Towards real-time UAV path replanning based on photogrammetry and learning-based approaches

Débora Paula Simões[1,3] and Henrique Candido de Oliveira[2,3]

[1] Federal Institute of South of Minas Gerais, inconfidentes, Brazil,
[2] Department of Geotechnics, Geomatics, and Mobility, University of Campinas, Campinas, Brazil, hcandido@unicamp.br
[3] Graduate Program in Civil Eng., School of Civil Eng., Architecture, and Urban Design, University of Campinas, Brazil

Keywords: Drone, Path planning, Deep learning, Monoplotting, SDK application.

Abstract

Unmanned Aerial Vehicles (UAVs) have contributed to a wide range of applications, becoming faster and more sustainable nowadays. However, given the significant increase in the number of UAVs, concerns regarding operational safety have grown. Autonomous UAV path planning must ensure compliance with safety requirements. This study proposes a real-time path replanning method focused on ensuring compliance with regulations governing UAV operations. Considering no-fly zones (NFZs) defined by both static (buildings) and dynamic (people) obstacles, a low-cost and replicable solution was implemented in four main steps: 3D offline path planning using the A* algorithm and Digital Elevation Models; human detection in UAV imagery using the YOLO11m model; estimation of the person’s 3D coordinates using Monoplotting; and experiments of real-time path replanning. During flight execution, imagery acquired by the UAV is transmitted to a server and, if a person is detected, path replanning is performed. The replanned route is then sent to the UAV controller to be executed via an SDK-based application. For flights at reduced speeds, the proposed method demonstrated feasibility in a computational environment (replanning time of 2.79 s). Simulated flight execution using the DJI Mobile SDK was successful. However, when relying on data transmission over Wi-Fi, the replanning duration on a local server (17.96 s) remained unsuitable for real-time operations. As future work, alternative solutions should be explored to ensure real-time processing. Despite the challenges, this study contributes by validating the open and free DJI MSDK application for path execution in a simulated environment, integrated with a listener application.

1. Introduction

Unmanned Aerial Vehicles (UAVs), popularly known as drones, have increasingly contributed to a wide range of applications, making them faster, more cost-effective, and sustainable. In addition to being adopted for well-established tasks such as mapping (Liu et al., 2025) and precision agriculture (Haque et al., 2025), these platforms have stood out in logistics (Aldao et al., 2025), search and rescue (Soorki et al., 2025), and surveillance and public safety (Park et al., 2025), driven by the development of efficient and intelligent hardware and software. As a result of the significant advancement of Artificial Intelligence over the past decade, autonomous UAV navigation and exploration have become increasingly reliable, particularly due to the use of deep learning algorithms.

In this context, autonomous path planning for VLOS (Visual Line of Sight) and BVLOS (Beyond Visual Line of Sight) operations must be carefully designed to ensure the safety of airspace, as well as people, animals, and properties within the operation area. In urban environments, the use of autonomous aerial platforms in BVLOS operations raises even greater concern. For this reason, regulations governing UAV operations and airspace access have been established worldwide (Stöcker et al., 2017). Although legislation varies between countries, common concerns include maximum flight altitudes and flying over non-involved people. Regarding the latter, some countries define a minimum horizontal distance between a person and the vertical projection of the UAV on the ground - such as in Brazil (ANAC, 2023) and Australia (Australian Government, 2023), where this distance must be 30 m. Since safe mission completion relies on proper path planning (Hu et al., 2023), it must therefore comply with the limitations imposed by regulations.

Given the complexity of urban environments and the need for spatially accurate representations, research has focused on dynamic and three-dimensional UAV path planning (Luo et al., 2024). Dynamic path planning addresses the challenge of replanning the UAV trajectory according to changes in flight conditions and the 3D environment (Luo et al., 2024), including the presence of moving obstacles, which characterizes online (real-time) path planning (Golabi et al., 2023). Several studies have proposed real-time UAV path planning methods (Peng and Cheng-Yu, 2023, He et al., 2023, Hu et al., 2023). However, although these authors consider dynamic obstacle detection, they do not address compliance with airspace access rules within their approaches.

Unlike most research on UAV path planning, this study proposes an exploration of a real-time path replanning method focused on operational safety. No-fly zones (NFZs) defined around static obstacles, such as previously mapped buildings in the area of interest, as well as dynamic obstacles (people detected along the UAV trajectory), are considered in the path planning process. To develop a replicable and low-cost method, this work includes: (i) the use of a heuristic technique (A* algorithm) for prior (offline) UAV path planning considering static obstacles mapped in a high-resolution orthomosaic; (ii) the three-dimensional analysis of the planned path using a point cloud obtained from an Airborne Laser Scanning (ALS) System; (iii) the real-time detection of dynamic obstacles (people) along the path using deep learning models; (iv) the adoption of the Monoplotting photogrammetric process to determine the localization of dynamic obstacles; and (v) the use of a free mobile Software Development Kit (SDK) application for mission execution.

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-XI-2-2026-741-2026 | © Author(s) 2026. CC BY 4.0 License.

The initial proposal of the real-time UAV path planning method was briefly presented in Simões et al. (Simões et al., 2025). However, the authors did not consider the implementation of a lowest-risk route when NFZs around buildings and people intersect, as addressed in the present study. Moreover, two computational experiments were conducted to evaluate the proposed method. In contrast, the main contribution of the present study is the development of a complete end-to-end architecture that integrates the DJI Mobile SDK, a listener application, and a FastAPI/Papermill server, enabling experimental evaluations within a simulated environment. Experiments were conducted using DJI Assistant 2 (Enterprise Series), with a minimum number of observations sufficient to ensure statistically reliable results regarding the time required for UAV path replanning and its feasibility for real-time execution in UAV operations.

The content of this paper is structured as follows: Section 2 presents details of each step of the proposed real-time UAV path replanning method. The qualitative and quantitative results obtained from the implementation are discussed in Section 3. Final considerations and suggestions for future work, highlighting the challenges encountered, are provided in Section 4.

2. Material and Methods

The proposed method for real-time path planning ensures UAV operational safety through four main stages: 3D offline path planning (offline processing – Section 2.1), human detection in UAV imagery, estimation of the person’s 3D coordinates, and real-time path replanning (online processing – Section 2.2). All algorithms for each stage are implemented in Python within a Jupyter Notebook environment. UAV flight execution is carried out using two applications implemented in Kotlin within Android Studio: a listener application and the official DJI Mobile Aircraft SDK (MSDK) (DJI Developer, 2025b) - described in Section 2.3.

2.1 Offline processing

The initial UAV path is planned prior to flight execution (offline path planning). To ensure path safety and maintain the minimum distance from buildings - as defined by airspace regulations - a minimum path between the origin and destination points is generated. This is done using the A* algorithm while avoiding the no-fly zones (NFZs) around buildings. In addition, a 3D path analysis is performed based on a Digital Surface Model (DSM) - a LiDAR point cloud with a density of 10 points/m^2 acquired by an Airborne Laser Scanning system - and a Digital Terrain Model (DTM), obtained by filtering the DSM using LAStools software (Rapidlasso, 2024). This analysis ensures a safe vertical separation between the UAV and the surface while preventing violations of the maximum permitted flight altitude (120 m). As a result of this process, an offline route composed of a set of straight-line segments is obtained.

From this initial route, flight execution is initiated using the DJI Mobile SDK. As this application requires a route file with specific configurations (DJI Developer, 2025a), an additional pre-flight step (offline processing) involves generating this file, which contains the initial segment of the path to be followed by the UAV. This process also generates data required for executing real-time path replanning (Figure 1).

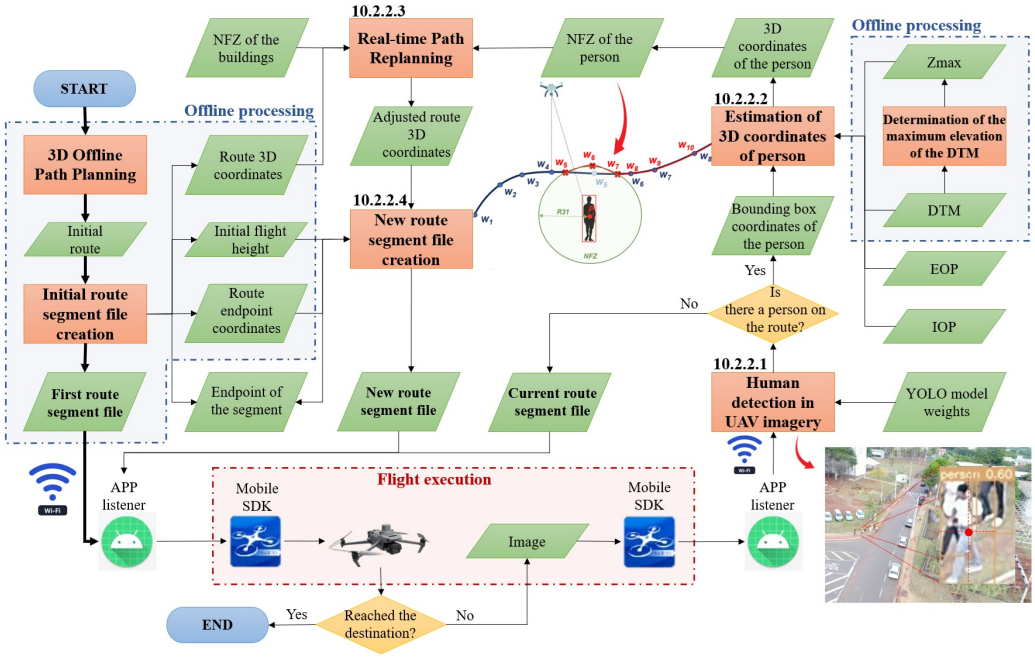

Figure 1. Method for real-time safe path replanning for UAVs.

2.2 Online processing

Although the preliminary UAV path planning ensures three-dimensional safety by avoiding building NFZs, there is still concern for individuals who are uninvolved or unaware of the UAV operation. The online path replanning process aims to ensure a safe distance between the UAV and any person detected along the flight path. Figure 1 illustrates the stages of the proposed method for real-time path replanning. The following subsections describe each stage in detail, including the inputs and outputs obtained at each step.

2.2.1 Human detection in UAV imagery

The UAV path is executed in segments by the DJI MSDK application (DJI Developer, 2025b). The first segment, generated

through offline processing (Section 2.1), is responsible for capturing an image from which individuals along the flight path are detected. For human detection in UAV-acquired images, the YOLO11m model (Jocher and Qiu, 2024) was adopted. This model was trained using the Unicamp-UAV dataset (Simões et al., 2026a). In this work, Simoes et al. (2026) evaluated this dataset considering the performance of the latest YOLO models: YOLOv7, YOLOv8m, YOLOv9m, YOLOv10m, and YOLO11m. Among these models, YOLO11m achieved the highest AP50 score and demonstrated suitability for real-time applications; therefore, it was selected for use in this study.

For comparison purposes, the YOLO12m model (Tian et al., 2025) was trained using the same dataset. Table 1 presents the training results. The experiments were carried out on a workstation equipped with 16 GB of RAM, an Intel Core i7-12700H 2.30 GHz processor, and an NVIDIA GeForce RTX 3060 GPU with 6 GB of RAM. Although YOLO12m achieved higher inference speed (FPS), YOLO11m was adopted in this study as it attained a slightly higher AP50. Since both models satisfy real-time constraints (low inference time), the model achieving the higher AP was selected, as AP is regarded as the primary metric for evaluating the performance of general-purpose object detectors (Akshatha et al., 2023). The definition of metrics used can be observed in (Simões et al., 2026b)

	Model	YOLO11m	YOLO12m
Train	P	0.824	0.865
	R	0.640	0.615
	F1-score	0.720	0.719
	AP50	0.777	0.770
Test	P	0.822	0.818
	R	0.637	0.627
	F1-score	0.718	0.710
	AP50	0.774	0.768
	Inference (ms)	13.6	10.3
	FPS	51.6	68.5

Table 1. Comparison between the training results of the YOLO11m and YOLO12m models.

The result of the YOLO model inference consists of the normalized image coordinates (x_{norm}, y_{norm}) of the center of the bounding box for each detected person, as well as the bounding box width and height. For the path replanning method, only the person closest to the UAV is considered. Therefore, only the detection with the highest y_{norm} value is selected. This is because the image is captured at a 45° inclination angle, always in front of the UAV. Under these conditions, the person closest to the UAV corresponds to the bounding box center with the largest row value among all detections (Figure 2).

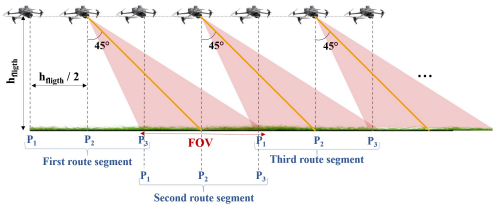

Figure 2. Flight execution by route segments.

2.2.2 Estimation of 3D coordinates of person

When a person is detected, it is necessary to determine their three-dimensional coordinates. To this end, the photogrammetric Monoplotting procedure (Fluehler et al., 2005) is employed. In this process, the Inverse Collinearity Equations (Mikhail et al., 2001) are used to calculate the planimetric coordinates (X and Y). Therefore, the Interior Orientation Parameters (IOP) of the sensor used to capture the UAV images are required, specifically the principal point in the image (x_0, y_0) and the focal length (f) - acquired from a pre-flight camera calibration. The Exterior Orientation Parameters (EOP) - X_0, Y_0, Z_0, κ, ϕ, ω - of the image in which the person was detected are also necessary, possible to be obtained from UAV GNSS and IMU sensors.

For the calculation of the initial coordinates (X_i and Y_i), the maximum elevation of the DTM is adopted ($Z_i = Z_{max}$). Since Z_{max} is unique for the study area, this value is defined before flight execution (the “Maximum elevation of the DTM” offline process - Figure 1). Through an iterative process, for each new position (X_i, Y_i) determined by the Inverse Collinearity Equations, a new Z_i value is defined by interpolation with the DTM, using the “lascontrol” function of the LAStools software (Rapidlasso, 2024), until a threshold is reached (Fluehler et al., 2005).

This stage results in the three-dimensional coordinates of the person closest to the UAV, as detected by the YOLO11m model. Simões et al. (2025) evaluated the planimetric accuracy of coordinates obtained through the monoplotting process. Considering images acquired at different flight altitudes (10, 20, 30, and 40 m) with a camera tilt angle of approximately 45, the average planimetric error was 1 m (Simões et al., 2025). The same average planimetric error was reported by Simões et al. (2023) when analyzing eight different positions in an oblique UAV image acquired at an altitude of 40 m (Simões et al., 2023). In this context, this average error is taken into account when defining NFZs around the detected person along the UAV path in order to ensure compliance with the minimum separation distance established by regulatory requirements.

2.2.3 Safe path replanning for UAV

Considering the initially defined 3D UAV path, the 3D coordinates of the person closest to the UAV, and the NFZs around buildings, real-time path replanning is performed. The Python library Shapely was adopted for implementing path replanning. It allows the execution of geometric operations similar to PostGIS outside a relational database management system, supporting fundamental geometric object types: Point, LineString, and Polygon (Gillies, 2025).

From the detected person’s position, a buffer is created corresponding to the NFZ around it (Figure 1), which the UAV must avoid to ensure operational safety. If the previously defined 3D UAV path intersects with the person’s NFZ, the path is replanned to circumvent the corresponding NFZ. For the first real-time update, the prior 3D path corresponds to the initial path determined during offline processing (before the flight). For subsequent replanning steps, this path corresponds to the output of the previous replanning, i.e., the path generated in the last update.

However, only circumventing the NFZ of the detected person may bring the new path too close to nearby buildings, compromising safety. Therefore, if the overlap between the person’s NFZ and building NFZs exceeds a predefined threshold - considering the limits established by local airspace regulations - the person’s NFZ is reduced to guarantee a minimum horizontal distance between the UAV and buildings. For example,

in Brazil, where the NFZ for buildings and people is 30 m (DECEA, 2023, ANAC, 2023), if the overlap between the NFZs exceeds 500 m^2, the person's NFZ buffer is reduced to 20 m. This ensures a minimum distance of 10 m to buildings.

In all analyses, the shortest path was maintained. That is, when dividing the NFZ into two parts based on its intersection with the prior 3D path, the UAV always circumvents the NFZ along the shortest direction. After the 2D analysis, the replanned path is generalized using the Ramer-Douglas-Peucker algorithm (Ramer, 1972, Douglas and Peucker, 1973), adopting the same generalization parameter of the offline stage (Section 2.1). Finally, altitude values are assigned to each waypoint of the replanned path. If the waypoint already existed in the prior 3D path, the corresponding altitude is assigned. Otherwise, the altitude of the previous waypoint is used.

Algorithm 1 summarizes the analyses required for real-time UAV path replanning, resulting in the new three-dimensional path.

Algorithm 1 Real-time UAV path replanning

if Previous route intersect NFZ_{person} **then**
  Calculate intersection area between NFZ_{person} and $NFZ_{buildings}$
  if Intersection area $>$ Maximum overlap **then**
    Redefine NFZ_{person}
  end if
  Find the shortest path around NFZ_{person}
  Plan a new route
else
  New route = previous route
end if
Generalize the current 2D route
for i in range (number of waypoints 2D route) **do**
  if $waypoint_i$ $\exists$ previous route **then**
    Assign the corresponding altitude value
  else
    Altitude of $waypoint_i$ = Altitude of $waypoint_{i-1}$
  end if
end for

If no person is detected along the flight path - that is, if the results of the previous steps described in Sections 2.2.1 and 2.2.2 are null - the path replanning stage is omitted.

2.2.4 New route segment file creation

After replanning the UAV path, it is necessary to prepare the file to be sent to the UAV and read by the SDK application for flight execution. To ensure person detection along the route and the corresponding path replanning when necessary, the UAV receives route segments whose length corresponds to the minimum distance required by current legislation between a person and the UAV's ground projection. Each segment consists of three waypoints, so that when the UAV reaches the second waypoint, a new image is captured. Until the UAV reaches the end of the segment, path replanning occurs in time to safely initiate a new segment, avoiding people and buildings (Figure 2).

In the proposed approach, the UAV flight height is defined based on the minimum safe distance between the aircraft and individuals. This is because the images in which people are detected are captured during flight. Since the dataset used to train the YOLO11m model consists of images acquired with a 45° camera tilt angle, the same tilt configuration is recommended in this study (Figure 2) to ensure accurate human prediction in UAV imagery (Section 2.2.1). Therefore, to guarantee complete coverage along the UAV trajectory and enable detection of dynamic obstacles, the flight height is defined as the minimum distance from the UAV to people as required by national UAV operational regulations.

In this context, once the 3D coordinates of the replanned path are available, the route is subdivided into segments with a length corresponding to the flight height. The geodetic coordinates (latitude and longitude) of the waypoints defined every “$h_{flight}/2$” meters (Figure 2) are then determined.

Since the SDK application considers flight height relative to the ground, the flight height takes the altitude of the first waypoint into account (defined during the offline path planning - Section 2.1). To maintain a consistent flight height along the route, any difference - positive or negative - between the altitude of the first waypoint and the subsequent waypoints is added to the initial user-defined flight height.

To determine the three waypoints that will compose each route segment, it is necessary to know the last waypoint of the previous route segment. This waypoint will serve as the first waypoint of the next segment. From this point, the two subsequent vertices are selected. These three waypoints constitute the new segment to be sent and executed by the UAV.

For the first path replanning, the last waypoint of the previous segment comes from the file creation step for the initial route segment, executed prior to flight (Figure 1). This offline step (Section 2.1) also generates the first segment to be traversed by the UAV, initiating the route with the three initial waypoints of the offline planned route. For each segment defined in the online stage (Section 2.2.4), the last waypoint of the segment is saved for use in defining the next segment during subsequent path replanning. In some cases, depending on the person's position along the route and the NFZ generated around it, the last waypoint of the previous segment may not be included in the newly replanned path. In this case, to ensure the flight continues while avoiding the person as quickly as possible, the first waypoint of the new segment is defined as the point closest ahead of the last waypoint of the previous segment.

The coordinates of the final point (destination) of the UAV path are saved during the offline process (Section 2.1). This information is crucial to ensure that the last segment of the path is generated correctly. The last segment has distinct configurations compared to the others, such as no image capture - since the route reaches its end - and the UAV landing configuration upon arrival at the destination.

For each route segment, a file is created following the route file format required by the SDK application used for flight execution. In this study, the DJI MSDK v5.15 application is adopted, which uses the WPML (WayPoint Markup Language) route file format (DJI Developer, 2025a). According to this standard, all route files have the extension “.kmz”.

2.3 Flight execution

For UAV flight execution, the DJI MSDK v5.15 application (DJI Developer, 2025b) was adopted. By default, the application provides several tools, including Wayline Management - referred to as Waypoint within the app - a function for automatic aircraft operations (DJI Developer, 2025b). The route file follows DJI's custom Waypoint Markup Language (DJI WPML) (DJI Developer, 2025a).

As a new “.kmz” file corresponding to the replanned route segment is loaded in real time, the SDK application must be able to execute the flight automatically and transfer the photo from the UAV’s SD card to the UAV controller. To achieve this, two main modifications were implemented in the official Waypoint function of the DJI MSDK application using Android Studio:

1. Route file (“.kmz”) import and automatic mission start - Algorithm 2;

Algorithm 2 Start of the automated mission

```
Location to the directory containing the route file
Media Manager function initialization
while Detecting of a new route file in the directory do
    Uploading the route file (.kmz)
    Current route file deletion
    Starting the mission
end while
Mission successfully completed
```

2. Automatic download of photos acquired during flight - Algorithm 3. This function runs in the background so that, as each new photo is captured, it is transferred to the UAV controller.

Algorithm 3 Media Manager function

```
Enabling “Media playback” (DJI Developer, 2025b)
Initial update of the camera file list
Prevents historical downloads and keeps track of the current
photo count
while Mission not completed do
    Photo count update
    if “currentCount” > “lastMediaCount” then
        Filters only RGB images
        Identifies the most recent image by date
        Copy of the selected photo to the default directory
    end if
end while
```

For each photo captured by the UAV during flight and transferred to the UAV controller’s default directory, person detection along the route is performed (Step 1 - Section 2.2.1), the three-dimensional coordinates of the person closest to the UAV are determined (Step 2 - Section 2.2.2), the UAV path is replanned (Step 3 - Section 2.2.3), and a new route file is generated (Step 4 - Section 2.2.4). It feeds the SDK application, and route execution continues until the UAV reaches its destination. For this purpose, an Android application (“listener”) developed in Kotlin was implemented. It is responsible for automating communication between the UAV controller and a local server, which, in turn, handles real-time flight path replanning.

The Android listener application runs continuously on the UAV controller in the background (ForegroundService) and:

1. Monitors the UAV controller’s directory where photos are saved during the execution of the DJI MSDK application - Waypoint function - and detects when a new photo is received. The application uses the “FileObserver” class, which is capable of detecting changes in a system directory.

2. Uploads the detected photo to the FastAPI server using the Retrofit library. The transfer is performed via the HTTP (Hypertext Transfer Protocol) - the standard protocol for file transfer over the Internet.

3. Monitors the server’s local folder and, upon detecting a new available route file (“.kmz”), saves it in the specific directory of the UAV controller via the MediaStore API.

In this context, a local FastAPI server (Ramírez, 2025) was created, running on a computer connected to the same local Wi-Fi network as the UAV controller. Upon receiving the images sent by the listener application, this main server executes the algorithms required for path replanning. For the automatic and sequential execution of the algorithms (Step 1 to 4), the Papermill tool (Python, 2024) is triggered by the server. The final output of this process is the route file (“.kmz”), which is sent to the UAV controller via the listener application. Once the execution of a route segment is completed, the new route file saved on the UAV controller is loaded by the SDK application. The iteration between the DJI MSDK and the Android listener application is illustrated in Figure 1.

The developed system employs a distributed architecture: the human detection process in UAV images (Step 1) is decoupled from the main pipeline and executed on a dedicated server. This server is started before the main server and loads the YOLO model only once into memory. It remains active throughout the flight and is accessed via an HTTP API by the human detection algorithm. Consequently, it is not necessary to load the YOLO11m model for each new image captured by the UAV, which enables fast inference suitable for real-time applications.

3. Experiments and discussions

In this section, the results of the proposed UAV path replanning method are presented. All experiments were conducted on a computer equipped with an NVIDIA GeForce RTX 3060 GPU with 6 GB of memory. The experiments were conducted using a DJI Mavic 3M UAV exclusively in a simulated environment, employing the DJI Assistant 2 (Enterprise Series) flight simulator (DJI, 2025).

3.1 Algorithmic feasibility of the real-time UAV path replanning method

Table 2 presents the execution times of each step in the online UAV path replanning process, measured in a Jupyter Notebook environment (ideal case). Each process was executed 20 times to compute the mean and standard deviation of the processing time.

Process	Mean (s)	Standard deviation (s)
1 - Human detection	0.505	0.132
2 - Monoplotting	1.712	0.411
3 - Path replanning	0.046	0.009
4 - Route (geodetic coords)	0.445	0.072
5 - Route file creation	0.087	0.028
Total time (s)	2.795	

Table 2. Time required for executing the algorithms using Jupyter Notebook for real-time UAV path replanning.

The average total processing time for the real-time UAV path replanning is 2.795 s. Considering that the UAV flight is conducted at a low speed, the proposed method proves to be feasible. For example, assuming a flight altitude of 30 m and that a waypoint is defined every 15 m along the path (Figure 2), for a flight speed of 5 m/s, UAV path replanning is satisfactory. This is because, from the second waypoint of each segment to the

end - when a new replanned route segment begins - the UAV would travel 15 m at 5 m/s (3 s), which is longer than the average total processing time of the algorithms.

When analyzing each process individually, it is observed that the determination of the 3D coordinates of the person detected along the UAV path (Process 2) requires the most time and exhibits the highest standard deviation. Furthermore, in computational experiments, for certain positions of the detected person in the image, the execution time of the Monoplotting process was high, making real-time path planning unfeasible (average of 31.345 $\pm$ 10.413 s). This is because the Monoplotting photogrammetric procedure is iterative: depending on the initial position of the person in the image and the initial altitude used in the Monoplotting process, convergence may be slow.

Although these isolated cases (outliers) were not considered when computing the average execution time of Process 2 (Table 2), such exceptions would lead to unsafe system behavior and therefore characterize a system failure, as they would preclude safe real-time operation. Defining a maximum execution time for the process could enable real-time performance; however, it would not guarantee safety, since the person to be avoided would not be taken into account. These results thus indicate that estimating the 3D coordinates of individuals along the UAV path via Monoplotting may not represent the most promising solution. Alternative approaches, such as Simultaneous Localization and Mapping (SLAM) or Monocular Depth Estimation (MDE) with onboard processing supported by embedded GPUs on the UAV (Zhou et al., 2024), constitute a viable direction for future work.

The second process with the longest execution time is Human Detection (Process 1). The total execution time of 0.505 s includes data input, YOLO11m model prediction, and selection of the person closest to the UAV. The average time for YOLO11m model prediction - including image pre-processing, inference, and post-processing on a dedicated server - is only 191.405 $\pm$ 32.403 ms, demonstrating the capability of the YOLO model for real-time human detection in UAV images. Figure 3 illustrates the outputs of this process for a sample image.

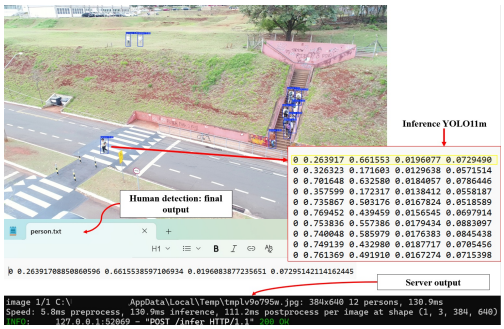

Figure 3. Results of the human detection process in UAV images.

Although the results confirm that the YOLO11m model is suitable for real-time human detection and that the evaluation metrics obtained from training on the Unicamp-UAV dataset (Table 1) outperform those reported for other models trained on different datasets - such as the study by Akshatha et al. (Akshatha et al., 2023) - further improvements in this processing stage are still required to ensure higher detection reliability. In particular, the recall value (R = 0.640) indicates a high rate of false negatives (i.e., undetected individuals), which may lead to severe or even fatal consequences in safety-critical applications. Consequently, human detection in UAV-acquired imagery remains an open research problem, with significant potential for improvement driven by the development of new deep learning models, such as YOLO26.

The process with the lowest average execution time is Path Replanning (Process 3), which demonstrates that the implementation logic is suitable for real-time replanning. Figure 4 presents qualitative results of this process. Notably, correct UAV path replanning can be observed in cases where the NFZ generated due to the detected person would result in a path very close to buildings in the flight area. Considering a minimum horizontal distance of 30 m between the person and the UAV’s vertical projection on the ground, for example, the minimum-risk path is traced maintaining a distance of at least 20 m from people. Figure 4 shows the initial NFZ around the person, with a radius of 31 m, considering the accuracy achieved through the Monoplotting process.

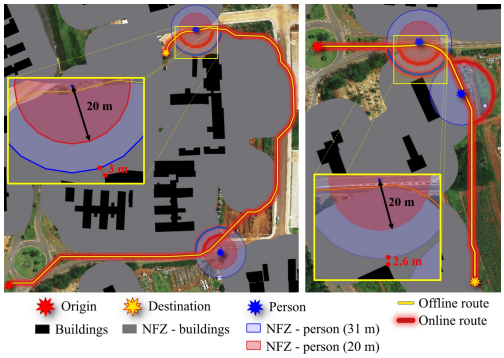

Figure 4. UAV path replanning considering static and dynamic obstacles.

Despite the promising qualitative results, the proposed method presents two main limitations. First, although it ensures avoidance of the person closest to the UAV along the planned path, this simplification may lead to unsafe trajectories in the presence of false negatives, i.e., undetected individuals, which limits its applicability in real-world UAV operations. Second, the accuracy of the replanned path strongly depends on precise estimation of the EOPs used in the monoplotting process (Simões et al., 2025). In the simulated experiments, EOPs were obtained via Phototriangulation, resulting in an average positional error of 1 m for the detected person along the UAV path. Conversely, EOPs derived directly from UAV image metadata may not provide comparable accuracy, potentially leading to replanned paths that violate minimum safety distances. Such inaccuracies pose significant risks in safety-critical UAV operations, particularly in dense urban environments.

The step described in Section 2.2.4 - “New route segment file creation” - is executed in two processes: (i) Route (geodetic coordinates), in which the waypoints of the new replanned route segment are defined; and (ii) Route file creation, in which the route file in DJI WPML format (“.kmz”) is generated. The latter process is the second fastest. Overall, the complete step (Step 4 - Section 2.2.4) takes an average of 0.532 s - the most computationally demanding step after Monoplotting. Despite its relatively high computational cost for defining a new route segment, the process remains feasible for real-time path replanning.

Figure 5 (a) shows the qualitative result of this step for one of the routes presented in Figure 4, without the presence of people along the path. The waypoint colors represent a single route segment, i.e., a “.kmz” file. Since the last waypoint of a previous segment corresponds to the first waypoint of the next segment, waypoint colors change every two points displayed. Visual analysis of the route files shown in Figure 5 (a) also indicates that, although the proposed method defines waypoints every “$h_{flight}/2$” meters, if the route contains curves and the waypoints are closer than this distance, their spacing is preserved. This ensures the UAV path avoids NFZs resulting from buildings and people.

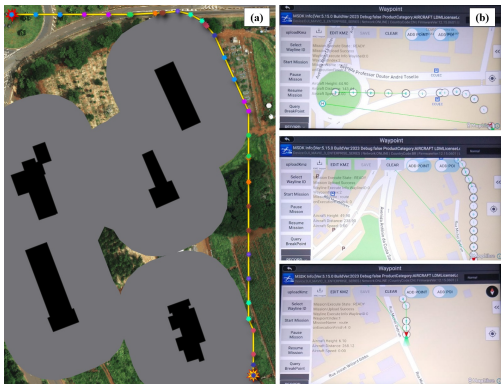

Figure 5. (a) Result of the route file creation stage in DJI WPML format, where each color corresponds to a new file. (b) Route execution using the DJI MSDK considering the same route files.

3.2 System feasibility of real-time UAV path replanning in a simulated environment

Although the average total processing time for the UAV path replanning is sufficient for real-time operation (Section 3.1), during UAV flight execution with the DJI Mavic 3M in a simulated environment, the average total time for path replanning is approximately 17.96 s (Table 3) - which is unfeasible for real-time operations.

Process	Mean (s)	Standard deviation (s)
1 - Human detection	3.198	1.267
2 - Monoplotting	2.217	0.426
3 - Path replanning	2.231	0.377
4 - Route (geodetic coords)	2.687	0.486
5 - Route file creation (“.kmz”)	2.098	0.328
Total server processing time (from image arrival to route file creation)	12.436 s $\pm$ 2.506	
Time between image capture and route file (.kmz) upload	17.960 s $\pm$ 2.380	
File transfer time ($server \leftrightarrow UAV controller$)	5.526 s $\pm$ 1.111	

Table 3. Online UAV path replanning time considering full system architecture and communication.

The total time between image capture by the DJI Mavic 3M UAV and the transfer of the route file to the UAV controller - measured from the notification of file reception by the listener application - includes both the total processing time of the algorithms on the server and the file transfer time. Comparing the total processing time of the algorithms executed directly via Jupyter Notebook (Table 2) with the execution on the server via Papermill (Table 3), the execution using Papermill is 4.45 times longer. This indicates that, although the UAV path replanning method proved satisfactory for real-time operation, the server configuration used is not appropriate.

Furthermore, there is a delay due to file transfer times - from the UAV controller to the server for the photo, and, after processing, from the server to the UAV controller for the route file (“.kmz”). This transfer time (average 5.526 s) is dependent on the Wi-Fi network being used. To identify and address this bottleneck, future tests should be conducted on different networks. Since photo transfer is the most time-consuming step, evaluating an image encoding method could be beneficial, as the route file transfer is nearly instantaneous.

Considering the execution time of each process on the server, the first process (Human Detection) is the most time-consuming. Although the use of a dedicated server for loading the YOLO11m model and performing inference on the UAV image reduces processing time, failures were observed on the dedicated server during the execution of a complete route. The dedicated server encountered an error that interrupted its operation, preventing the execution of subsequent processes. Therefore, the configuration of the main server needs to be reviewed in future work.

Despite this failure caused by the dedicated server for image inference, the UAV route execution in a simulated environment using DJI MSDK, DJI Assistant 2 (Enterprise Series) and listener applications was successful (Figure 5 - b). It can be observed that waypoint numbers “0”, “1”, and “2” correspond to the route segments traversed by the UAV, i.e., the route files executed. For each waypoint labeled “0”, a new “.kmz” file is executed - which corresponds to the waypoint colors shown in Figure 5 (a), defined according to the output of the route file creation algorithm (Section 2.2.4).

4. Conclusions

The qualitative results of the proposed UAV path replanning method confirm its feasibility for ensuring operational safety concerning both detected people along the trajectory and buildings within the flight area. As directions for future work, real-time estimation of population density is suggested, enabling the UAV to avoid more densely populated areas. Such an approach may further enhance public safety, even in the presence of false negatives, i.e., undetected individuals.

If the operation does not require high flight speed, the current implementation (algorithms executed in a Jupyter Notebook environment) is satisfactory for real-time operations. In contrast, when running the operation in a simulated environment, a challenge arises regarding the algorithm processing time on the local server. Although the use of Papermill is simple and easily reproducible, it presented limited performance when executing sequential algorithms. Modifications in the server implementation are required to guarantee real-time processing. As alternative solutions, it is suggested to: (i) convert the algorithms into “.py” files, eliminating the need to execute a Jupyter kernel as required by Papermill, which would increase execution speed; (ii) implement all algorithms in a single notebook and use an alternative Python library for execution; or (iii) refactor the notebooks into Python functions to be executed in a single script. Furthermore, the use of a compiled programming language, such as C++, represents an alternative implementation strategy for future research and may yield performance gains

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-XI-2-2026-741-2026 | © Author(s) 2026. CC BY 4.0 License.

of an order of magnitude or more. In future work, practical tests using the DJI MSDK and the listener application in simulated environments may be conducted to evaluate these alternative approaches to reduce processing time. Addressing this bottleneck would enable real-world outdoor flight experiments, which have not yet been conducted and currently represent a limitation of this research.

The dedicated server used for YOLO model loading and inference also exhibited failures, which may be related to reading the received photos - for instance, corrupted or invalid files. Additional tests are needed to identify the cause of these sporadic failures in order to mitigate them, either through a more robust server implementation or by adopting alternative methods for sending and handling images to the local server. In a fully optimized implementation without server bottlenecks, evaluating the YOLO26 model - the latest version released by Ultralytics - may yield promising results for human detection in UAV-acquired images.

Despite these challenges, one contribution of this study to future research is the validation of DJI MSDK use for route execution, including modifications to the original Kotlin code to adapt it to the proposed scenario, together with a listener application. This is particularly relevant because the MSDK application is open-access and freely available for developers working with DJI UAVs.

Moreover, DJI MSDK is supported by several UAV models beyond the DJI Mavic 3M adopted in this study. Consequently, the range of potential applications that can rely on this framework increases, and its route execution functionality has been successfully demonstrated in this work. As an example of an emerging application that may benefit from the DJI MSDK and its functionalities, UAV-based logistics operations stand out. The DJI FlyCart 30 UAV, for instance, specifically designed for such operations, is a platform supported by the Payload SDK (PSDK), and a mobile application developed with the MSDK (DJI MSDK) is required to control the payload device to perform specific actions and tasks (DJI Developer, 2025c).

Acknowledgments

The authors thank FAEPEX (Teaching, Research and Extension Support Fund) for supporting this research [grant 2498/24], the Graduate Program of Civil Engineering from University of Campinas for providing the infrastructure, and the Federal Institute of Education, Science and Technology of the South of Minas Gerais (IFSULDEMINAS) for its support to the first author.

References

Akshatha, K., Karunakar, A., Satish Shenoy, B., Phani Pavan, K., Dhareshwar, C. V., Johnson, D. G., 2023. Manipal-UAV person detection dataset: A step towards benchmarking dataset and algorithms for small object detection. *ISPRS Journal of Photogrammetry and Remote Sensing*, 195, 77–89. https://linkinghub.elsevier.com/retrieve/pii/S0924271622003008.

Aldao, E., Veiga-López, F., Chanel, C. P., Watanabe, Y., González-Jorge, H., 2025. Dynamic UAV trajectory optimisation for parcel delivery with integrated third-party risk mitigation. *Reliability Engineering & System Safety*, 262, 111178.

ANAC, 2023. Regulamento Brasileiro da Aviação Civil - RBAC-E nº 94 Emenda nº 03. Requisitos Gerais para aeronaves não tripuladas de uso civil.

Australian Government, 2023. Civil Aviation Safety Regulations 1998. Legislation. Statutory Rules No. 237.

DECEA, 2023. *ICA 100-40: Aeronaves Não Tripuladas e o Acesso ao Espaço Aéreo Brasileiro*. Ministério da Defesa - Comando da Aeronáutica, Brazil.

DJI, 2025. DJI Assistant 2 (Enterprise Series). https://www.dji.com/br/downloads/softwares/photo.

DJI Developer, 2025a. DJI WPML. https://developer.dji.com/doc/cloud-api-tutorial/en/api-reference/dji-wpml/overview.html.

DJI Developer, 2025b. Mobile SDK - Android MSDK v5.15.0 Version Release Notes. https://developer.dji.com/doc/mobile-sdk-tutorial/en/.

DJI Developer, 2025c. Payload SDK. https://developer.dji.com/doc/payload-sdk-tutorial/en/.

Douglas, D. H., Peucker, T. K., 1973. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: the international journal for geographic information and geovisualization*, 10(2), 112–122.

Fluehler, M., Niederoest, J., Akca, D., 2005. Development of an Educational Software System for the Digital Monoplotting. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 36, ETH Zurich, Institute of Geodesy and Photogrammetry. ISPRS Workshop Commissions VI/1 - VI/2: Tools and Techniques for E-Learning.

Gillies, S., 2025. The Shapely User Manual - Shapely 2.1.2 documentation. https://shapely.readthedocs.io/en/stable/manual.html.

Golabi, M., Ghambari, S., Ashayeri, S. A., Jourdan, L., Idoumghar, L., 2023. A Multi-objective 3D Offline UAV Path Planning Problem with Variable Flying Altitude. P. Legrand, A. Liefooghe, E. Keedwell, J. Lepagnot, L. Idoumghar, N. Monmarché, E. Lutton (eds), *Artificial Evolution*, Springer Nature Switzerland, 187–200.

Haque, K. M. S., Joshi, P., Subedi, N., 2025. Integrating UAV-based multispectral imaging with ground-truth soil nitrogen content for precision agriculture: A case study on paddy field yield estimation using machine learning and plant height monitoring. *Smart Agricultural Technology*, 12, 101542.

He, Q., Wang, R., Wang, Z., Wang, Y., 2023. A Dynamic Trajectory Planning Algorithm for Urban Ultra-Low Altitude UAVs Based on a Fusion Heuristic Algorithm. *2023 6th International Conference on Artificial Intelligence and Big Data (ICAIBD)*, 148–153.

Hu, D., Wang, X., Fu, Y., 2023. UAV Real-Time Path Planning Based on Heuristic Angle Search Strategy in an Unknown Environment. *2023 2nd International Symposium on Control Engineering and Robotics (ISCER)*, 216–222.

Jocher, G., Qiu, J., 2024. Ultralytics YOLO11. https://github.com/ultralytics/ultralytics.

Liu, Y., Akbar, A., Yu, T., Yu, Y., Kong, Y., Gao, J., Wang, H., Li, Y., Zhao, H., Liu, C., 2025. ARTEMIS: A real-time efficient ortho-mapping and thematic identification system for UAV-based rapid response. *ISPRS Journal of Photogrammetry and Remote Sensing*, 229, 396-421.

Luo, J., Tian, Y., Wang, Z., 2024. Research on Unmanned Aerial Vehicle Path Planning. *Drones*, 8(2).

Mikhail, E. M., Bethel, J. S., McGlone, J. C., 2001. *Introduction to Modern Photogrammetry*. John Wiley & Sons.

Park, S., Son, S. B., Jung, S., Kim, J., 2025. Dynamic Quantum Federated Learning for UAV-Based Autonomous Surveillance. *IEEE Transactions on Vehicular Technology*, 74(5), 8158-8170.

Peng, C.-C., Cheng-Yu, W., 2023. Design of constrained dynamic path planning algorithms in large-scale 3D point cloud maps for UAVs. *Journal of Computational Science*, 67, 101944.

Python, 2024. Papermill. https://github.com/nteract/papermill.

Ramer, U., 1972. An iterative procedure for the polygonal approximation of plane curves. *Computer graphics and image processing*, 1(3), 244–256.

Ramírez, S., 2025. FastAPI. https://fastapi.tiangolo.com/.

Rapidlasso, 2024. LAStools. https://rapidlasso.de/product-overview/.

Simões, D. P., de Oliveira, H. C., dos Santos, R. L., 2023. Analysis of Exterior Orientation Parameters on Monoplotting procedure for avoiding obstacles in UAV real-time routing. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, X-1/W1-2023, 423–429.

Simões, D. P., de Oliveira, H. C., dos Santos, R. L., 2025. Human detection with YOLO for last-mile delivery applications using UAVs. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*. Accepted for publication. Latin American GRSS and ISPRS Remote Sensing Conference (LAGIRS), 10-13 November 2025, Iguazu Falls, Brazil.

Simões, D. P., de Oliveira, H. C., Pereira, D. R., 2026a. Unicamp-UAV: An open dataset for human detection in UAV imagery. *ISPRS Journal of Photogrammetry and Remote Sensing*, 231, 119-136.

Simões, D. P., de Oliveira, H. C., Pereira, D. R., 2026b. Unicamp-UAV: An open dataset for human detection in UAV imagery. *ISPRS Journal of Photogrammetry and Remote Sensing*, 231, 119-136. https://www.sciencedirect.com/science/article/pii/S0924271625004149.

Simões, D. P., Oliveira, H. C. d., Barbosa, L. A., 2025. Real-time path planning for UAV with a focus on safety. https://proceedings.science/sbsr-2025/trabalhos/real-time-path-planning-for-uav-with-a-focus-on-safety?lang=pt-br. Proceedings of the XXI Brazilian Symposium on Remote Sensing. Publisher: Galoá.

Soorki, M. N., Aghajari, H., Ahmadinabi, S., Babadegani, H. B., Chaccour, C., Saad, W., 2025. Catch Me If You Can: Deep Meta-RL for Search-and-Rescue Using LoRa UAV Networks. *IEEE Transactions on Mobile Computing*, 24(2), 763-778.

Stöcker, C., Bennett, R., Nex, F., Gerke, M., Zevenbergen, J., 2017. Review of the Current State of UAV Regulations. *Remote Sensing*, 9(5).

Tian, Y., Ye, Q., Doermann, D., 2025. YOLOv12: Attention-Centric Real-Time Object Detectors. *arXiv preprint arXiv:2502.12524*.

Zhou, L., Wu, G., Zuo, Y., Chen, X., Hu, H., 2024. A Comprehensive Review of Vision-Based 3D Reconstruction Methods. *Sensors*, 24(7). https://www.mdpi.com/1424-8220/24/7/2314.