypoint is defined by geographic coordinates, which specify the latitude and longitude values of the location. Therefore, to define the UAV flight plan, it is necessary to have the reference information of the waypoints to be reached.

Algorithm 1 describes the sequence of processes that the flight controller or autopilot must follow to reach the location of the nodes, collect data, and return to the starting point. Initially, the algorithm obtains the vehicle's current location (Algorithm 1, line 1), which is taken as the launch point, or HOME location. Next, for a specific node within the list of waypoints or node locations (ListWayPoints), the vehicle must complete the Armed and Takeoff process (Algorithm 1, line 3). Upon reaching a defined altitude (FlightHeight), the autopilot uses the geographical coordinates of a waypoint to start its route using the function DriveVehicle (Algorithm 1, line 4). The vehicle must not carry out another process until it reaches the location of interest. Due to the GPS's precision, it is impossible to reference an exact point, and hence a margin of error is allowed (WaypointMarginError) when estimating the distance between the vehicle's current location point and the target waypoint (Algorithm 1, lines 5 to 9).

Once the vehicle reaches a waypoint, complementary algorithms are started to ensure precision landing and data collection in a node (Algorithm 1, line 11). This process is

Algorithm 1: Data collection

Data: FlightHeight, WaypointList, WaypointMarginError

```

1 HOME = CurrentLocation()
2 for Waypoint in WaypointList do
3   ArmedAndTakeoff(FlightHeight)
4   GuideTheVehicle(Waypoint)
5   while TRUE do
6     dist = GetDistance(CurrentLocation, Waypoint)
7     if dist <= WaypointMarginError then
8       exit
9     end
10  end
11  Next algorithm ...
12 end
13 ArmedAndTakeoff(FlightHeight)
14 GuideTheVehicle(HOME)
15 Landing()
16 DisarmVehicle()

```

performed for each waypoint. When all waypoints are visited, the UAV starts the Arming and Takeoff process, uses the waypoint HOME to return to the takeoff place (function Land), and then performs the Disarm function (Algorithm 1, lines 13 to 16).

The precision error of the GPS module and the MarginErrorWP introduced in Algorithm 1, combined with the height above the ground at which the vehicle is located, do not guarantee that the marker on the platform landing gear platform is within the camera's field of view. The camera's detection area depends on the field of view, which is determined by the vehicle's height. The detection area encompasses the entire surface that can be captured by the camera.

The first complementary algorithm performs the marker localization process using a search area defined by the detection areas. Figure 7 illustrates an example of a location area. This process is governed by distance and velocity vectors that map the entire area until the marker is detected.

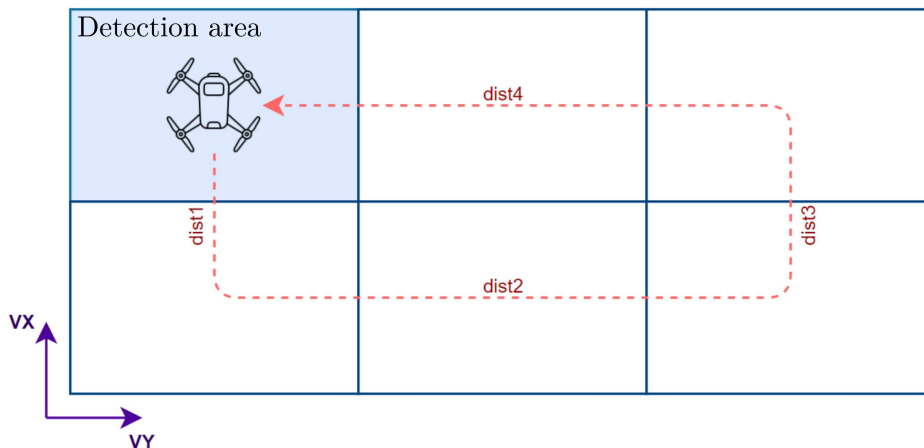


FIGURE 7. Location area

A UAV establishes its course relative to North (0°) according to the point of interest, pointing in the direction of the waypoint in question. However, the marker's orientation on the landing pad may not be the same as the vehicle's heading when advancing to the location. To correct the vehicle's heading, features of the ArUco marker are used to estimate a heading angle that aligns the vehicle with the marker.

The camera's resolution determines the dimension (in pixels) of a captured frame. OpenCV uses a two-axis reference system, XY , with its origin located in the upper left corner of the data frame. Figure 8 shows a detected marker, its orientation relative to the data frame (β), and the orientation of the data frame relative to the NED reference frame. When an ArUco marker is detected, its orientation is indicated by the upper left or guide vertex.

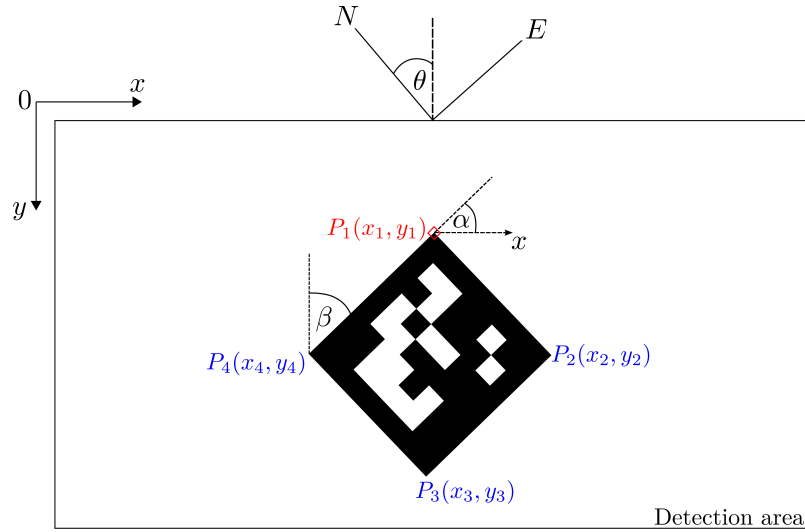


FIGURE 8. Marker orientation relative to data *frame* and NED frame

The marker's orientation relative to the data frame (β) can be determined using the angle of inclination of the line (P_1, P_4, α). The value of α can be obtained using Equation (3). After obtaining α , β can be determined by adjusting the inclination angle based on its sign and the position of the guide vertex relative to the other vertices.

$$\alpha = \arctan \left(\frac{y_2 - y_1}{x_2 - x_1} \right) \quad (3)$$

The position of the guide vertex can be determined by comparing the distances between each vertex and the origin. Algorithm 2 presents the procedure followed to calculate β . Initially, the algorithm finds the vertex closest to the origin (minposition) by determining the index of the smallest distance in the list of vertex-origin distances, ListDistOr (Algorithm 2, line 1). Then, evaluate the sign of α and the position of the vertex closest to the origin to obtain β by adding 90° or 270° , depending on the quadrant in which the vertex is located (Algorithm 2, lines 2 to 14). The angle θ is the yaw angle required to align the vehicle and the marker, and its value is the sum of the vehicle's current heading and β .

The descent of the vehicle is predominantly affected by wind speed. Velocity vectors are computed to approach and descend the vehicle towards the marker. To calculate these vectors, the central point (p) of the capture, a tolerance window or subarea with a width of $2 \times m$ pixels, and four approximation bands (f) are defined, as illustrated in Figure 9. The tolerance window ensures that a range of variation is maintained when positioning the marker's center. The UAV descends to a minimum landing height, initiating an

Algorithm 2: Calculation of β

Data: α , VODistList

```

1 minposition = index(min(VODistList))
2 if  $\alpha < 0$  then
3   if minposition == 1 || minposition == 4 then
4     |  $\beta = 90 + \alpha$ 
5   end
6   if minposition == 2 || minposition == 3 then
7     |  $\beta = 270 + \alpha$ 
8   end
9 else
10  if minposition == 1 || minposition == 2 then
11    |  $\beta = 270 + \alpha$ 
12  end
13  if minposition == 3 || minposition == 4 then
14    |  $\beta = 90 + \alpha$ 
15  end
16 end

```

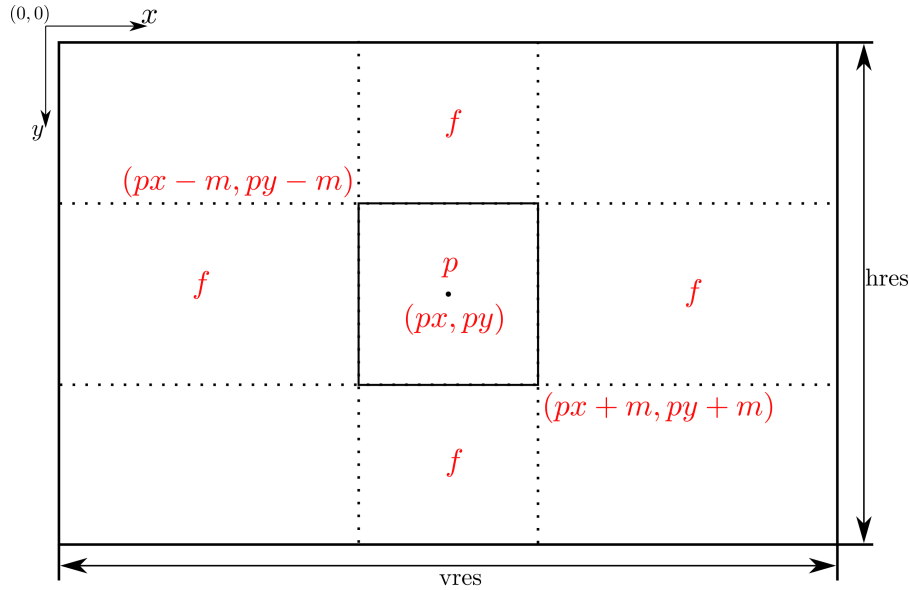


FIGURE 9. Tolerance window

autonomous landing process controlled entirely by the autopilot. The marker's size determines the minimum landing height.

3.4. Network topology and protocol. Figure 10 depicts a simplified diagram of the sensor network. This experiment focuses on the cluster head of a set of nodes and assumes that the aggregation has been completed, as described in the work proposed by [36]. Path planning is not within the scope of this work; a static scenario is used for the experiment, and path planning is built statically. The network topology comprises three components: RaspiUAV, Server, and RaspiNodes.

- **Server:** The DTN node is the endpoint or receiver of the data collected by the UAV. The server location is defined as the *HOME* or take-off point of the UAV.

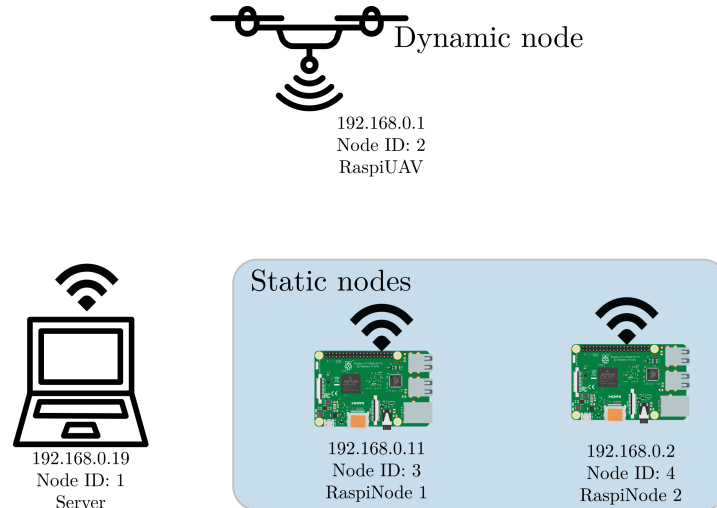


FIGURE 10. System schematic

- **Static node:** It is an SBC board, Raspberry Pi, used as a cluster head for the WSN network. This element establishes communication with the mobile node through a wireless link to transmit data. Additionally, an ArUco marker is associated with each static node, which is used by the artificial vision and flight algorithms to achieve a landing.
- **Mobile node:** The server is a crucial element in the system and acts as a relay for the other nodes in the network. It executes artificial vision algorithms to search and detect the marker and facilitate the landing of the UAV. A Raspberry Pi 3B board is attached to the frame of the RaspiUAV to enable wireless communication with other nodes. The network access point is configured as 192.168.0.0/24.

A DTN network transmits the data from the static nodes (cluster heads) to the server. The UAV departs from the server and visits each network node to collect and store the data onboard. Once the UAV has visited all the nodes, it returns to the starting point and forwards the collected data to the server. Figure 11 illustrates the scenario where the experiment was conducted.

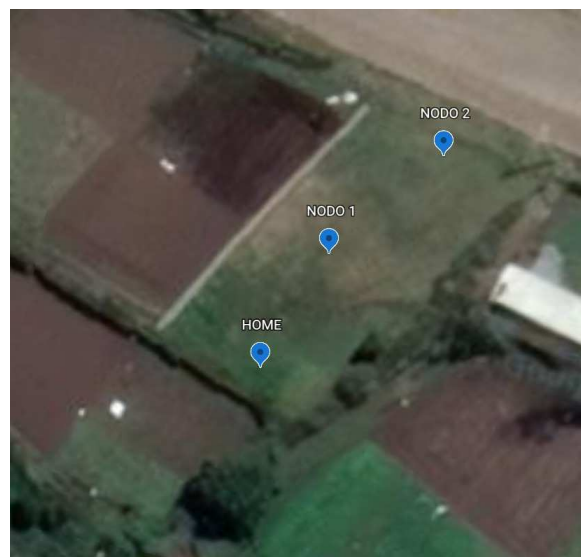


FIGURE 11. Scenario

The DTN is deployed using the project interplanetary overlay network (ION) DTN project. ION-DTN (<https://sourceforge.net/projects/ion-dtn/>) is an open-source implementation of the DTN architecture described in Internet RFC 4838. Data transmission between a static node and the UAV begins when the vehicle executes all flight algorithms for the specific node. The ION-DTN configuration requires defining of contacts with the mobile node.

The DTN nodes are deployed using Docker on Ubuntu and Raspbian Buster operating systems. In the experiment, a computer with an Intel Core i7 processor and 8 GB RAM is used as the server, and a Raspberry Pi 3B+ with 2 GB RAM is used as the mobile node. Contacts are established based on the transfer route, and the IP protocol is used as the transport protocol under port 4556. A static routing group is defined for the static nodes to transfer the data through mobile node or mule.

4. Results.

4.1. Marker detection. Although the Raspberry Pi camera module has not distortion, the Turnigy camera has a wider field of view, capturing a larger detection area than the Raspberry Pi camera at the same distance. To reduce the distortion of the Turnigy camera's capture, several tests were performed using the method described in the previous section. Figure 12 illustrates the capture before and after the distortion correction process for the Turnigy camera. It is worth noting that the distortion correction process reduces the image's vertical resolution and introduces a percentage of error at the edges of the image.



FIGURE 12. Distorted image when captured (left); image after removing distortion (right)

Several tests were conducted in different scenarios to analyze the performance of the capture devices mentioned in Section 3.2.

One test involved placing the capture device at different heights relative to markers of different sizes to determine the maximum detection height per marker size. The results show that the Raspberry Pi module can detect a marker at a higher altitude than the Turnigy camera, as shown in Figure 13.

Another test involved determining the minimum distance required for successful detection based on marker size. This distance determines the minimum landing height. Due to its wide-angle lens, the Turnigy camera allows successful marker detection at a shorter distance than the camera module. This distance influences the precision of the landing since the vehicle's movements at low altitudes allow for a better approach to the marker. Figure 14 illustrates the minimum distance required for successful detection by marker size.

Furthermore, tests were conducted to analyze the influence of lighting on detection. Both natural and artificial lighting scenarios were tested, and the results showed that natural light affects the detection in the Raspberry Pi module, resulting in intermittent detection in consecutive frames, as shown in Figure 15. This behavior occurs for all marker sizes and is not observed when the Turnigy camera is used.

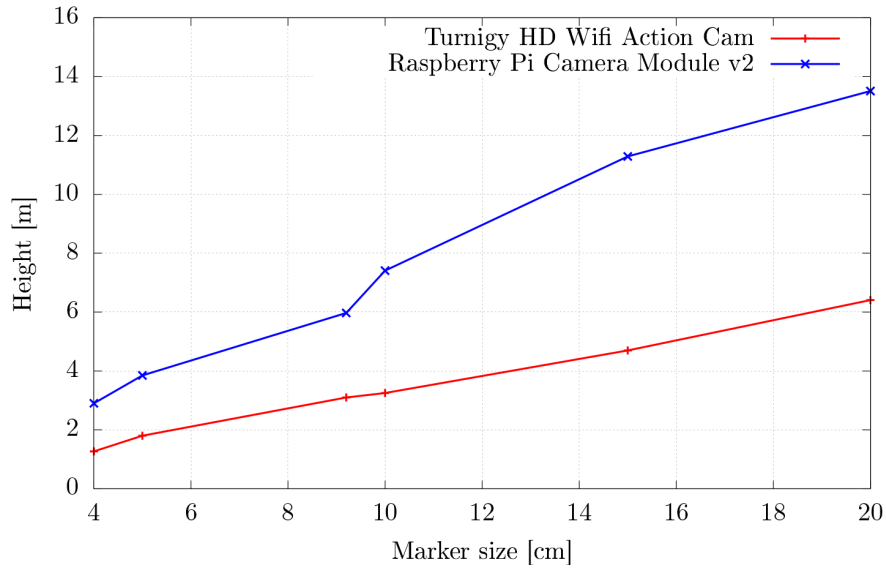


FIGURE 13. Maximum marker detection height ArUco

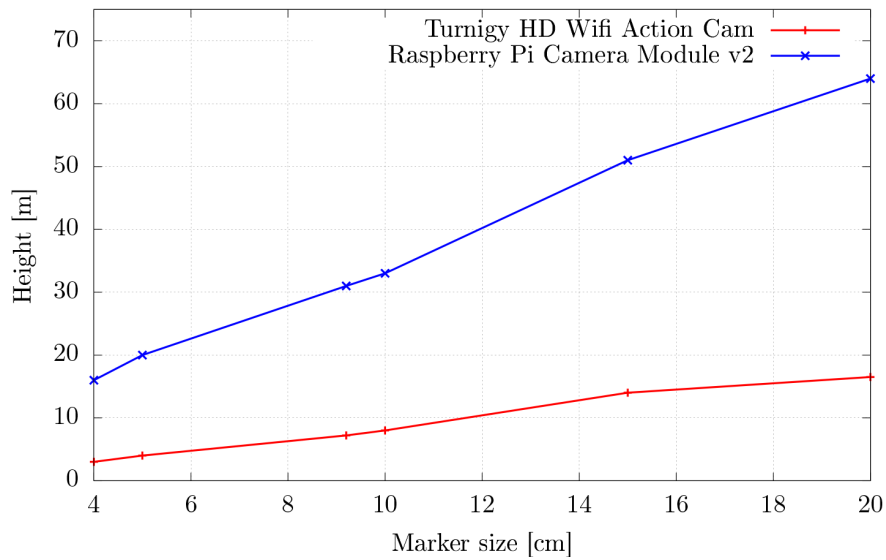


FIGURE 14. Minimum landing height



FIGURE 15. Intermittence in the detection of the marker in consecutive frames caused by the influence of direct natural light on the marker

The Turnigy camera was selected thanks to the results obtained in the marker detection. Figure 16 shows the detection of a 20 cm marker using this device.

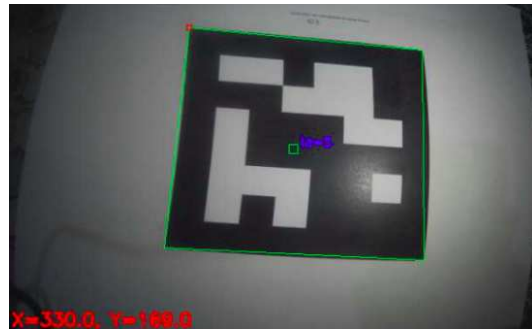


FIGURE 16. Marker detection ArUco using the Turnigy camera in an environment with direct artificial lighting

4.2. Flight algorithms. Tests were conducted using Algorithm 1 with five and seven route points (waypoints) defined. The vehicle initiates the landing process when the distance between the UAV's current location and reference points is within the allowable margin of error. The margin of error was set to 45 cm. If the distance is less than 45 cm, the vehicle remains in flight and cannot perform any other processes. The tests were performed with a backup pilot who controlled the vehicle using radio control, and the pilot could switch off the control provided by the SBC board at any time if an error or problem occurred during the test.

The experiments to evaluate the orientation algorithm focused on taking off the vehicle from the marker, correcting the course, and landing. Table 2 presents ten target yaw values (marker orientations) and the corresponding yaw correction angle yaw calculated using Algorithm 2 for each case. The average error between the target bearing angle and the calculated yaw is 4.3° . The yaw correction on the vehicle is correctly achieved by setting an 8° tolerance margin.

TABLE 2. Heading correction error

Marker orientation $[\circ]$	Calculated yaw $[\circ]$	Error $[\circ]$
50	57	7
24	29	5
345	349	4
130	134	4
176	178	2
212	217	5
232	239	7
278	282	4
20	22	2
355	358	3

A single waypoint flight plan was deployed to measure the influence of the position error on the UAV's landing. The waypoint referred to a marker placed on the ground, and the vehicle took off from a nearby location, headed to the reference location, and landed. Multiple tests were carried out, measuring the distance between the precise location of the reference point and the center of the camera of the vehicle landing site. The results of twelve tests are presented in Figure 17. As shown in the figure, one of these landings was less than 1 m from the benchmark, while most were less than 2 m. The mean distance from the benchmark was 143.2 cm, with a standard error of 0.1 m and a mean square error of 0.34 m.

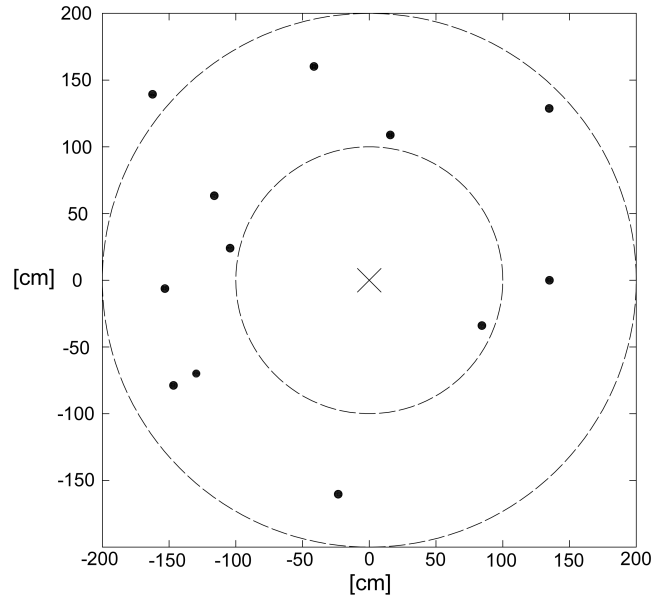


FIGURE 17. Landing the vehicle with LAND mode

The same process was carried out using complementary flight algorithms. The error distance is defined as the distance between the camera's center and the marker. Figure 18 shows the results of twelve landings, out of which nine landings had an error distance of less than 50 cm. Figure 19 shows the landing with the shortest error distance obtained during the development of the tests, which was 3 cm. The average error distance for all the tests was 22.86 cm, with a standard error of 6.3 cm and a mean square error of 21 cm. The landing height that produced the best results was 50 cm. A lower height caused erratic flight, which was caused, in this case, by an error in the flight controller's barometer.

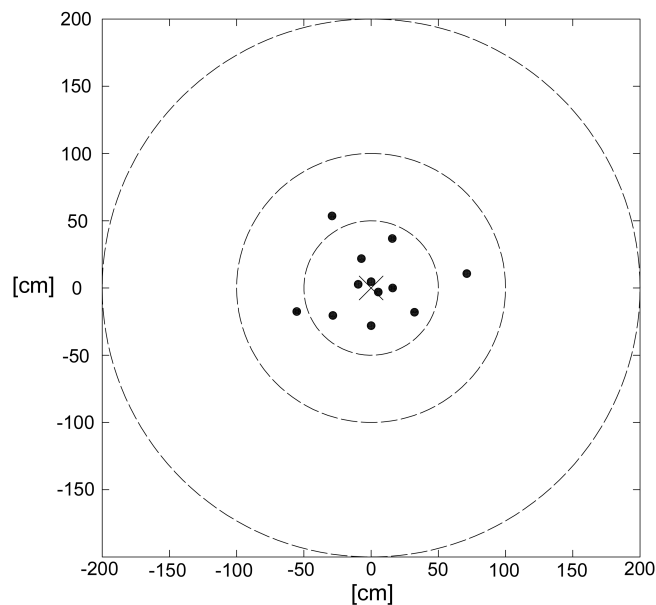


FIGURE 18. Landing using precision landing system



FIGURE 19. Precision landing

4.3. Data collection. The performance tests of ION-DTN were carried out using three DTN nodes, as shown in Figure 11. The separation between mobile and static nodes was not greater than 6 meters in the two scenarios considered:

- Scenario 1: Transmitting information from node 2 to the server (node HOME) through the UAV (node 1) was tested.
- Scenario 2: Transmission of information from node 2 to node 1.

In the experiments, the network data traffic was captured to calculate the maximum delivery time, maximum throughput, and delivery probability to analyze the performance of ION-DTN.

The delivery time was analyzed by sending files of varying sizes in Scenario 1, where the nodes were spaced three meters apart. The tests were conducted within the communication range of the nodes and without disconnection from any of them. Table 3 presents the delivery time interval, file size, and number of packets sent. The delivery time interval refers to the time taken for end-to-end data transmission.

TABLE 3. Delivery time in function of file size

File size [MB]	Delivery time [s]	Number of packets
0.1	2.84	260
0.3	6.37	546
0.5	8.33	595
1.5	24.64	1529
3	40.57	2861
4	59.36	2927

The results demonstrate that the delivery time increases as the file size increases. Figure 20 shows the average delay in packet delivery by file size.

The throughput analysis was conducted in the second scenario with no node disconnection time. Three separation distances between nodes were considered. Figures 21 and 22 show the achieved throughput in data transmission for different separation distances between nodes. When the separation distance is zero, i.e., the vehicle has completed the landing process, the data transmission takes place in less than one second, as shown in Figure 21. On the other hand, when the vehicle is in flight with a separation distance of

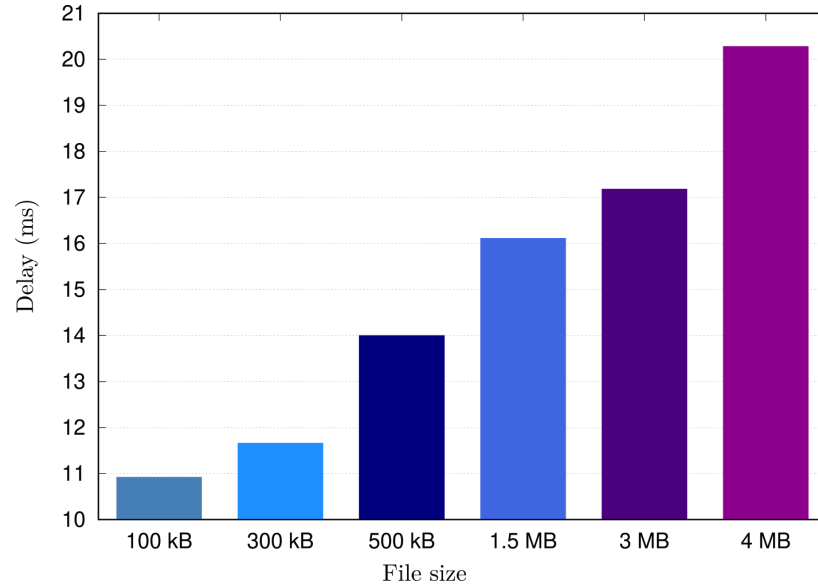
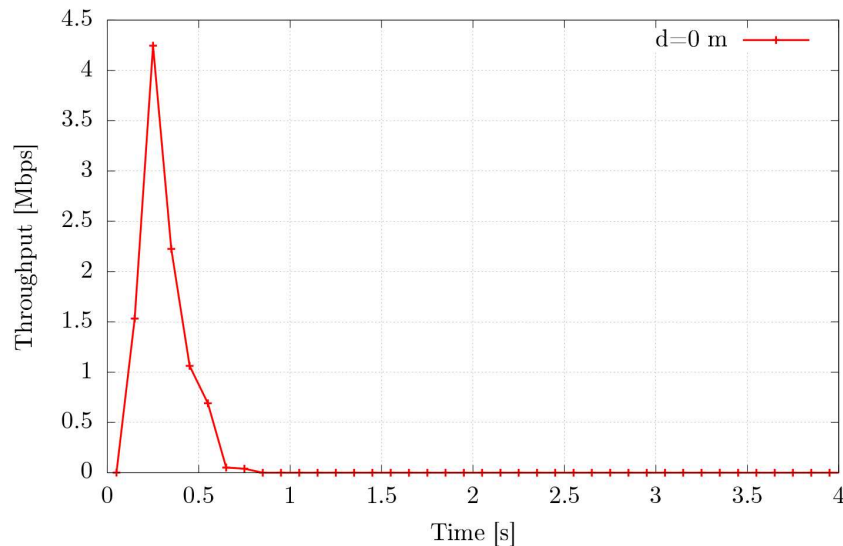


FIGURE 20. Average delivery time in function of file sizes

FIGURE 21. Throughput in sending files for different distances with UAV on land, $d = 0$ m

3 and 6 meters, the transmission of information takes up to 4 seconds due to the retransmissions that occur, as shown in Figure 22. Furthermore, it is observed that the maximum achievable throughput decreases as the separation between nodes increases.

The delivery probability was analyzed in Scenario 1, with modification made to the disconnection time of the Node 1 (server). The disconnection time refers to when the server is not connected to the mobile node because the mobile node is collecting data from a node outside the wireless communication range of the server. Table 4 shows the disconnection times of 1 minute, 3 minutes, and 10 minutes, the delivery time interval, and the number of packets sent for each of the three transmissions per disconnection time.

Different file sizes were sent in each test, and the results showed that the delivery time increased as the file size increased. However, the delivery time interval was less than the time obtained in the average delay experiments. The transmission was made directly from the mobile node, which stored the packets sent from the static node. Additionally, all

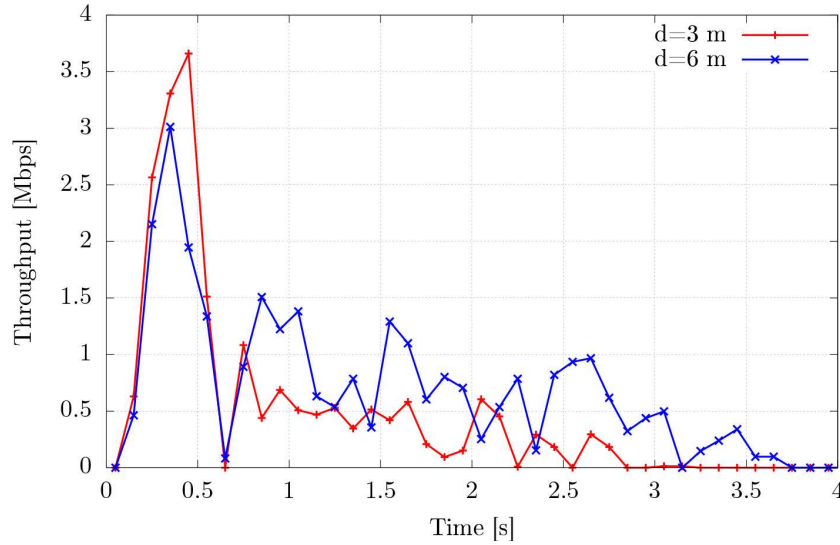


FIGURE 22. Throughput in sending files for different distances with UAV on the flight, $d = 3$ m and $d = 6$ m

TABLE 4. Delivery time interval with disconnection

Disconnection time [min]	Test	Time interval [s]	Number of packets
1	1	0.34	260
	2	1.26	595
	3	3.63	1529
3	1	0.85	260
	2	1.57	546
	3	8.95	2861
10	1	1.16	546
	2	1.65	595
	3	4.47	1529

the files sent arrived at the server without any error, setting the delivery probability to 1, thus verifying that the DTN store and forward process worked correctly in the proposed scenario.

The source code for this project is available on <https://github.com/fabianastudillo/uav-dtn>.

5. Conclusions. In conclusion, the development of applications for systems utilizing unmanned aerial vehicles (UAVs) is accompanied by challenges related to hardware variability, particularly regarding sensor and actuator compatibility with firmware loaded in the controller. This study addressed these challenges by designing a landing algorithm that employs a camera to capture marker images, extract environmental information, and utilize features for precise vehicle control during the landing process. The success of marker detection, a crucial aspect of the landing control system, is influenced by factors such as illumination and detection height, which depend on the capture device's characteristics.

Artificial vision algorithms offer numerous advantages for UAV developing processes, despite the additional power consumption associated with their use. However, integrating artificial vision algorithms into the landing process can potentially enhance precision and reduce energy consumption during information transmission, making it a promising solution.

To mitigate the impact of wind speed and HDOP (horizontal dilution of precision) on flight algorithm performance, it is essential to conduct meticulous sensor calibration and optimize autopilot parameters. Through careful analysis and fine-tuning of algorithm parameters during isolated tests, significant improvements have been achieved in marker detection, orientation correction, and the determination of minimum landing height. The flight algorithms have demonstrated favorable results with minimal errors, with the orientation correction algorithm exhibiting an average error of 4.3° and the landing algorithm maintaining an average distance of 22.86 cm between the camera center and the marker.

Regarding the DTN, the performance analysis of ION-DTN has showcased the feasibility of the data collection process using store and forward mechanism, regardless of the disconnection time from the server. Landing has emerged as the optimal strategy for reducing UAV battery consumption and achieving shorter transmission times at higher data rates.

Despite active development and support from professionals and companies, drones still face limitations, primarily in battery power consumption. The duration and capacity of the battery can vary significantly, with increased size, peripherals, and capacity leading to higher power requirements for vehicle control and lift. Hence, carefully considering the drone's total weight and battery capacity is crucial during its design and configuration.

Future research directions in this field could focus on the following aspects.

- 1) **Improvement of flight algorithm performance under adverse conditions:** Investigating advanced calibration methods and adaptive autopilot algorithms to enhance the resilience of UAV systems against environmental factors like wind speed and HDOP would lead to more reliable and accurate flight operations.
- 2) **Exploration of alternative networking protocols:** While DTN networks have demonstrated potential for UAV data transmission, exploring alternative networking protocols that can offer efficient and reliable communication in challenging environments, such as mesh networks or hybrid communication systems, could further optimize data collection and transmission processes.

By addressing these research directions, the development and utilization of UAV systems can continue to progress, enabling a wide range of applications in various fields, including aerial photography, surveillance, infrastructure inspection, delivery services, and disaster response.

REFERENCES

- [1] J. B. Ernst, Energy-efficient next-generation wireless communications, *Handbook of Green Information and Communication Systems*, Elsevier Inc., pp.371-392, 2013.
- [2] C. Wang, F. Ma, J. Yan, D. De and S. K. Das, Efficient aerial data collection with UAV in large-scale wireless sensor networks, *International Journal of Distributed Sensor Networks*, vol.11, no.11, 286080, 2015.
- [3] ArduPilot, *Copter Home – Copter Documentation*, <http://ardupilot.org/copter/>, Accessed on 2023-02-07.
- [4] S. Fu, L. Zhao, Z. Su and X. Jian, UAV based relay for wireless sensor networks in 5G systems, *Sensors*, vol.18, no.8, 2413, 2018.
- [5] I. Jawhar, N. Mohamed and J. Al-Jaroodi, UAV-based data communication in wireless sensor networks: Models and strategies, *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*, pp.687-694, 2015.
- [6] K. Fall, K. L. Scott, S. C. Burleigh, L. Torgerson, A. J. Hooke, H. S. Weiss, R. C. Durst and V. Cerf, *Delay-Tolerant Networking Architecture*, <https://www.hjp.at/doc/rfc/rfc4838.html>, 2007.
- [7] E. Mahoney, *Disruption Tolerant Networking: Reliable Solar System Internet Connection*, <https://www.nasa.gov/content/dtn>, 2016.
- [8] C. Wang and H. Ma, Data collection in wireless sensor networks by utilizing multiple mobile nodes, *2011 7th International Conference on Mobile Ad-hoc and Sensor Networks*, pp.83-90, 2011.

- [9] D. Herring, *Precision Farming*, <https://earthobservatory.nasa.gov/images/1139/precision-farming>, 2001.
- [10] A. McBratney, B. Whelan, T. Ancev and J. Bouma, Future directions of precision agriculture, *Precision Agriculture*, vol.6, no.1, pp.7-23, 2005.
- [11] M. Alsafasfeh, I. Abdel-Qader, B. Bazuin, Q. Alsafasfeh and W. Su, Unsupervised fault detection and analysis for large photovoltaic systems using drones and machine vision, *Energies*, vol.11, no.9, 2252, 2018.
- [12] T. Zhao and H. Jiang, Landing system for AR.Drone 2.0 using onboard camera and ROS, *2016 IEEE Chinese Guidance, Navigation and Control Conference (CGNCC)*, pp.1098-1102, <http://ieeexplore.ieee.org/document/7828941/>, 2016.
- [13] P. Smyczynski, L. Starzec and G. Granosik, Autonomous drone control system for object tracking: Flexible system design with implementation example, *2017 22nd International Conference on Methods and Models in Automation and Robotics (MMAR)*, pp.734-738, <http://ieeexplore.ieee.org/document/8046919/>, 2017.
- [14] M. F. Sani and G. Karimian, Automatic navigation and landing of an indoor AR. drone quadrotor using ArUco marker and inertial sensors, *2017 International Conference on Computer and Drone Applications (ICoNDA)*, pp.102-107, <http://ieeexplore.ieee.org/document/8270408/>, 2017.
- [15] K. Fujii, K. Higuchi and J. Rekimoto, Endless flyer: A continuous flying drone with automatic battery replacement, *Proc. of IEEE 10th International Conference on Ubiquitous Intelligence and Computing (UIC 2013) and IEEE 10th International Conference on Autonomic and Trusted Computing (ATC 2013)*, pp.216-223, <http://ieeexplore.ieee.org/document/6726212/>, 2013.
- [16] V. Sudevan, A. Shukla and H. Karki, Vision based autonomous landing of an unmanned aerial vehicle on a stationary target, *2017 17th International Conference on Control, Automation and Systems (ICCAS)*, pp.362-367, <http://ieeexplore.ieee.org/document/8204466/>, 2017.
- [17] J. Janousek and P. Marcon, Precision landing options in unmanned aerial vehicles, *2018 International Interdisciplinary PhD Workshop (IIPhDW)*, pp.58-60, <https://ieeexplore.ieee.org/document/8388325/>, 2018.
- [18] K. T. Putra, R. O. Wiyagi and M. Y. Mustar, Precision landing system on H-Octocopter drone using complementary filter, *2018 International Conference on Audio, Language and Image Processing (ICALIP)*, pp.283-287, <https://ieeexplore.ieee.org/document/8455553/>, 2018.
- [19] X. Xu, Z. Wang and Y. Deng, A software platform for vision-based UAV autonomous landing guidance based on markers estimation, *Science China Technological Sciences*, vol.62, pp.1825-1836, 2019.
- [20] Y. Liu, H. Bei, W. Li and Y. Huang, Survey of UAV autonomous landing based on vision processing, in *Advances in Intelligent Networking and Collaborative Systems. INCoS 2020. Advances in Intelligent Systems and Computing*, L. Barolli, K. Li and H. Miwa (eds.), Springer, Cham, 2021.
- [21] L. Xin, Z. Tang, W. Gai and H. Liu, Vision-based autonomous landing for the UAV: A review, *Aerospace*, vol.9, no.11, 634, 2022.
- [22] C. Giannini, A. A. Shaaban, C. Buratti and R. Verdone, Delay tolerant networking for smart city through drones, *2016 International Symposium on Wireless Communication Systems (ISWCS)*, pp.603-607, <http://ieeexplore.ieee.org/document/7600975/>, 2016.
- [23] N. Uchida, N. Kawamura, T. Ishida and Y. Shibata, Proposal of autonomous flight wireless nodes with delay tolerant networks for disaster use, *2014 8th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, pp.146-151, <http://ieeexplore.ieee.org/document/6975455/>, 2014.
- [24] T. Matsumi, A. G. Ramonet and T. Noguchi, UAV aided search and rescue using delay tolerant networks, *IEICE Technical Report*, vol.119, no.110, pp.185-190, 2019.
- [25] C. Caillouet, F. Giroire and T. Razafindralambo, Efficient data collection and tracking with flying drones, *Ad Hoc Networks*, vol.89, pp.35-46, 2019.
- [26] C. Wang, F. Ma, J. Yan, D. De and S. K. Das, Efficient aerial data collection with UAV in large-scale wireless sensor networks, *International Journal of Distributed Sensor Networks*, vol.11, no.11, 286080, <http://journals.sagepub.com/doi/10.1155/2015/286080>, 2015.
- [27] J. R. Martinez-De Dios, K. Lferd, A. De San Bernabé, G. Núñez, A. Torres-González and A. Ollero, Cooperation between UAS and wireless sensor networks for efficient data collection in large environments, *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol.70, nos.1-4, pp.491-508, 2013.

- [28] H. Liang, W. Gao, J. H. Nguyen, M. F. Orpilla and W. Yu, Internet of Things data collection using unmanned aerial vehicles in infrastructure free environments, *IEEE Access*, vol.8, pp.3932-3944, 2019.
- [29] O. Bouhamed, H. Ghazzai, H. Besbes and Y. Massoud, A UAV-assisted data collection for wireless sensor networks: Autonomous navigation and scheduling, *IEEE Access*, vol.8, pp.110446-110460, 2020.
- [30] C. Dragana, G. Stamatescu, L. Ichim and D. Popescu, Interlinking unmanned aerial vehicles with wireless sensor networks for improved large area monitoring, *2017 4th International Conference on Control, Decision and Information Technologies (CoDIT)*, pp.359-364, 2017.
- [31] OpenCV, *Camera Calibration and 3D Reconstruction – OpenCV 2.4.13.7 Documentation*, https://docs.opencv.org/2.4/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html, Accessed on 2023-02-07.
- [32] Z. Zhang, A flexible new technique for camera calibration, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.22, no.11, pp.1330-1334, 2000.
- [33] A. Mordvintsev and K. Abid, *OpenCV-Python Tutorials Documentation Release 1*, vol.2, pp.209-210, https://opencv-python-tutroals.readthedocs.io/_/downloads/en/latest/pdf/, 2017.
- [34] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas and M. J. Marín-Jiménez, Automatic generation and detection of highly reliable fiducial markers under occlusion, *Pattern Recognition*, vol.47, no.6, pp.2280-2292, 2014.
- [35] P. Kumar, R. Kumar, S. Anand, E. N. Ganesh and V. Prithiviraj, Quad band signal strength monitoring system using quadcopter and quad phone, *Journal of Green Engineering*, vol.5, pp.1-22, 2015.
- [36] S. Wan, Y. Zhang and J. Chen, On the construction of data aggregation tree with maximizing lifetime in large-scale wireless sensor networks, *IEEE Sensors Journal*, vol.16, no.20, pp.7433-7440, 2016.

Author Biography



Edisson Paul Cabrera received his Electronics and Telecommunications Engineering degree from the University of Cuenca, Ecuador, in 2021. He works on IT and his current research interests include network infrastructure, information security, and data analysis.



Jefferson Santiago Agila received his Electronics and Telecommunications Engineering degree from the University of Cuenca, in 2021, Ecuador. He works on broadband networks, fiber optics, data networks, and data analysis. His research interests focus on data analysis, machine learning, and IoT focused on IPv6 addressing.



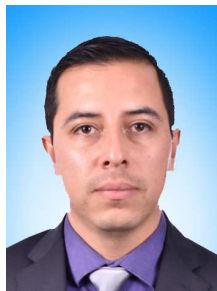
Fabian Astudillo-Salinas received the B.S.E. (C.S.) degree from University of Cuenca, Cuenca, Ecuador, in 2007, and the M.S. and Ph.D. degrees from the Institut National Polytechnique de Toulouse, Toulouse, France, in 2009 and 2013, respectively. Since 2013, he has been a full-time researcher with the Department of Electrical, Electronics and Telecommunications, University of Cuenca, Cuenca, Ecuador. His research interests include network coding, wireless sensor networks, vehicular networks, networked control systems, simulation of networks, performance of networks, cybersecurity and HPC.



Andres Vazquez-Rodas received the Electronics Engineering degree in 2004 from the Universidad Politécnica Salesiana, Cuenca – Ecuador, the Master degree in Telematics Engineering (Honors) from the University of Cuenca – Ecuador in 2010, and the Ph.D. degree from the Networking Department of the Universitat Politècnica de Catalunya BarcelonaTech (UPC), Spain in 2015. He was also an assistant professor at the Universidad Politécnica Salesiana until 2017. Since 2015 he is a full-time professor of the University of Cuenca at the Department of Electrical, Electronics and Telecommunications (DEET) and the Engineering Faculty. His research interests include wireless mesh networks, wireless sensor networks, industrial networking, mobile networks, and complex systems.



Santiago Patricio Torres is currently a full-time professor at the University of Cuenca, in Ecuador. He holds a Ph.D. in Electrical Engineering since 2007. Dr. Torres has been in postdoctoral research stays at the University of Campinas (2010-2013), in Brazil, Cornell University in the US (2011), and KU Leuven (2021, 2023). His research interests lie in the expansion planning of electric energy systems and artificial intelligence applications to engineering problems. Dr. Torres is IEEE Senior Member, and also a member of the IEEE PES Analytic Methods for Power Systems and the IEEE Grid Enhancing Technologies (GETs) working groups.



Ismael Minchala-Avila received his B.S.E.E. degree in 2006 from the Salesian Polytechnic University, Ecuador, and his M.Sc. and Ph.D. degrees from Tecnológico de Monterrey, Mexico, in 2011 and 2014, respectively. From summer 2012 to summer 2013, he was a visiting scholar at Concordia University in Montreal, Canada. Between 2017-2018 he was a Postdoctoral Fellow at Tecnológico de Monterrey in the Climate Change Research Group. Dr. Minchala has authored over 60 indexed publications, including journal articles, conference proceedings, book chapters, and a book.