Virtual Reality (2024) 28:66
https://doi.org/10.1007/s10055-024-00965-6

ORIGINAL ARTICLE

Real-time safe validation of autonomous landing in populated areas: from virtual environments to Robot-In-The-Loop

Hector Tovanche-Picon[1] · Javier González-Trejo[2] · Ángel Flores-Abad[3] · Miguel Ángel García-Terán[1] · Diego Mercado-Ravell[2,4]

Received: 13 June 2023 / Accepted: 8 February 2024 / Published online: 5 March 2024

Abstract

Safe autonomous landing for Unmanned Aerial Vehicles (UAVs) in populated areas is a crucial aspect for successful integration of UAVs in populated environments. Nonetheless, validating autonomous landing in real scenarios is a challenging task with a high risk of injuring people. In this work, we propose a framework for safe real-time and thorough evaluation of vision-based autonomous landing in populated scenarios, using photo-realistic virtual environments and physics-based simulation. The proposed evaluation pipeline includes the use of Unreal graphics engine coupled with AirSim for realistic drone simulation to evaluate landing strategies. Then, Software-/Hardware-In-The-Loop can be used to test beforehand the performance of the algorithms. The final validation stage consists in a Robot-In-The-Loop evaluation strategy where a real drone must perform autonomous landing maneuvers in real-time, with an avatar drone in a virtual environment mimicking its behavior, while the detection algorithms run in the virtual environment (virtual reality to the robot). This method determines the safe landing areas based on computer vision and convolutional neural networks to avoid colliding with people in static and dynamic scenarios. To test the robustness of the algorithms in adversary conditions, different urban-like environments were implemented, including moving agents and different weather conditions. We also propose different metrics to quantify the performance of the landing strategies, establishing a baseline for comparison with future works on this challenging task, and analyze them through several randomized iterations. The proposed approach allowed us to safely validate the autonomous landing strategies, providing an evaluation pipeline, and a benchmark for comparison. An extensive evaluation showed a 99% success rate in static scenarios and 87% in dynamic cases, demonstrating that the use of autonomous landing algorithms considerably prevents accidents involving humans, facilitating the integration of drones in human-populated spaces, which may help to unleash the full potential of drones in urban environments. Besides, this type of development helps to increase the safety of drone operations, which would advance drone flight regulations and allow their use in closer proximity to humans.

Keywords Unmanned aerial vehicles · Visual-based landing · Virtual environments · Robot-In-The-Loop · Validation · Virtual reality

✉ Diego Mercado-Ravell
diego.mercado@cimat.mx

Hector Tovanche-Picon
hector.tovanche@uacj.mx

Javier González-Trejo
javier.gonzalez@cimat.mx

Ángel Flores-Abad
afloresabad@utep.edu

Miguel Ángel García-Terán
angel.garcia@uacj.mx

[1] Department of Industrial Engineering and Manufacturing, The Autonomous University of Ciudad Juarez, 32584 Ciudad Juárez, Chihuahua, Mexico

[2] Campus Zacatecas, Center for Research in Mathematics CIMAT AC, 98160 Zacatecas, Zacatecas, Mexico

[3] Aerospace and Mechanical Engineering Department, The University of Texas at El Paso, El Paso, TX 79968, USA

[4] Investigadores CONAHCYT, Mexico City, Mexico

1 Introduction

The role that Unmanned Aerial Vehicles (UAVs) have adopted in recent years to assist humans in diverse application fields such as precision agriculture, package delivery, and recreational activities, among others, has been greatly improved thanks to the high mobility and flexibility that this kind of devices provides jointly with the increasingly lower manufacturing cost. However, even with the enormous progress achieved with these vehicles, their true potential has been hindered in human-populated places, such as in urban scenarios, mainly due to the risk of injuring people in case of an accident, considerably limiting their application only to controlled environments and rural areas.

1.1 Regulations

Accordingly, these vehicles are subject to several safety constraints and legal regulations (Administration 2021). In the USA, these regulations encompass pilot certification and operational limitations. For instance, the vehicles must weigh less than 55 pounds, and they cannot fly above 400 feet above sea level, or on top of structures. Even more stringent are the limitations that prohibit drones from flying over people, which hinders their integration into various civilian applications for human assistance, Pinkam et al. (2019); Sanchez-Rodriguez et al. (2020); Heo et al. (2022). In the EU, similar regulations are in place, particularly concerning take-off and landing maneuvers. The legislation stipulates that the chosen sites must be suitable, with no risk of collision with the UAV (Bassi 2019; McTegg et al. 2022). In the UK, the Civil Aviation Authority allows flying drones or model aircraft weighing less than 250 g closer to people than 50 ms and over them, with the condition not to fly over crowds. However, they emphasize the inherent risks of flying near people (Authority 2023).

Even with these harsh constrains, it is still not guaranteed the safety of people in case of emergency landings caused by system failure such as signal loss, adverse environmental conditions, low battery, or human errors. In that sense, providing UAVs with emergency landing protocols, including autonomous landing, will help to prevent accidents, considerably boosting the drones' potential for assisting humans in civilian applications, especially in urban areas, close to people.

1.2 Challenges in autonomous landing algorithms

Although existing commercial drones may offer autonomous landing capabilities, they are normally limited to return to the take-off position (home) and descend slowly. In the best case scenarios, commercial drones may be capable of autonomous landing in a priori known visual tags.

Regarding research in the domain of autonomous landing algorithms, recent works have seen a proliferation of techniques, particularly focused on the detection of landing zones, obstacle avoidance, and semantic segmentation. Earlier approaches were often centered around the recognition of predefined landing markers, tailored to various scenarios. For instance, Garcia-Pardo et al. (2002) harnessed classic vision techniques to identify open, flat regions suitable for landing. In contrast, Nguyen et al. (2018) leveraged hardware such as UWB (Ultra-Wideband) for precision landing, emphasizing the importance of accurate positioning. Khazetdinov et al. (2021) adopted ArUco tags for precise landing location determination, providing a reliable visual marker-based approach. Wang and Wei (2023), on the other hand, focused on identifying landing markers on mobile platforms, introducing flexibility into the landing process. Johnson et al. (2005) introduced a novel approach to identify safe landing sites by efficiently applying structure-from-motion techniques to create detailed elevation maps of the landing area. This elevation map allows for the detection of hazards, enabling the algorithm to select a safe landing site based on the identified terrain features. In Chatzikalymnios and Moustakas (2022), the authors introduced an algorithm for landing site detection that evaluated multiple factors, including terrain flatness, inclination, and steepness. In another interesting work, Marcu et al. (2018) utilized synthetic data to train a system to estimate depth from in-flight images and segment them into "safe landing" and "obstacle" regions.

Another research area focuses on identifying safe landing areas through the generation of precise point clouds. For example, Yang et al. (2022) utilized point clouds for determining safe landing areas, while Ariante et al. (2021) employed LiDAR technology. Furthermore, Mittal et al. (2019) used synthetic data to simulate collapsed buildings, contributing to depth estimation for ensuring safe landings in emergency scenarios. It is important to note that while these methodologies have made significant advancements in the field of autonomous landing, they often rely on the availability of predefined landing markers or well-structured environments. Additionally, some techniques involve specialized and often expensive hardware, which can limit their applicability in complex unstructured scenarios, or are useless for widespread use in most commercial drones.

While these approaches are adequate in scenarios where predetermined landing locations or unobstructed landing zones are available, they may not be well-suited for highly dynamic, densely populated, unstructured environments. With our study, we venture into the intricate domain of autonomous safe landing in unstructured, densely populated,

and dynamic scenarios. The landscape in these environments is notably more complex, involving real-time obstacle detection, considerations of the human presence, and adapting to varying environmental conditions. This represents a complex open research problem, which still requires a lot of research effort to find robust and reliable solutions accounting for the different situations that may emerge in real scenarios, particularly in emergency landing situations.

The following research has contributed to the task of autonomous landing and their application in dynamic urban settings, by processing the UAV's sensory data with machine learning techniques, to tackle the complexity of the different scenarios. For instance, Nabavi et al. (2022) conducted research on automatic landing control of a UAV, and their work focused on developing advanced landing control algorithms using monocular camera data. Lee et al. (2021) delved into package delivery using autonomous drones using generative adversarial networks, and this study explored the challenges and possibilities of package delivery via autonomous drones within urban airspace. Their findings provided valuable insights into the intricacies of urban drone operations, including landing. In Alam and Oluoch (2021), a comprehensive review of taxonomy of landing scenarios is offered, providing a structured framework for understanding the various scenarios in which autonomous landing is applicable. This taxonomy serves as a reference point for categorizing landing challenges in urban settings. Furthermore, Kakaletsis et al. (2022) introduced a computer vision-based UAV flight safety pipeline, which laid the foundation for robust UAV safety in populated areas.

In addition to DNN (Deep Neural Networks)-based methods, some studies have explored the utilization of semantic segmentation for precise landing zone identification. Guerin et al. (2021) conducted research on semantic segmentation, developing techniques to classify image data at the pixel level, which allows for the clear distinction between suitable landing areas and those with potential obstacles or safety concerns. Kinahan and Smeaton (2021) also examined image segmentation to identify safe landing zones for unmanned aerial vehicles, where they employed semantic segmentation techniques to identify landing zones, addressing the need for accurate spatial understanding in dynamic urban environments. Many of these techniques rely on DNNs for their implementation. For example, Tzelepi and Tefas (2019) introduced a graph-embedded convolutional neural networks for crowd detection in a drone flight, leveraging DNNs to detect human crowds from drone imagery in dynamic urban settings. Castellano et al. (2020) contributed with crowd detection through fully convolutional neural networks, focusing on the application of DNNs to navigate around human crowds during drone landing procedures in populated areas, addressing safety concerns in densely populated regions. Mitroudas et al. (2023) introduced an Embedded lightweight approach for safe landing in populated areas, emphasizing the development of a lightweight algorithm for ensuring safe landings in densely populated regions, addressing real-time obstacle detection and collision avoidance in dynamic environments. Abdollahzadeh et al. (2022) explored the safe landing zones detection for UAVs, which utilized deep regression techniques to identify safe landing zones in densely populated and dynamic environments. Additionally, Liu et al. (2019) examined geometric and physical constraints for drone-based head plane crowd density estimation, integrating geometric and physical constraints into DNNs for crowd density estimation, improving our understanding of crowd dynamics during UAV operations in populated areas. However, it is important to note that while DNN-based methods show promise, they often assume the availability of predefined landing zones or limited environmental disturbances. In densely populated and dynamic urban scenarios, the need for real-time obstacle detection, collision avoidance, robust crowd tracking, and landing safety extends beyond the capabilities of existing methods.

As shown, the work in the literature presents many solutions that tackle the dynamism of an urban setting for the safe landing of the UAV in the population below it. However, most of the claims are made through the evaluation of metrics in datasets, without the field tests, and for that reason, it is imperative to have a real-time validation tool for all of those techniques to rigorously evaluate the presented theoretical results without potentially harming people in a real scenario.

1.3 Virtual reality applications in robotics

Before deployment of autonomous landing strategies in human-populated environments, it is absolutely necessary to validate them thoroughly and rigorously in order to guarantee their safe and correct behavior; hence, it is required to test them numerous times and quantify their performance and reliability, especially in challenging conditions as the ones encountered in real urban scenarios. For an algorithm that looks to avoid injuries and prevent accidents, it is a paradox to be tested in real conditions before a proper validation, since the system is likely to put in danger the same people that it is aiming to protect. It is also very important to widely test beforehand these vision-based algorithms to characterize its performance in the desired conditions, and compare different approaches to fine-tune them and select the best-suited ones; hence, in this work, we consider the use of virtual reality in robotics for safe validation of these autonomous landing algorithms.

Virtual reality applications in robotics have been explored in previous studies, revealing several advantages for testing autonomous algorithms in realistic environments. For

instance, the concept of a "digital twin" has gained prominence as it facilitates broader access to robotic platforms for educational purposes (Orsolits et al. 2022). Moreover, virtual reality extends its utility to the study of human–robot interaction, particularly emphasizing spatial computing and interaction (Delmerico et al. 2022).

Furthermore, the simulation of Unmanned Aerial Vehicles (UAVs) in realistic environments has been addressed in the literature. Notably, Guerra et al. (2019) developed FlightGoggles, a simulation framework replicating real-world scenarios and sensors, enabling Hardware-In-The-Loop validation. Their approach leverages Unity as a rendering engine and ROS as a robotic framework, providing photo-realistic environments using photogrammetry techniques. This framework was originally intended for racing drones; hence, it focuses on indoor scenarios useful for fast-moving drones equipped with a frontal camera However, it is important to note that while their work contributes significantly to the field of simulation, it does not specifically address the autonomous safe landing problem, and it confines algorithm testing to the ROS environment. We encourage interested readers to explore the FlightGoggles framework as another option for realistic simulations and creations on applications in indoor environments, primarily where the main sensor is a frontal camera. In contrast, our proposed validation framework may be useful for applications in complex outdoor urban environments, focusing on using a down-looking camera as the main sensor, as is the case for autonomous landing missions.

In this regard, the virtual environments provide a safe and inexpensive manner to test algorithms for autonomous vehicles (see Fig. 1). The virtual environments must offer close-to-real-world conditions to provide a useful approximation of the performance of the tested system. Also, the physics and dynamics of autonomous vehicles can be replicated to have close-to-real-world behavior. Nonetheless, despite the big efforts in generating realistic simulations, there will always be a domain gap between simulated and real-world scenarios to take into consideration (Sankaranarayanan et al. 2018). In this sense, the use of virtual environments for testing autonomous navigation cannot guarantee its complete success in real-world applications, but provides a powerful, safe, and cheap way, maybe the only available one, to test and compare different approaches as an intermediate validation step.

Fig. 1 Safe validation of UAVs autonomous landing in populated areas using virtual environments

1.4 Contributions

In previous work (Gonzalez-Trejo and Mercado-Ravell 2021), we have proposed a lightweight DNN architecture for crowds detection, suitable for real-time embedded applications, and provided a first solution to detect Safe Landing Zones (SLZ) in populated environments. Later on, in Gonzalez-Trejo et al. (2021), we considerably improved the SLZ detection, considering the camera movement, and implemented multiple-instance trackers of the SLZ along time. The performance of the strategy was only evaluated offline using two public image datasets with aerial views, Venice (Liu et al. 2019) and VisDrone (Zhu et al. 2020), as well as with data taken from a drone manually operated during landing missions in real populated scenarios. Nevertheless, due to the harsh safety constraints involved in the experiments, real-time validation during fully autonomous landing missions is still an important open challenge before deployment in real missions.

Henceforth, in this work, we present a new pipeline using virtual environments to evaluate visual-based autonomous landing algorithms (see Fig. 1), study their performance under diverse conditions, and compare different solutions. The proposed solution must select in real-time a target SLZ according to different criteria; then, a simulated UAV has to move to a suitable area and perform autonomous landing without putting people at risk. To do so, the drone and its physics are simulated using the AirSim plug-in. The virtual environment was rendered using Unreal Engine 4, a powerful graphics engine that offers photo-realistic characteristics, providing a close representation of urban scenarios with people. We tested the performance of the algorithm for autonomous landing in several randomized and dynamic environments, unknown to the UAV, with people present in the scene. We performed a wide range of tests under different conditions and scenarios, allowing some of the people to move with random walks, using diverse domain randomization such as density (people by square meter), people's initial position and appearance, floor texture, and weather conditions. The study suggests that the use of visual-based autonomous landing strategies in populated environments considerably helps to reduce the risk of accidents involving humans, allowing for a better integration of drones to assist humans. Furthermore, the validation using virtual environments allowed us to fine-tune and evaluate different

approaches, in order to find the best-suited solutions. In that regard, we evaluated different selection criteria to determine the "best" SLZ, and found that the use of the oldest SLZ tends to be more robust to the movement of the people in the scene, although the biggest SLZ seems to be safer in the sense of the distance to the closest person. Furthermore, the use of photo-realistic virtual environments and physics-based simulation proved to be a powerful tool, and an important validation step, to safely validate and evaluate the performance of autonomous landing strategies, guaranteeing the people's safety. Also, other evaluation stages have been included in the evaluation pipeline, including Software-In-The-Loop and Hardware-In-The Loop strategies, in order to test beforehand the behavior of the landing algorithms in the embedded autopilot. As a final evaluation stage, we propose the use of a Robot-In-The-Loop approach, where the physics gap introduced by the simulated drone is removed by using a real drone instead, performing the autonomous landing mission in real-time, in an open wide area, while the vision-based SLZ detection algorithms run in a realistic virtual environment, some sort of augmented reality to the robot. This strategy allows us to safely test in real-time the proposed strategies, in allegedly the closest-to-reality possible way, without jeopardizing any human being or material goods.

Our work focuses on the general and challenging problem of autonomous landing in real, complex, unstructured, densely populated urban environments, considering moving objects with unknown dynamics, and aims to offer a comparison benchmark and a validation pipeline using virtual reality tools. The goal is to prevent accidents and further increase the potential of drones for civilian applications, particularly in urban environments, and to motivate more people to contribute to find more robust, reliable, and resilient solutions for this complicated problem. The subsequent sections of this paper will delve into a detailed analysis and comparative study.

The main contributions are summarized as follows:

1. A baseline framework, using virtual reality tools, for safe and accurate comparison with other future works aiming to solve the problem of autonomous landing in populated environments.
2. A complete safe evaluation pipeline for visual-based autonomous landing algorithms is proposed, allowing to test landing algorithms without putting people at risk, while reducing the time and resources required for testing in real-world. This includes photo-realistic randomized virtual environments, physics-based simulated drones, Software-/Hardware-In-The-Loop, and Robot-In-The-Loop approaches.
3. Real-time implementation and evaluation of an autonomous landing strategy. This includes validation in a Robot-In-The-Loop approach, with a real drone autonomously landing in a virtual scenario.
4. Proposal of new evaluation metrics suitable for the autonomous landing task in human-populated environments.
5. Quantitative validation of the proposed pipeline through multiple iterations using domain randomization techniques in key aspects such as the people characteristics, people number and initial position, people's movements, texture and background of the scene, illumination changes, and weather conditions.

The rest of this work is organized as follows: In Sect. 2, we discuss the SLZ detection algorithm. In Sect. 3, we introduce the safe validation framework, including the baseline framework, the virtual environment, and the Software-In-The-Loop and Hardware-In-The-Loop implementations. Section 4 presents the main results obtained in the experiments, presenting the evaluation metrics and the quantitative validation. Finally, in Sect. 5, we provide some conclusions.

2 SLZ detection algorithm

The main objective of the SLZ detection algorithm is to propose zones free of people where the UAV can land without harming humans. The algorithm proposes SLZs in real-world coordinates during the mission, accounting for the camera and crowd movements (Gonzalez-Trejo et al. 2021). For that matter, the algorithm is divided into three sub-modules: (1) a lightweight DNN density map generator that detects the people in the image, (2) a custom algorithm to find suitable circular SLZs in real-world coordinates from the people-free regions, and (3) a multiple-instance object tracker by means of Kalman Filters (KF) that tracks the SLZs along frames, adding robustness for the sudden changes in the segmentation due to the camera and people's movement (note that in real-world, people move at will with unknown dynamics).

2.1 Crowd detection

To obtain suitable SLZ candidates, the first step of our algorithm is to infer the people's location from video streams in real-time. Most of the methods in the literature, and the one used by our algorithm, use density maps, which provide the number of people and their spatial location in an image, in the form of a heatmap. These density maps are trained to detect the people's heads in the image, given that the head is the most visible part of the people from aerial views, providing superior performance in scenarios where the crowd density is high, and the visibility of the people's bodies is reduced due to occlusions. In such scenarios, the landing of

the UAV in SLZ is crucial. In that regard, we developed a lightweight neural network called Bayes Loss Pruned Compact Convolutional Neural Network (BL Pruned CCNN) to generate these density maps at low computational cost, which is suitable for real-time embedded implementation in drones (Gonzalez-Trejo and Mercado-Ravell 2021).

The DNN is composed of nine fully convolutional layers divided by a head and a backbone, following the Multi-Column Convolutional Neural Network (MCNN) architecture described in Zhang et al. (2016). The head is constituted by three columns of one layer, each with different kernel sizes that capture features of small, medium, and big-sized heads in the image. At the end of the first layer, the backbone processes the features obtained by the network head to generate the density map; it is a heatmap encoding the inferred location and distribution of the people in the scene. The network was intentionally trained to overestimate the people in the scene, providing additional safety. We trained and fine-tuned the neural network using the Bayes Loss Ma et al. (2019), achieving a Mean Square Error (MSE) of 241.77 in the crowd-counting task. Finally, we reduced the original number of parameters from 0.075 M to 0.065 by pruning its channels and successfully implemented it in real-time on an embedded processor.

2.2 Safe landing zone proposals

Once our density map generator infers the crowd location, the next step is to find the SLZ in real-world coordinates. From the output of the density map D, we obtain a binary occupancy map O, with pixel values equal to either 0 or 255, by applying a threshold to the density map D. This occupancy map represents either crowded or free zones. For safety reasons, a dilation transformation is applied to the occupancy map O, to overestimate the people's location.

From there, using the camera pose from the proprioceptive sensors on the drone, we project O from the image plane to the world frame. Henceforth, we model our projection transformation using the thin-lens camera model, where we define three reference frames: $\mathbf{F_W}$ represents the inertial or world reference, the UAV body frame $\mathbf{F_B}$, located at its center of mass, and the camera frame $\mathbf{F_C}$. The homogeneous transformation to project coordinates from $\mathbf{F_B}$ to $\mathbf{F_W}$ is defined as $\mathbf{T}_B^W \in SE(3)$, and it is obtained from the onboard UAV sensors. The homogeneous transformation from $\mathbf{F_B}$ to $\mathbf{F_C}$ is defined as $\mathbf{T}_C^B \in SE(3)$, which is known a priori.

An important element of our pipeline that allows us to project from the image to the inertial frame is the assumption on which most of the heads are approximately located in a common plane known as the head's plane P in $\mathbf{F_W}$, which is parallel to the floor at the average people's height h_t. Nevertheless, we have observed good robustness to small variations in the head's height assumption, due to people seated or kids in the scene.

After our model is defined, to perform the projection of the occupancy map O to the head's plane P, denoted as O^W, we define a grid $G \in P$, as proposed in the sampler module of the Spatial Transformer Network (Jaderberg et al. 2015). Each element $G_{i,j}$ stores coordinates $\{x_W, y_W\}$ of the plane P, referred as $(G_{i,j}^x, G_{i,j}^y)$. For each $G_{i,j}$, we map a coordinate (x_I, y_I) in the image I as follows:

$$\lambda \cdot (x_I, y_I, 1)^T = \mathbf{K}\mathbf{T}_B^C\mathbf{T}_W^B \cdot (G_{i,j}^x, G_{i,j}^y, h_h, 1)^T. \tag{1}$$

where λ is a scale factor and $\mathbf{K}$ are the intrinsic parameters of the camera, defined in the Unreal Engine. Using Eq. 1, we query the value for each coordinate of the occupancy map O^W.

Once the occupancy map in the head's plane is established, we obtain a distance map C from where each nonzero pixel (people-free) contains the distance to the nearest zero pixel value (occupied zone). By calculating the maximum value of C ($\max\{C\}$), we effectively find the radius and center of the biggest circular SLZ. This SLZ is filled into the occupation map O^W, marking it as if it were occupied, and the process is repeated up to N_p times, in order to find the N_p larger zones free of people, which are considered as candidate SLZ.

2.3 Multiple-landing zone tracking

By themselves, the N_p candidate SLZ previously found can be used to flag the location in the head's plane where the UAV can land. Nevertheless, due to the constant movement of both the camera and the crowd through the scenario, the location of the SLZ may vary abruptly, potentially pointing the UAV to a landing zone with people on it. In that regard, we track the location of multiple SLZ using KFs, smoothing out the movement of the SLZ between each k frame of the video stream, ensuring their temporal consistency and adding robustness against people moving. For each SLZ i, let us consider a vector state at time k equal to

$$\mathbf{x}_k^{(i)} = \left[x_k^{(i)}, y_k^{(i)}, r_k^{(i)}, \dot{x}_k^{(i)}, \dot{y}_k^{(i)}, \dot{r}_k^{(i)}\right]^T, \tag{2}$$

where $(x_k^{(i)}, y_k^{(i)})$ are the coordinates of the center of the i-th SLZ in the plane P, $r_k^{(i)}$ the radius, and $\dot{x}_k^{(i)}, \dot{y}_k^{(i)}, \dot{r}_k^{(i)}$ the rates of change of the center coordinates and the radius, respectively. We model the displacement of the SLZ due to the movement of both the camera and the people using the constant velocity model, accounting for the acceleration in the process noise ω with normal distribution (Saho 2017); it is

$$\mathbf{x}_{k+1}^{(i)} = \begin{bmatrix} \mathbf{I}_3 & \Delta t\mathbf{I}_3 \\ \mathbf{0} & \mathbf{I}_3 \end{bmatrix} \mathbf{x}_k^{(i)} + \boldsymbol{\omega} \quad | \quad \boldsymbol{\omega} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}), \tag{3}$$

where $\mathbf{0}$ is a matrix of zeros, $\mathbf{I}_3 \in \mathbb{R}^{3\times3}$ is the identity matrix, and Δt is the time step. The process noise covariance, which models the acceleration, is defined as

$$\mathbf{Q} = \sigma_a \begin{bmatrix} \frac{\Delta t^4}{4}\mathbf{I}_3 & \frac{\Delta t^3}{2}\mathbf{I}_3 \\ \frac{\Delta t^3}{2}\mathbf{I}_3 & \Delta t^2\mathbf{I}_3 \end{bmatrix}, \tag{4}$$

where σ_a represents the acceleration uncertainty, obtained empirically. We use the SLZ measurements as they are inferred by the previous step. More formally, the measurement model is defined as

$$\mathbf{z}_k^{(i)} = \begin{bmatrix} \mathbf{I}_3 & \mathbf{0} \end{bmatrix} \mathbf{x}_k^{(i)} + \boldsymbol{\nu} \quad | \quad \boldsymbol{\nu} \sim \mathcal{N}(\mathbf{0}, \mathbf{R}), \tag{5}$$

where $\boldsymbol{\nu}$ is the measurement noise with covariance matrix $\mathbf{R}$.

The algorithm maintains a M_p number of KF trackers where $M_p \leq N_p$. We associate a KF to a SLZ candidate through the Intersection over Union (IoU) between the area of the detected SLZ A_i and the SLZ tracked by the KF A_j; it is

$$IoU = \frac{A_i \cap A_j}{A_i \cup A_j}. \tag{6}$$

Having calculated the IoU for all pairs of candidates and KFs, we associate a candidate with the KF that overcomes a threshold μ, using the Hungarian algorithm (Bruff 2005). Each time an existing KF tracker fails to match with a suitable SLZ candidate, a counter is increased. If the counter exceeds a threshold μ_1, the tracker is eliminated. Finally, since for some frames the SLZ candidates cannot be calculated, we can use the prediction step of the KF until the next batch of SLZ is inferred.

3 Safe validation framework

3.1 Virtual environment

For the virtual environment and the simulated drone, it was employed a PC with an I7 6700 processor, 32 GBs RAM memory, and due to the heavy computational burden of running both the virtual environment and the autonomous landing algorithm in real-time, a RTX 2080 graphics card from NVIDIA was also used.

As presented in Shah et al. (2018), the AirSim plugin provides a high-fidelity simulation for UAVs, with full integration between a flight controller, a physics engine, inertial sensors, video cameras, and depth sensors, among other capabilities. AirSim can be integrated with the Unreal Engine and Unity, two of the most used engines for 3D recreation. In this work, the selected engine is Unreal Engine 4 due to its photo-realistic rendering capabilities.

In order to recreate realistic urban scenarios, the open-source package City Park Environment collection Lite was used (SilverTm 2021); it contains high quality assets ideal for recreating city parks. The scenarios present in this environment are suitable to test our algorithms, providing a realistic quality with rich textures, in a wide variety of challenging urban-like environments. Figure 2 presents some samples of the selected scenarios for this work, including a soccer field, a parking lot, basketball courts, and a field with trees.

To represent people crowds in the virtual environment, the *Twinmotion*-posed Human pack was selected, and this pack contains 142 different 3D-scanned human characters with high-resolution textures, considering diverse characteristics such as age, gender, racial diversity, and a wide range of clothes, head accessories, and poses. These “actors" (in Unreal 4 context) could be identified as humans by the SLZ detection

Fig. 2 Sample scenarios from the City Park Environment SilverTm (2021), containing a soccer field, a parking lot, basketball courts, and a field with trees

algorithm, allowing the cross-domain application of a model trained with real-world data and deployed in a virtual environment, allowing us to test the visual-based autonomous landing algorithms without putting people at any risk.

3.2 Domain randomization

The main purpose of domain randomization, during the validation stage of an algorithm, is to provide enough simulated variability to represent real-world conditions. We randomized from uniform distributions various important aspects at each trial, for instance, the test scenario (parks, sports courts, parking lots, streets, etc.); people characteristics (gender, racial group, hairstyle, clothes, accessories, etc.); number of people, initial position and pose; actors movement; texture of the floor; scene illumination; weather condition; and drone's initial pose.

3.2.1 Actors placement

To create several scenarios from a larger scene in the City Park Environment, first a Region Of Interest (ROI) is randomly selected for each autonomous landing mission in the form of a square of $30m^2$. Then, the number of actors $n_a \in [80, 120]$ within the ROI is randomly generated, as well as the initial position of the actors in the ROI, avoiding overlap between actors, all of these using uniform distributions. Also, the kind of actor and their characteristics are randomly generated. The initial position of the drone in the ROI is also randomized.

3.2.2 Actors movement

After the initial position of every actor is set, an algorithm randomly selects some of the actors to move during the experiment, in order to emulate the dynamic nature of scenes containing people, where the people are constantly moving at will, with dynamics unknown to the vision algorithm. In this work, we introduce a two-dimensional random walk algorithm implemented on a lattice. The lattice, represented as $\mathbb{Z}^d$, is the discrete space in which the actor moves. Let $(x_k^a, y_k^a) \in \mathbb{Z}^d$ denote the actor's position in the lattice at time step k. The random walk algorithm updates the actor's position in the lattice by adjusting its horizontal and vertical coordinates.

Specifically, the update process is described by Eq. (7), where x_k^a represents the actor's horizontal position at time k. x_{k-1}^a is the actor's previous horizontal position. α_x represents the increment in the horizontal direction, and it can take one of the values from the set $\{-\lambda, 0, \lambda\}$. This value is randomly generated at each time step. Similarly, y_k^a and y_{k-1}^a represent the actor's vertical positions. α_y represents the increment in the vertical direction, with the same set of possible values as α_x. λ is a constant scalar that ensures that the actor's movement remains bounded within the lattice. The actor's position is updated 10 times per second according to this random walk algorithm. This equation governs the actor's movements on the lattice, allowing for exploration of its spatial behavior.

$$\begin{aligned} x_k^a &= x_{k-1}^a + \alpha_x; \\ y_k^a &= y_{k-1}^a + \alpha_y \end{aligned} \tag{7}$$

3.3 Autonomous landing implementation

As previously stated, AirSim is used to simulate the physics of a virtual UAV, and it provides useful information of the emulated most common sensors in a drone, such as an Inertial Measurement Unit (IMU), altimeter, and different cameras. This way, integration with the virtual drone in AirSim is analogous to integration with a real drone and allows us to test different movement policies and landing strategies. Figure 3 represents the algorithm used for the autonomous landing task. In the first stage, the drone's pose from the simulated IMU and a frame from a virtual down-looking camera attached to the drone are sent to the SLZ algorithm described in Sect. 2. Then, a set of candidate landing areas is provided in real-time by the SLZ vision algorithm, parameterized by the center of the circular SLZ in real-world $\mathbf{F_W}$ coordinates, the radius and an object identifier (ID) for each SLZ. The detected SLZ is updated at each iteration, and being this a mobile platform in a possibly dynamic environment (people moving), the available landing regions may be changing dynamically, resulting in one of the main challenges for this problem. Hence, it is necessary to define adequate criteria to select a landing zone. Indeed, one of the advantages of the proposed virtual framework is the ability to study different criteria for selecting "the best" SLZ, for instance, considering the size (biggest SLZ), the distance to the drone, or the time consistency (oldest SLZ). After selecting a target landing zone, a 3D coordinate in $\mathbf{F_W}$ is parsed as a waypoint command to the drone, 2m above the center of the selected SLZ in the head's plane P. If the altitude of the drone is less than or equal to 2m and a valid SLZ is underneath the UAV, a landing command is sent to the drone and the mission is over, otherwise, the drone must move toward the target SLZ and iterate the algorithm updating the image frame and the UAV pose.

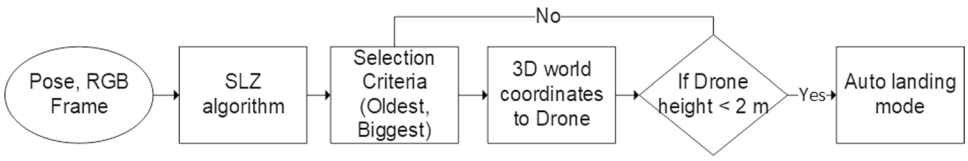

Fig. 3 Flowchart of the autonomous landing algorithm based in the visual-based SLZ detection algorithm

Algorithm 1 Algorithm of the validation process for the autonomous landing algorithm using virtual environments

```
Test_No ← input
Require: N = 0
while N ≤ Test_no do
    start_environment()                                  ▷ Set Environmet
    set_initial_poses()
    set_weather()
    go_to_initial_position()
    Start_SLZ_algorithm()                  ▷ Get simulated POSE and frame
    Pose ← Get_UAV_POSE()
    Frame ← Get_frame()
    if Altitude ≥ 2m and Criteria = Oldest then
        x, y ← SLZ[0]                                    ▷ Get oldest landing zone
    else if Altitude ≥ 2m and Criteria = Biggest then

        for SLZ do
            for Radius do
                if Radius(J) ≥ Radius(J + 1) then
                    Biggest ← Radius(J)
                    Radius(J) ← Radius(J + 1)
                    Radius(J + 1) ← Biggest
                end if
            end for
        end for                                          ▷ Sort biggest radius
        x, y ← Biggest
        Send_target_coordinates()
        if Altitude ≤ 2m then
            auto_landing()
            break()                                      ▷ Landing if below 2m
            N ← N + 1
        end if
    end if
end while
```

3.4 Software-/Hardware-In-The-Loop

To enhance the evaluation of the proposed algorithm, a Software-In-The-Loop (SITL) approach was employed. During this stage, the algorithm was integrated with a simulated drone using the PX4 stack as the flight controller firmware. The simulation environment, AirSim, facilitated the replacement of its built-in controller with SITL. The communication flow between the algorithm and the simulated drone is illustrated in Fig. 4.

In the proposed SITL validation, the PX4 SITL stack is connected to MAVProxy, creating multiple connection points. One of these points is connected to the proposed safe landing algorithm through the MAVSDK library. The connection to AirSim is achieved through some parameters modification; then, the video feed from the simulated environment in Airsim is used by the algorithm to compute the target landing zone and send it as a waypoint to the SITL, which is continuously updated in the AirSim simulation. Finally, QGroundControl connects to MAVProxy for monitoring the drone's state.

On the other hand, Hardware-In-The-Loop evaluation is also enabled, in order to test the autonomous landing algorithms implemented in the embedded autopilot to be used in the real drone. To achieve our objective, we utilized an approach that is akin to the flowchart proposed in the Software-In-The-Loop (SITL) stage. However, we made modifications to the SITL PX4 block by substituting it with our own autopilot, which is the NXP FMUK66FMU running the PX4 firmware. This substitution enabled us to connect

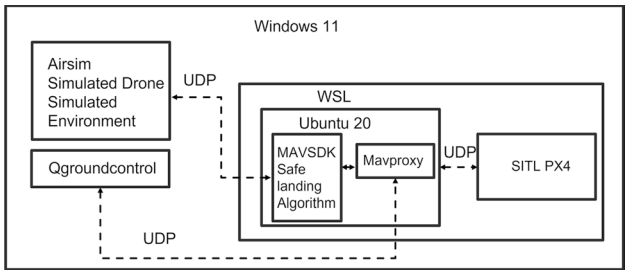

Fig. 4 Software-In-The-Loop flowchart

to the real hardware, thereby adding another layer of safety, and to test the control output of our SZL algorithm within the context of a simulated environment.

3.5 Robot-InThe-Loop

The "Robot-In-The-Loop" approach is a testing methodology that involves the integration of a physical robot and a simulation environment in real-time. In this scenario, a real drone connects to a virtual avatar drone in the simulated environment and receives input from the SLZ algorithm, which is fed by the simulated video streaming from the virtual avatar drone. This setup allows for the evaluation of the performance of the real drone in a controlled, virtual environment. In the following sub-sections, we will detail the platform specifications and our implementation.

3.5.1 Experimental platform

Regarding the experimental drone, a high-strength carbon fiber frame was utilized with a diagonal measurement of 500 mm and a central plate measuring 150 mm by 150 mm to mount electronic components securely. To manage flight operations, the NXP FMUK66FMU autopilot was selected due to its advanced features, including a barometric pressure sensor, 3-axis accelerometer, 3-axis magnetometer, and multiple communication connectors such as UART, PPM, CAN, SPI, and GPIO for precise motor control. To meet lifting power requirements, four 920-KV brushless DC (BLDC) motors were integrated with 40A OPTO ESC motor speed drivers, and a 4 S 5000-mAh LiPo battery was selected to ensure long-lasting flight capabilities. Figure 5 illustrates the experimental platform used in this work.

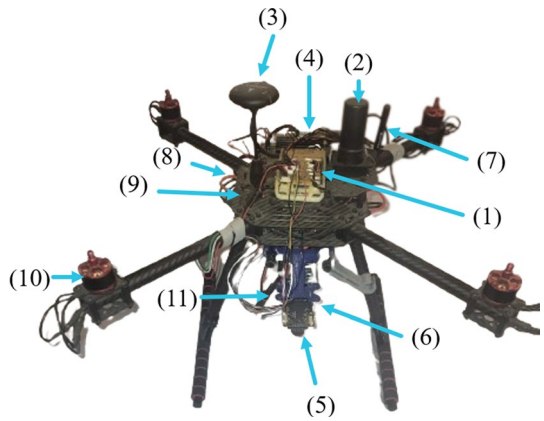

Fig. 5 Experimental platform components: 1.- Flight controller. 2.- RTK. 3.- GPS. 4.- Companion computer (Jetson Nano + Wi-Fi). 5.- Optical flow. 6.- TfMini LiDAR. 7.- RF telemetry. 8.- ESC controller. 9.- Carbon fiber frame. 10.- Brushless motors. 11.- LiPo battery

3.5.2 External components

To achieve precise positioning of the agent, a combination of sensors was employed. The primary source of the drone's position was obtained using a pair of Holybro's H-RTK F9P units, one mounted on the drone and the other on the ground station. A M8N GPS was utilized as a secondary source of data, and an optical flow camera was also employed. The height of the agent is critical for the algorithm, and the height data were redundantly acquired using a single-point LiDAR. The data from all of these sources were seamlessly integrated using an extended Kalman Filter, which was implemented in the PX4 firmware.

As a companion-embedded computer, a Jetson Nano was utilized, which was connected to the flight controller through a serial interface. The MAVProxy library was employed to transmit the pose information of the drone to the ground station in real-time. This setup enabled efficient communication and accurate positioning of the drone during flight operations.

3.5.3 Proposed Robot-In-The-Loop communication flow

The validation of the safe landing algorithm using real hardware provides a more realistic evaluation of the results. As a preliminary step before testing the algorithm with real people, we propose the Robot-In-The-Loop validation. In this approach, the real drone moves within its environment according to the safe landing algorithm, but instead of using the real camera of the drone, the video feedback from the virtual avatar drone in the simulator is utilized. This avatar drone mimics the movements of the real drone, but in a virtual environment full of people, where the vision-based algorithms provide the safe landing zone from the video stream of the virtual avatar. This way, autonomous landing is accomplished in a real drone, but without jeopardizing any human being.

Figure 6 illustrates the communication between the drone and the ground station through radio frequency telemetry and Wi-Fi. On the ground station, a Ubuntu 20 distribution running as a Windows subsystem Linux allows for communication between the MAVProxy and a Python library called

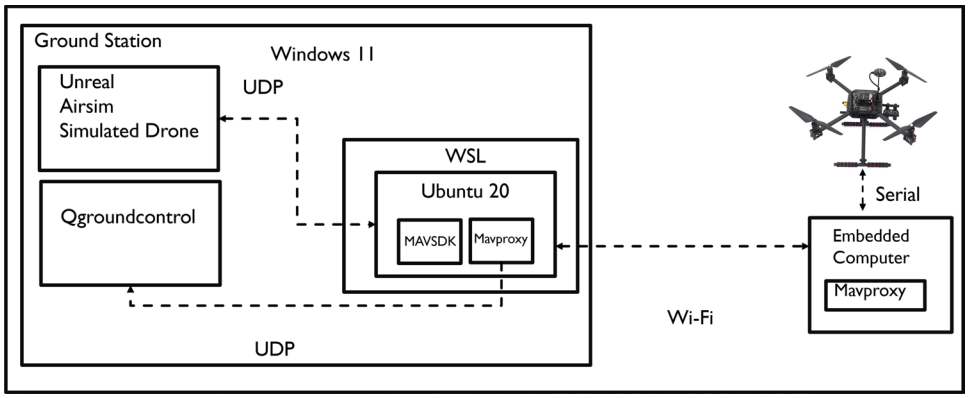

Fig. 6 Robot-In-The-Loop diagram. A real drone performs autonomous landing based on the SLZ detected by an avatar drone in a virtual environments, which mimics the movements of the real drone. The virtual environment, the avatar drone, and the vision-based SLZ detection algorithms run in a ground station computer, which send the selected LSZ to the embedded computer on the real drone via Wi-Fi

MAVSDK. On the outer layer, within the Windows operating system, the simulated drone mimics the pose of the real drone and provides video feedback in the AirSim environment. For safety and visualization purposes, QGroundControl is utilized.

This approach offers a safe test-bed for the validation of the algorithm under real flight conditions, ensuring that the results are as close to reality as possible.

4 Experimental results

To validate the autonomous landing algorithm for UAVs in populated areas, several tests have been carried out in the virtual environment, evaluating the algorithm under a wide variety of scenarios with different conditions. For each scenario condition, 100 iterations of autonomous landing missions have been performed. We consider four different scenario conditions: static actors, 10% of the actors moving, 20% of the actors moving, and dynamic weather conditions, including dust, rain, snow, and fog. At each landing iteration, random initial conditions are generated as described in the previous section. Furthermore, two different criteria have been studied for the SLZ selection, considering the biggest SLZ available and the oldest SLZ. Bigger landing zones provide increased safety and are robust against movements in the boundary of the SLZ; meanwhile, older SLZ tends to be the more consistent areas over time, aiming for larger robustness against people movements. Algorithm 1 describes the steps of each autonomous landing iteration; first, the random environment is initialized, setting initial positions for each actor and the weather conditions; then, the UAV takes off and moves toward a random initial position; afterward, the autonomous landing algorithm is started, and a target landing zone is selected at each time step based on the selected criteria, checking for collisions with the human actors, note that the detection algorithm may fail to detect some of the people, or the people movement may cause some actors to trespass the selected SLZ. Finally, different metrics are obtained, as described in the following Sub-section. At the end of the 100 runs, some statistics are obtained and a quantitative comparison is possible. A video demonstrating the autonomous landing evaluation is available at https://youtu.be/AwQzNdVE0ZU

4.1 Evaluation metrics

In order to measure the performance of the landing algorithm in real-time, we propose a set of evaluation metrics. The *"successful landing rate"* is defined as the number of

Table 1 Performance comparison for autonomous landing under different conditions

Scenario conditions / Strategy	Success (%)	Warning 1 m/1.5 m (%)	SLZ area (m^2)	Avg. Nearest (m)	IoU
Static / Random landing	59	2.7 / 10	*	1.15	*
Static / Oldest SLZ	97	0.0 / 0.4	6.15	2.21	0.48
Static / Biggest SLZ	99	0.0 / 0.3	9.62	2.45	0.59
Moving actors 10% / Oldest SLZ	87	0.06 / 0.16	4.59	1.97	0.39
Moving actors 10% / Biggest SLZ	80	0.01 / 0.1	9.07	1.70	0.60
Moving actors 20% / Oldest SLZ	84	0.06 / 0.14	4.37	1.58	0.42
Moving actors 20% / Biggest SLZ	76	0.09 / 0.2	8.04	1.51	0.54
Dynamic weather / Oldest SLZ	86	0.08 / 0.16	4.49	1.84	0.52
Dynamic weather / Biggest SLZ	80	0.06 / 0.1	6.95	2.07	0.54

missions in which the virtual UAV was able to land without colliding with any human actor. This is the most important metric in our study, providing a direct measure of the number of accidents prevented by the algorithm.

However, just considering the rate of successful landings does not provide a complete picture of the performance of the SLZ algorithm and the selection criteria; hence, other metrics are proposed. The *"Warning score"* represents the percentage of times some actor ended within a safety radius, either 1m or 1.5m, around the landing point, and provides an idea of the risk of accident, that is, how many times certain approach ended up close to produce an accident. Moreover, the *"SLZ Area"* is an important metric on its own, given that larger landing zones offer increased safety to the people around, in case of errors, and allows aware people to move apart to avoid collision. The *"Nearest person"* measures the average Euclidean distance between the drone and the closest person during landing, where larger values are preferred. Finally, the *"IoU"* provides a measure of the accuracy of the proposed algorithm to detect the real available SLZ; it is computed for the target landing zone against the closest ground truth people-free region.

4.2 Results

Table 1 presents the obtained results over 100 iterations for each case of study. To establish a comparison baseline, 100 uniform random landings were also included in this experiment, considering only static actors, emulating the scenario where a drone must perform a blind emergency landing. Given that not all the ROI was occupied by people, there is a possibility that some of the landings were successful. In this case, the random landings reached a 59% of success. Then, we evaluated two different criteria of interest to select a landing area, the biggest SLZ and the oldest SLZ, first for a static environment. The biggest SLZ is expected to add safety to the landing mission providing extra space, it also should increase the robustness with respect to people miss-detections, and give more margin for aware people to step aside. On the other hand, the oldest SLZ is expected to be more consistent over time, adding robustness against moving people; effectively, in real scenarios, the oldest SLZ would tend to stay in regions where people do not circulate often. As expected, for the case of a static scenario, both criteria performed fairly well, reaching 97% and 99%, respectively, which results in an improvement of almost 40% when compared with the blind random landings, demonstrating that the use of autonomous emergency landing can considerably reduce the risk of accidents involving people.

Afterward, the performance of the proposed strategy was tested in an environment with moving actors, which is one of the main challenges of autonomous landing in human-populated environments, where people move with unknown dynamics. Due to computational limitations, we only tested the strategies for movement in the 10% and 20% of the actors in the scene. Nevertheless, this is enough to analyze the behavior of the landing algorithm under dynamic conditions and evaluate the robustness of different strategies against people movement. Adding mobile actors in the scene considerably increases the difficulty of the task, resulting in a decrease in the performance of the strategies. Furthermore, the proposed pipeline was tested under dynamic weather conditions, such as dust; mist; rain; snow; and illumination changes, casting similar results, proving that the algorithm generalizes well to adverse scenarios and is robust to noisy conditions.

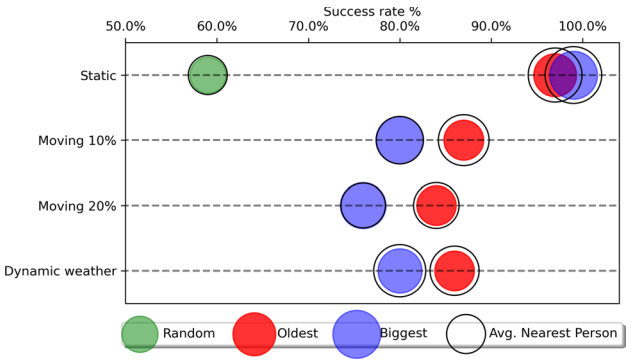

Fig. 7 Performance analysis by success rate, in static, dynamic actors, and dynamic weather conditions. The color-filled circles denote the average size of the SLZ proposed by the detection algorithm, while the empty circles represent the average distance to the nearest person at landing. Green circles stand for the uniformly random strategy, red denotes the oldest SLZ strategy, and blue represents the biggest SLZ strategy. The radius of the green circle represents $1m$ and provides the scale for the other circles. The oldest SLZ shows better performance in terms of success rate, while the biggest SLZ provides a larger clearance

Figure 7 depicts the success rate between the random strategy and the SLZ algorithms in the four considered cases of study: it is static, 10% moving actors, 20% moving actors, and dynamic weather. Furthermore, the size of the color-filled circle represents the average size of the SLZ proposed by each algorithm, along the 100 iterations, while the empty circle shows the average distance to the nearest person. Red circles stand for the oldest SLZ, blue for the biggest, and green for the random bling landing with uniform distribution. Contrary to the autonomous landing solutions, the size of the green-filled circle corresponding to the random landing is of unitary radius ($1m$) and provides the scale for comparison. Bigger circles (filled and empty) and further to the right solutions are to be preferred, since they correspond to solutions with larger landing areas and with higher success rate, respectively.

We can appreciate that, although there is still room for improvement, both autonomous landing strategies, oldest

and biggest, obtained good results even in harsher circumstances, when compared with the random landing in the easiest static scenario. More in particular, the obtained results suggest that the use of the oldest SLZ is better suited to account for people moving with unknown dynamics, given that the oldest SLZ tends to be in regions where people are less likely to be found. Another important metric is the SLZ area, represented as colored circles, considering that bigger areas are to be preferred, the oldest SLZ strategy tended to decrease around 30% compared with the area obtained by the biggest SLZ strategy, in each of the scenarios and strategy combinations, suggesting that there is a trade-off between being robust against people moving, and providing additional safety by choosing a larger SLZ. Similarly, we measure the average distance over iterations, from the landing spot to the nearest person in the scene, as presented in Fig. 7, denoted in empty circles, where larger circles indicate a larger clearance distance. As expected, the biggest SLZ outperformed the oldest also in this criteria.

It can also be noted that the average distance between the landing spot and the nearest person, in all of the proposed scenarios, provides a larger distance than the radius of the proposed landing zone, presenting bigger black circles than the colored circles for both strategies. This is expected since the SLZ detection algorithms were intentionally designed to overestimate the people detection and their location (false positives are preferred over false negatives), as an additional safety measure, due to the delicate nature of the task. Also, in all the cases, the use of autonomous landing algorithms increased the average distance to the nearest person, even in harsher scenarios, when compared to the random landings in the easiest static scenario. On the other hand, in Table 1, we also evaluate the similarity of the found SLZ against the closest available ground truth landing zone, which provides a measurement on the capacity of the algorithm to accurately find a good SLZ. In this sense, the biggest SLZ strategy outperforms the oldest one.

Fig. 8 Robot-In-The-Loop test. A real robot performs autonomous landing without putting any human at risk, based on the safe landing zones detected in real-time by a vision algorithm running in an avatar drone in a virtual environment. The bottom left figure shows the virtual avatar, while the bottom right depicts a snapshot from the down-looking camera on the real drone

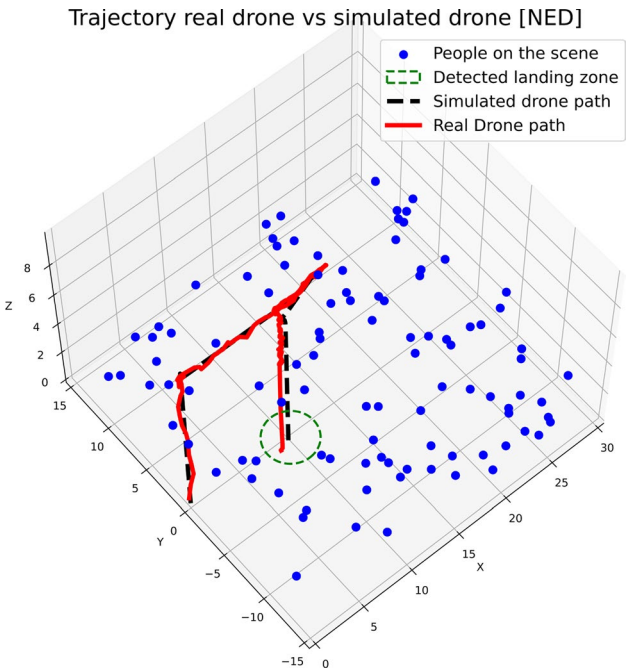

Fig. 9 Experimental autonomous landing mission using the Robot-In-The-Loop validation approach. The blue dots mark the virtual people on the scene, the green dashed circle depicts the selected landing zone, while the solid red line and the dashed black line represent, respectively, the paths followed by the real drone and its virtual avatar

4.2.1 Robot-In-The-Loop results

In the context of the real drone experiment, preliminary results have been obtained. The real drone successfully performed an autonomous mission; Fig. 8 shows the real drone performing a mission, taking as input the command results from the SLZ algorithm and using video feed from the simulated AirSim environment. As depicted in Fig. 9, the path followed by the drone (represented in the NED frame) is shown in the context of virtual people on the scene. It can be observed that the selected landing zone, indicated by a dashed green circle, is in a clear area, leading to a successful landing.

5 Conclusions and future work

Autonomous landing of UAVs in populated environments is a new and exciting research area, with a lot of challenges and great potential for integrating drones with humans in real-world applications. In fact, providing UAVs with these capabilities is a key aspect to explode the full potential of drones

in assisting humans in urban environments, particularly for preventing accidents in emergency landing situations. However, due to the risk involved, it is of crucial importance to fail-proof such autonomous landing algorithms beforehand, in order to fine-tune the algorithms and select the best-suited ones, and more importantly, to guarantee the safety of the people around, which turns out to be a very difficult task.

In this work, we have proposed an evaluation framework using virtual environments for safely and thoroughly testing the autonomous landing algorithms in populated areas. The virtual environments were generated with the Unreal graphics engine, whereas a virtual drone is simulated using AirSim. This framework allows us to iterate the autonomous landing experiment more than 900 times, under different randomized conditions, reducing the time and resources needed for evaluation, tuning, and testing. Moreover, the evaluation pipeline further includes Software-/Hardware-In-The-Loop validation, along with Robot-In-The-Loop tests. This allows us to test the proposed landing strategies at different stages and assure their good performance without putting any human or material goods at risk.

Also, two variations of a vision-based safe landing zones detector were evaluated in real-time, selecting either the biggest available landing zone or the oldest available one. Different metrics were computed, and a quantitative comparison study was provided. The study suggests that the use of autonomous landing algorithms in populated environments considerably reduces the risk of injuring people. Furthermore, it was found that the oldest safe landing zone tends to be more robust in the case of a dynamic environment where people is moving in the scene, obtaining up to 86% of success rate in the most difficult case of study, with a 20% of the actors moving. Nevertheless, the biggest SLZ offers an additional degree of safety for the landing, given that it is more robust to false positive people detections, and provides more space for aware people to react and move apart, hence becoming a good alternative approach. In conclusion, the best solution would probably be a combination of both criteria in the form of a multi-objective cost function.

The use of virtual environments coupled either with virtual or with real drones (Robot-In-The-Loop) has proven to be a powerful tool for safe validation and quantitative evaluation of autonomous landing strategies in human-populated environments, and an absolutely necessary step before deployment in real scenarios, without jeopardizing people's safety. Although there is still room for improvement, the obtained results are fairly promising, especially when taking into account the level of difficulty and the safety constraints involved in the task, and they may really help to reduce the risk of injuring people during drone's deployment in urban areas, hence potentiating the use of drones in civilian applications.

Author contributions All authors contributed to the study conception and design. Material preparation, data collection and analysis, and the real-time implementation and validation were performed by HTP, development and implementation of the vision-based algorithm were performed by JGT, conceptualization, data analysis, and paper review were performed by ÁFA, experimental testing and data acquisition were performed by MÁGT, project administration and supervision, paper review and editing were performed by DMR. The first draft of the manuscript was written by HTP, and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Funding The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.

Data availability Data sharing not applicable to this article as no datasets were generated during the current study.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval Not applicable.

Consent to participate Not applicable.

References

Abdollahzadeh S, Proulx P.-L, Allili M.S, Lapointe J.-F (2022) Safe landing zones detection for uavs using deep regression. In: 2022 19th conference on robots and vision (CRV), pp. 213–218 . https://doi.org/10.1109/CRV55824.2022.00035

Administration F.A (2021) Unmanned aircraft systems (UAS)

Alam MS, Oluoch J (2021) A survey of safe landing zone detection techniques for autonomous unmanned aerial vehicles (uavs). Expert Syst Appl 179:115091. https://doi.org/10.1016/j.eswa.2021.115091

Ariante G, Ponte S, Papa U, Del Core G (2021) Safe landing area determination (slad) for unmanned aircraft systems by using rotary lidar. In: 2021 IEEE 8th international workshop on metrology for AeroSpace (MetroAeroSpace), pp. 110–115 . https://doi.org/10.1109/MetroAeroSpace51421.2021.9511669

Authority U.K.C.A (2023) Rules and categories of drone flying. https://www.caa.co.uk/drones/rules-and-categories-of-drone-flying/flying-in-the-open-category/

Bassi E (2019) European drones regulation: today's legal challenges. In: 2019 International conference on unmanned aircraft systems

(ICUAS), pp. 443–450. https://doi.org/10.1109/ICUAS.2019.8798173
Bruff D (2005) The assignment problem and the hungarian method. Notes for Math 20(29–47):5
Castellano G, Castiello C, Mencar C, Vessio G (2020) Crowd detection in aerial images using spatial graphs and fully-convolutional neural networks. IEEE Access 8:64534–64544. https://doi.org/10.1109/access.2020.2984768
Chatzikalymnios E, Moustakas K (2022) Landing site detection for autonomous rotor wing uavs using visual and structural information. J Intell Robot Syst Theory Appl. https://doi.org/10.1007/s10846-021-01544-6
Delmerico J.A, Poranne R, Bogo F, Oleynikova H, Vollenweider E, Coros S, Nieto J.I, Pollefeys M (2022) Spatial computing and intuitive interaction: bringing mixed reality and robotics together. CoRR:2202.01493
Garcia-Pardo PJ, Sukhatme GS, Montgomery JF (2002) Towards vision-based safe landing for an autonomous helicopter. Robot Auton Syst 38(1):19–29
Gonzalez-Trejo JA, Mercado-Ravell DA (2021) Lightweight density map architecture for uavs safe landing in crowded areas. J Intell Robot Syst 102(1):7. https://doi.org/10.1007/s10846-021-01380-8
Gonzalez-Trejo J, Mercado-Ravell D, Becerra I, Murrieta-Cid R (2021) On the visual-based safe landing of uavs in populated areas: a crucial aspect for urban deployment. IEEE Robot Autom Lett 6(4):7901–7908. https://doi.org/10.1109/lra.2021.3101861
Guerin J, Delmas K, Guiochet J (2021) Certifying emergency landing for safe urban uav. In: 2021 51st Annual IEEE/IFIP international conference on dependable systems and networks workshops (DSN-W), pp. 55–62. IEEE Computer Society, Los Alamitos, CA, USA . https://doi.org/10.1109/DSN-W52860.2021.00020. https://doi.ieeecomputersociety.org/10.1109/DSN-W52860.2021.00020
Guerra W, Tal E, Murali V, Ryou G, Karaman S (2019) Flightgoggles: Photorealistic sensor simulation for perception-driven robotics using photogrammetry and virtual reality. In: 2019 IEEE/RSJ International conference on intelligent robots and systems (IROS), pp. 6941–6948. https://doi.org/10.1109/IROS40897.2019.8968116
Heo S-N, Chen J, Liao Y-C, Lee H-H (2022) Auto-splitting d* lite path planning for large disaster area. Intell Serv Robot 15:289–306. https://doi.org/10.1007/s11370-022-00416-8
Jaderberg M, Simonyan K, Zisserman A, Kavukcuoglu K (2015) Spatial transformer networks. preprint arXiv arXiv:1506.02025 [cs.CV]
Johnson A, Montgomery J, Matthies L (2005) Vision guided landing of an autonomous helicopter in hazardous terrain. In: IEEE international conference on robotics and automation. https://doi.org/10.1109/robot.2005.1570727
Kakaletsis E, Symeonidis C, Tzelepi M, Mademlis I, Tefas A, Nikolaidis N, Pitas I (2022) Computer vision for autonomous uav flight safety: an overview and a vision-based safe landing pipeline example. ACM Comput Surv 54:1–37. https://doi.org/10.1145/3472288
Khazetdinov A, Zakiev A, Tsoy T, Svinin M, Magid E (2021) Embedded aruco: a novel approach for high precision uav landing. Institute of Electrical and Electronics Engineers Inc., https://doi.org/10.1109/SIBCON50419.2021.9438855
Kinahan J, Smeaton AF (2021) Image segmentation to identify safe landing zones for unmanned aerial vehicles. arXiv: 2111.14557
Lee W, Alkouz B, Shahzaad B, Bouguettaya A (2021) Package delivery using autonomous drones in skyways, vol. 5, pp. 48–50. ACM. https://doi.org/10.1145/3460418.3479289. https://dl.acm.org/doi/10.1145/3460418.3479289
Liu W, Lis K, Salzmann M, Fua P (2019) Geometric and physical constraints for drone-based head plane crowd density estimation. In: 2019 IEEE/RSJ international conference on intelligent robots and systems (IROS). https://doi.org/10.1109/iros40897.2019.8967852
Liu W, Salzmann M, Fua P (2019) Context-aware crowd counting. In: 2019 IEEE/CVF Conference on computer vision and pattern recognition (CVPR), pp. 5094–5103. IEEE Computer Society, Los Alamitos, CA, USA. https://doi.org/10.1109/CVPR.2019.00524. https://doi.ieeecomputersociety.org/10.1109/CVPR.2019.00524
Marcu A, Costea D, Licaret V, Pirvu M, Leordeanu M, Slusanschi E (2018) Safeuav: learning to estimate depth and safe landing areas for uavs from synthetic data. In: European conference on computer vision (ECCV) UAVision Workshop
Ma Z, Wei X, Hong X, Gong Y (2019) Bayesian loss for crowd count estimation with point supervision. In: Proceedings of the IEEE international conference on computer vision, pp. 6142–6151
McTegg SJ, Kurdi FT, Simmons S, Gharineiat Z (2022) Comparative approach of unmanned aerial vehicle restrictions in controlled airspaces. Remote. Sens. 14:822
Mitroudas T, Balaska V, Psomoulis A, Gasteratos A (2023) Embedded light-weight approach for safe landing in populated areas. arXiv: 2302.14445 [cs.RO]
Mittal M, Mohan R, Burgard W, Valada A (2019) Vision-based autonomous UAV navigation and landing for urban search and rescue. CoRR:1906.01304
Nabavi Y, Asadi D, Ahmadi K (2022) Automatic landing control of a multi-rotor uav using a monocular camera. J Intell Robot Syst. https://doi.org/10.1007/s10846-022-01655-8
Nguyen TH, Cao M, Nguyen T.-M, Xie L (2018) Post-mission autonomous return and precision landing of uav, pp. 1747–1752. IEEE, https://doi.org/10.1109/ICARCV.2018.8581117, https://ieeexplore.ieee.org/document/8581117/
Orsolits H, Rauh S.F, Estrada J.G (2022) Using mixed reality based digital twins for robotics education. In: 2022 IEEE international symposium on mixed and augmented reality adjunct (ISMAR-Adjunct), pp. 56–59. https://doi.org/10.1109/ISMAR-Adjunct57072.2022.00021
Pinkam N, Newaz AAR, Jeong S, Chong NY (2019) Rapid coverage of regions of interest for environmental monitoring. Intel Serv Robot 12:393–406. https://doi.org/10.1007/s11370-019-00290-x
Saho K (2017) Kalman filter for moving object tracking: Performance analysis and filter design. In: de Oliveira Serra, G.L. (ed.) Kalman Filters-Theory for advanced applications. IntechOpen, https://doi.org/10.5772/intechopen.71731
Sanchez-Rodriguez J-P, Aceves-Lopez A, Martinez-Carranza J, Flores-Wysocka G (2020) Onboard plane-wise 3d mapping using super-pixels and stereo vision for autonomous flight of a hexacopter. Intel Serv Robot 13:273–287. https://doi.org/10.1007/s11370-020-00312-z
Sankaranarayanan S, Balaji Y, Jain A, Lim S, Chellappa R (2018) Learning from synthetic data: Addressing domain shift for semantic segmentation. In: 2018 IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp. 3752–3761. IEEE Computer Society, Los Alamitos, CA, USA. https://doi.org/10.1109/CVPR.2018.00395. https://doi.ieeecomputersociety.org/10.1109/CVPR.2018.00395
Shah S, Dey D, Lovett C, Kapoor A (2018) Airsim: High-fidelity visual and physical simulation for autonomous vehicles. Springer Proceedings in Advanced Robotics 5:621–635. https://doi.org/10.1007/978-3-319-67361-5_40
SilverTm: city park environment collection LITE (2021). https://www.unrealengine.com/marketplace/en-US/product/city-park-environment-collection-lite
Tzelepi M, Tefas A (2019) Graph embedded convolutional neural networks in human crowd detection for drone flight safety. IEEE Trans Emerg Topics Comput Intell. https://doi.org/10.1109/tetci.2019.2897815

Wang J, Wei C (2023) A novel air-ground coordinated approach for UAV autonomous landing on a mobile platform. In: Fu W, Gu M, Niu Y (eds.) Proceedings of 2022 international conference on autonomous unmanned systems (ICAUS 2022), pp. 2033–2043. Springer, Singapore

Yang L, Wang C, Wang L (2022) Autonomous uavs landing site selection from point cloud in unknown environments. ISA Trans 130:610–628. https://doi.org/10.1016/j.isatra.2022.04.005

Zhang Y, Zhou D, Chen S, Gao S, Ma Y (2016) Single-image crowd counting via multi-column convolutional neural network. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). https://doi.org/10.1109/cvpr.2016.70

Zhu P, Wen L, Du D, Bian X, Hu Q, Ling H (2020) Vision meets drones: past, present and future. preprint arXiv:2001.06303