AIAA SciTech Forum
6-10 January 2025, Orlando, FL
AIAA SCITECH 2025 Forum

10.2514/6.2025-1322

Verification and Validation of a Vision-Based Landing System for Autonomous VTOL Air Taxis

Ayoosh Bansal*†
University of Illinois at Urbana-Champaign, Urbana, Illinois, 61801

Duo Wang*‡ §
University of Nevada, Reno, Nevada, 89557

Mikael Yeghiazaryan*¶, Yangge Li‖, Chuyuan Tao**
University of Illinois at Urbana-Champaign, Urbana, Illinois, 61801

Hyung-Jin Yoon††
Tennessee Technological University, Cookeville, Tennessee, 38505

Prateek Arora‡‡, Christos Papachristos§§, Petros Voulgaris¶¶
University of Nevada, Reno, Nevada, 89557

Sayan Mitra***, Lui Sha††† and Naira Hovakimyan‡‡‡
University of Illinois at Urbana-Champaign, Urbana, Illinois, 61801

Autonomous air taxis are poised to revolutionize urban mass transportation. A key challenge inhibiting their adoption is ensuring the safety and reliability of the autonomy solutions that will control these vehicles. Validating these solutions on full-scale air taxis in the real world presents complexities, risks, and costs that further convolute the challenge of ensuring safety and reliability of these autonomous vehicles. Verification and Validation (V&V) frameworks play a crucial role in the design and development of highly reliable systems by formally verifying safety properties and validating algorithm behavior across diverse operational scenarios. Advancements in high-fidelity simulators have significantly enhanced their capability to emulate real-world conditions, encouraging their use for validating autonomous air taxi solutions, especially during early development stages. This evolution underscores the growing importance of simulation environments, not only as complementary tools to real-world testing but as essential platforms for evaluating algorithms in a controlled, reproducible, and scalable manner.

This work presents a V&V framework for a vision-based landing system for air taxis with vertical take-off and landing (VTOL) capabilities. Specifically, we use Verse, a tool for formal verification, to model and verify the safety of the system by obtaining and analyzing the reachable sets. To conduct this analysis, we utilize a photorealistic simulation environment. The simulation environment, built on Unreal Engine, provides realistic terrain, weather, and sensor characteristics to emulate real-world conditions with high fidelity. To validate the safety analysis results, we conduct extensive scenario-based testing to assess the reachability set and robustness

*Authors contributed equally.
†Postdoctoral Research Associate, Computer Science, 201 North Goodwin Avenue, Urbana, IL 61801
‡Postdoctoral Research Associate, Mechanical Engineering Department, 1664 N. Virginia Street, Reno, NV 89557
§Visiting Scholar at UIUC, Mechanical Science & Engineering Department, 105 S Mathews Ave, Urbana, IL 61801
¶Visiting Scholar at UIUC, Mechanical Science & Engineering Department, 105 S Mathews Ave, Urbana, IL 61801
‖Graduate Student, Electrical and Computer Engineering, 306 N. Wright St., Urbana, IL 61801
**Graduate Student, Mechanical Science & Engineering Department, 105 S Mathews Ave, Urbana, IL 61801
††Assistant Professor, Mechanical Engineering Department, 1 William L Jones Dr, Cookeville, TN 38505
‡‡Graduate Student, Department of Computer Science & Engineering, 1664 N. Virginia Street, Reno, NV 89557
§§Assistant Professor, Department of Computer Science & Engineering, 1664 N. Virginia Street, Reno, NV 89557
¶¶Professor, Mechanical Engineering Department, 1664 N. Virginia Street, Reno, NV 89557
***Professor, Electrical and Computer Engineering, 306 N. Wright St., Urbana, IL 61801
†††Donald B. Gillies Chair in Computer Science, Computer Science, 201 North Goodwin Avenue, Urbana, IL 61801, USA.
‡‡‡Professor, Mechanical Science & Engineering Department, 105 S Mathews Ave, Urbana, IL 61801

of the landing algorithm in various conditions. This approach showcases the representativeness of high-fidelity simulators, offering an effective means to analyze and refine algorithms before real-world deployment.

I. Introduction

The rapid development of autonomous aerial vehicles has significantly advanced modern aviation, enabling advancements in applications such as surveillance, delivery, agriculture, and disaster response. Vertical Take-Off and Landing (VTOL) aircrafts are a class of aerial vehicles capable of ascending and descending vertically, eliminating the need for runways. This capability is achieved through various designs, including rotorcraft, tiltrotor aircraft, and various emerging electric VTOL models. These systems promise efficiency and mobility, but ensuring their reliability remains a critical challenge, particularly for safety-critical operations such as landing in a cluttered urban environment. Autonomy algorithms, which govern these systems, are required to function reliably across a wide range of operational conditions, making rigorous Verification and Validation (V&V) frameworks indispensable. These frameworks systematically verify safety properties and validate the performance of these algorithms, ensuring they can be deployed with high confidence in real-world environments.

Robustness of the autonomy algorithms is integral to the safe and efficient operation of autonomous VTOL aircrafts [1–3]. Autonomy systems typically comprise of three major components: perception, path planning, and control. Perception algorithms enable the vehicle to interpret its surroundings by processing data from onboard sensors such as cameras, LiDAR, and radar to identify suitable landing zones and assess environmental features [4]. Path planning algorithms compute feasible and optimized trajectories that consider vehicle dynamics, environmental constraints, and mission objectives [5]. Control algorithms ensure the precise execution of these trajectories, maintaining stability and robustness throughout complex maneuvers such as vertical landing [6, 7]. The seamless integration of these components is essential for achieving reliable performance, as failures in any one of them can compromise the overall safety of the system. This interconnected nature of autonomy algorithms underscores the need for holistic development and testing to ensure their reliability across diverse and dynamic operational scenarios.

Hybrid system verification offers a mathematical framework to capture and analyze the complex interactions of continuous dynamics and discrete transitions that characterize autonomous systems. The increasing complexity of such systems, driven by advanced sensors, perception modules, and controllers, has spurred the development of tools to analyze both linear and nonlinear system behaviors. Validation, on the other hand, addresses the practical aspects of autonomy algorithm performance by testing them in simulated or real-world environments. It aims to assess whether these systems meet operational requirements when subject to realistic conditions, including sensor noise, environmental variability, and actuator imperfections [8]. Together, verification and validation provide a complementary approach to assessing autonomy algorithms, ensuring both theoretical soundness and practical reliability.

Current V&V practices for autonomy algorithms rely on simulation environments due to their cost-effectiveness and controllability. Simulators enable repeatable experiments, controlled scenario design, and accelerated testing. However, inevitably, simulation environments cannot capture the full complexity and details of the real-world, leading to the simulation-to-reality (sim-to-real) gap. Sim-to-real gap here refers to the differences in an autonomy's solution performance within the simulated environment versus the real-world, specifically, the autonomy algorithms that perform well in simulations may fail to achieve the same level of reliability in real-world applications [9, 10]. This gap arises from factors such as incomplete sensor modeling, unrealistic environmental interactions, and oversimplified dynamics in the simulation environment. The persistence of this gap poses challenges for safety-critical applications, e.g., autonomous landing, where small inaccuracies can lead to significant consequences.

Recent advances in simulation technologies, particularly those leveraging 3D rendering engines like Unreal Engine [11], have mitigated some of the limitations of traditional simulation environments by providing high-fidelity virtual settings. CARLA [12], a simulation platform originally designed for ground vehicles and built on Unreal Engine, has been extended to support aerial vehicles [13]. This extension enables the testing of perception and control algorithms for air vehicles in realistic simulated environments. However, despite these advancements, significant challenges persist in achieving the level of fidelity required to close the sim-to-real gap. Addressing these challenges necessitates systematic efforts to enhance simulation realism, ensure algorithm robustness across diverse scenarios, and streamline the transfer of autonomy solutions from simulation to real-world applications.

In this work, we focus on the V&V of vision-based landing algorithms for VTOL aircraft within a photorealistic simulation environment. Vision-based algorithms present unique validation challenges due to their reliance on high-quality sensor data and sensitivity to unpredictable factors such as lighting, texture, and occlusions. Our approach utilizes

the *Verse* [14] library to formally verify safety properties via reachability analysis, ensuring that the landing algorithm adheres to safety constraints under diverse conditions. The simulation environment offers a detailed representation of terrain, sensor characteristics, and environmental variations, facilitating realistic validation of perception and planning components. Through multiple experimental settings, we assess the landing algorithm's robustness and analyze the vehicle's reachable set, providing a rigorous evaluation of its performance. By focusing on a photorealistic simulation environment, we provide insights into the strengths and limitations of current V&V methods and lay the groundwork for future integration of these algorithms into hardware platforms. This integration will improve the overall V&V process, ensuring the reliability and safety of autonomous aerial systems in the real-world.

The key contributions of this paper are:

1) We demonstrate the application of formal verification techniques using *Verse*, providing a detailed analysis of safety properties of our landing algorithms and reachability under varying conditions.
2) We present a validation framework that incorporates extensive scenario testing. Our work highlights its potential for bridging the sim-to-real gap and guiding the deployment of future hardware systems.

By addressing these challenges, this study contributes to advancing the reliability and safety of autonomy algorithms, facilitating their deployment in aviation systems with increased confidence.

II. Literature Review

Autonomous operations of VTOL Unmanned Aerial Vehicles (UAVs) rely heavily on robust V&V processes, advanced autonomy algorithms, and high-fidelity simulation environments. These elements are critical for ensuring reliable and safe performance enabling the future deployment of autonomy algorithms in real-world applications. In this context, V&V efforts are specifically aimed at verifying the algorithms underlying autonomous decision-making, ensuring their robustness and correctness in various operational scenarios. This section reviews related works and their limitations, highlighting the gaps that motivate the contributions made in this work.

Tiltrotor VTOL UAV: UAVs are generally classified into multirotor, fixed-wing, and hybrid VTOL types, each offering distinct operational advantages and limitations. *Multirotor UAVs*, such as quadcopters, are capable of vertical take-off, landing, and hovering, making them suitable for tasks like aerial photography and surveillance in confined areas. However, they typically have limited flight endurance and speed due to the high energy consumption required to maintain lift [15]. *Fixed-wing UAVs* resemble traditional airplanes, utilizing wings to generate lift, which allows for longer flight endurance and higher speeds, making them ideal for missions like large-area surveillance. Nonetheless, they require runways or launch systems for take-off and landing and lack the ability to hover [16].

Hybrid VTOL UAVs combine the vertical take-off and landing capabilities of multirotors with the efficient forward flight of fixed-wing aircraft, offering operational flexibility in diverse environments. This hybrid capability enables hybrid VTOL UAVs to perform missions that demand both hovering and long-range flight without the need for runways. The configuration of hybrid VTOL aircraft can vary, for example, tiltrotor (e.g., Bell V-280 [17]) and tilt-wing (e.g., NASA GL-10 [18]). A notable example of a hybrid VTOL UAV is the MiniHawk-VTOL [19–21], whose small scale is particularly suitable for testing autonomy algorithms. This rapidly prototyped tricopter/fixed-wing hybrid aircraft is designed for autonomous operations, featuring solar-recharge capability to support extended missions without the need for physical intervention. The MiniHawk-VTOL exemplifies the practical application of hybrid VTOL technology, combining efficient flight performance with operational flexibility. In this work, we utilize the MiniHawk-VTOL platform to explore and validate autonomy algorithms, leveraging its design to address challenges inherent in VTOL UAV operations.

V&V for autonomy algorithms: Much of the existing research in autonomy algorithms (perception, path planning and control) has been developed primarily for multirotor and fixed-wing platforms, with limited direct application to hybrid VTOL configurations. *Perception algorithms*, which are minimally influenced by the characteristics of hybrid VTOL UAVs, can leverage established methods from multirotor and fixed-wing UAVs. For instance, vision-based systems for obstacle avoidance and target tracking, commonly used in multirotors [22, 23], and sensor fusion techniques for situational awareness in fixed-wing UAVs [24, 25], can be adapted to hybrid VTOL platforms without significant modifications. *Path planning* and *control* algorithms, however, require greater customization due to the unique dynamics of hybrid VTOL UAVs, which must transition seamlessly between hovering and forward flight. Algorithms like A* and its variants [26, 27], developed for multirotor and fixed-wing platforms, provide a foundation for hybrid VTOL path planning but often require further refinement to account for operational scenarios. Similarly, control strategies such as PID controllers and model predictive control (MPC) are insufficient on their own for hybrid VTOLs, which demand hybrid approaches capable of managing complex transitions between flight modes [7]. Some state-of-the-art

controllers require direct control of motor thrust and torque to achieve precise trajectory tracking and adaptability under complex conditions. However, in outdoor hardware experiments, directly controlling motor thrust and torque is typically impractical and unsafe due to the high risk of crashing the vehicle. To mitigate these risks, this work adopts a 3D A* planner for path planning and utilizes mature and conservative low-level PID controllers in ArduPilot flight stack [28]. This setup limits the control input to waypoints, providing a robust and reliable approach for future outdoor experiments while reducing the likelihood of hardware failures.

Traditional V&V techniques, often designed for deterministic and linear systems, struggle with the stochastic nature and adaptive behaviors of autonomy algorithms, particularly in hybrid VTOL UAVs where transitions between hover and forward flight modes introduce non-linear dynamics and unpredictable environmental interactions [29]. Tools like *Hylaa* [30], *DryVR* [31], and *CORA* [32] have enabled the verification of systems with thousands of continuous variables, addressing challenges such as scalability and computational efficiency. Building on these advancements, the *Verse* library [14] extends the capabilities of hybrid system verification by enabling modeling and safety analysis of multi-agent systems with intuitive Python interfaces. *Verse* allows non-deterministic transitions and facilitates reachability analysis, offering a robust platform for analyzing the safety of complex autonomy algorithms. These tools not only ensure mathematical rigor but also provide actionable insights into system behavior under varying conditions [33, 34].

Simulation-Based V&V for Hybrid VTOL UAVs: Simulation platforms play an essential role in the Verification and Validation (V&V) of autonomy algorithms for hybrid VTOL UAVs, providing a controlled and repeatable environment to test performance of autonomy algorithms before real world deployment. Existing simulation-based V&V solutions face significant limitations [35] when applied to hybrid VTOL UAVs. Few frameworks are able to provide high fidelity environments, so they face challenges to accurately verify vision-based perception algorithms, particularly under complex lighting, weather, or terrain conditions. Additionally, these platforms lack dedicated models for hybrid VTOL UAV dynamics, such as the transitions between hovering and forward flight. Few existing solutions provide comprehensive support for hybrid VTOL UAVs, highlighting a critical gap in current simulation-based V&V. Addressing these shortcomings, this work leverages high fidelity photo-realistic simulator CARLA enhanced with VTOL-specific dynamics, MiniHawk (simulated using Gazebo) [20] and NASA GUAM [36], to ensure robust V&V of autonomy algorithms.

This paper aims to address these gaps by leveraging an advanced simulation environment integrated with simulation models of hybrid VTOL to enhance the robustness of simulation-based V&V for autonomy algorithms. By focusing on the challenges specific to hybrid VTOL UAVs, this work contributes to bridging the sim-to-real gap and advancing the reliability of autonomy systems for future real-world applications.

III. Simulation Environment

The simulation environment used in this work is designed to support high-fidelity simulations for autonomous air taxis operating in dynamic urban settings, as illustrated in Fig. 1. It builds on the popular urban environment simulator CARLA [12], which has been extensively utilized for the verification and validation (V&V) of autonomous ground vehicles in prior research [37–41]. CARLA, developed using Unreal Engine [11], enables the creation of photorealistic 3D environments with realistic graphics and high-fidelity physics simulations. It also provides emulation of various sensors, including cameras and LiDAR, generating corresponding sensor data for the simulated environments. However, CARLA natively supports ground vehicles only.

To address this limitation, simulation environments that integrate aerial vehicle support in CARLA were developed [13]. This simulation environment integrates external high-fidelity physics engines to emulate the aerial vehicle. The simulation initially supported an advanced urban air mobility passenger vehicle [36], shown in Fig. 1. To this, we add the MiniHawk-VTOL [20, 21], a compact tiltrotor aerial vehicle with VTOL capabilities, that is the focus of this work. MiniHawk's physics are simulated with high-fidelity in Gazebo [42], which computes the MiniHawk's pose. The automated landing system evaluated in this work, as integrated within the simulation framework, is illustrated in Fig. 2. Details of the landing system design are provided in Section V.A. All components of the autonomous system communicate using the publisher-subscriber model facilitated by the Robot Operating System (ROS) [43] middleware.

This framework leverages advanced simulation tools to create a robust and flexible testing environment that closely mimics real-world conditions, narrowing the sim-to-real gap from the simulation side. While Gazebo provides a graphical interface for visualizing the vehicle and its surroundings, the visualizations and corresponding sensor data lack realism, leading to a sim-to-real gap when using perception modules for decision-making. By relying on CARLA's high-fidelity simulation, the gap is significantly reduced as instead of using coarse images from Gazebo, we utilize photorealistic images generated by CARLA.

Fig. 1 An overview of a simulation environment within CARLA with integrated air taxi.

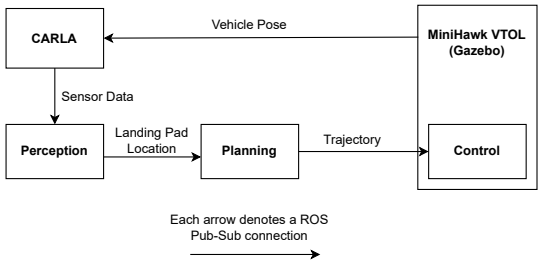


Fig. 2 An overview of the simulation loop for the automated landing system in air taxis.

The simulation environment also offers extensive customization options for simulation scenarios. Configurable parameters include the initial pose of the ego vehicle*, the target landing pad location, weather conditions, wind conditions, vehicle parameters, and sensor configurations. The scenarios and conditions used in this work are described in Section V.B.

IV. Verification and Validation Framework

A. Formal Verification with *Verse*

We are interested in determining whether the VTOL aircraft can land safely in time without colliding with any obstacles. An execution of the VTOL landing system is the sequence of states that the aircraft travels starting from a certain initial state. For the VTOL landing system, given a set of initial states X_0, the reachable set at time t, denoted by $Reach(X_0, t)$, is the union of all possible executions starting from the initial set X_0 at time t. Given the landing region G and the unsafe set U as the requirement for safe landing, the aircraft starting from X_0 can land safely at time t_f without colliding with the obstacles if

$$Reach(X_0, t_f) \subseteq G \text{ and} \tag{1}$$

$$Reach(X_0, t) \cap U = \emptyset, \ \forall\, 0 \leq t \leq t_f. \tag{2}$$

In this work, we employ the hybrid system verification tool Verse [14], a Python-based library designed for verifying multi-agent hybrid scenarios, to analyze the VTOL aircraft system. Verse uses black-box simulators to describe system dynamics and implements the simulation-based reachability algorithm from DryVR [31] to compute over-approximations of reachable sets with probabilistic accuracy guarantees. We integrate the simulation pipeline described in Section III as the black-box simulator for Verse.

Fig. 3 illustrates the VTOL landing verification architecture. Starting from a prescribed set of initial conditions, Verse randomly samples initial states and generates corresponding trajectories using the simulation pipeline. These trajectories are then used to compute an over-approximation of the reachable set. By checking this reachable set against obstacle regions and desired goal sets, we can determine whether the system meets the specified safety and performance requirements.

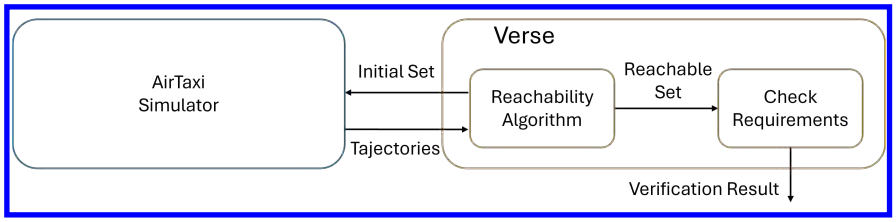
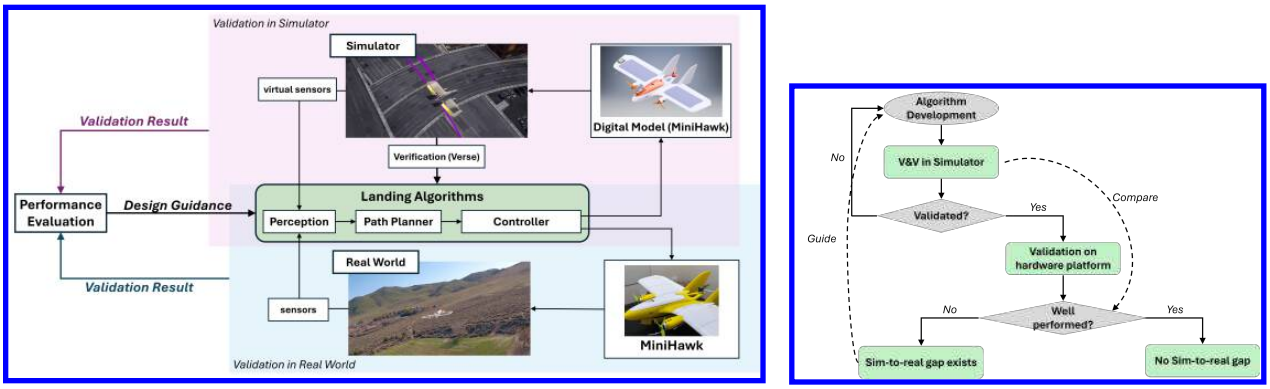


Fig. 3 Verification Architecture for VTOL landing system.

*The ego vehicle refers to the vehicle that is autonomously controlled and observed in the simulation.

B. Validation Methodology and Bridging the Sim-to-Real Gap

Unlike verification, which establishes the correctness of algorithms under formal specifications, validation focuses on evaluating the performance of the system under diverse conditions to confirm their real-world applicability. For hybrid VTOL UAVs, this includes analyzing the ability to handle transition phase, navigate dynamic environments and adapt to disturbances. One challenge in validation is bridging the simulation-to-real (sim-to-real) gap. Simulators often simplify certain aspects of reality, such as sensor noises and environmental disturbances. These simplifications can lead to overestimation of algorithm performance and result in failures during hardware deployment. Given the complex dynamics and the requirement of outdoor experiments, this challenge is acute for hybrid VTOL vehicles. To address this problem, some real-to-simulation (real-to-sim) efforts are necessary, which involve incorporating characteristics from the real world into the simulator to make the simulation environment as close as possible to reality, thereby narrowing the sim-to-real gap. Furthermore, after obtaining validation results of the verified algorithms from simulator and real-world experiments, evaluating the performance differences can provide valuable feedback to guide the design and refinement of autonomy algorithms, contributing to further narrowing of the sim-to-real gap, as shown in Fig. 4.



(a) (b)

Fig. 4 A framework using simulated and real-world validation results to improve verified autonomy algorithms.

In this work, to narrow the sim-to-real gap, we adopt a multi-faceted approach:

- Utilizing high-fidelity simulation environment developed on CARLA and enhanced with VTOL dynamics, to provide a repeatable environment for verifying the autonomy algorithms and testing.
- Using the model of the real-world vehicle platform (MiniHawk) in simulation.
- Training perception algorithms with real-world photos enhances robustness by exposing models to visual variations that simulators cannot fully imitate.

To further address the gap, we propose that the following approaches can help:

- Integrating autonomy algorithms into real-time systems that combine hardware components with simulated environments to evaluate performance under realistic conditions (hardware-in-the-loop testing).
- Conducting controlled outdoor experiments that progressively increase in complexity, starting with simpler scenarios to identify and address potential failure points before deploying in fully operational environments.

These methods provide a framework for ensuring the correctness of autonomy algorithms for air taxis and enabling deployment in real-world applications.

V. Case Study and Analysis

A. Autonomous Landing System

The simulation environment and the V&V framework developed in this work are designed to be general and support a wide range of autonomy algorithms for VTOL UAVs. To demonstrate the effectiveness of the process and to facilitate a detailed analysis, we select the following specific perception, path planning, and control algorithms as representative case studies. These choices are motivated by their relevance to MiniHawk-VTOL and their alignment with the objectives of this study.

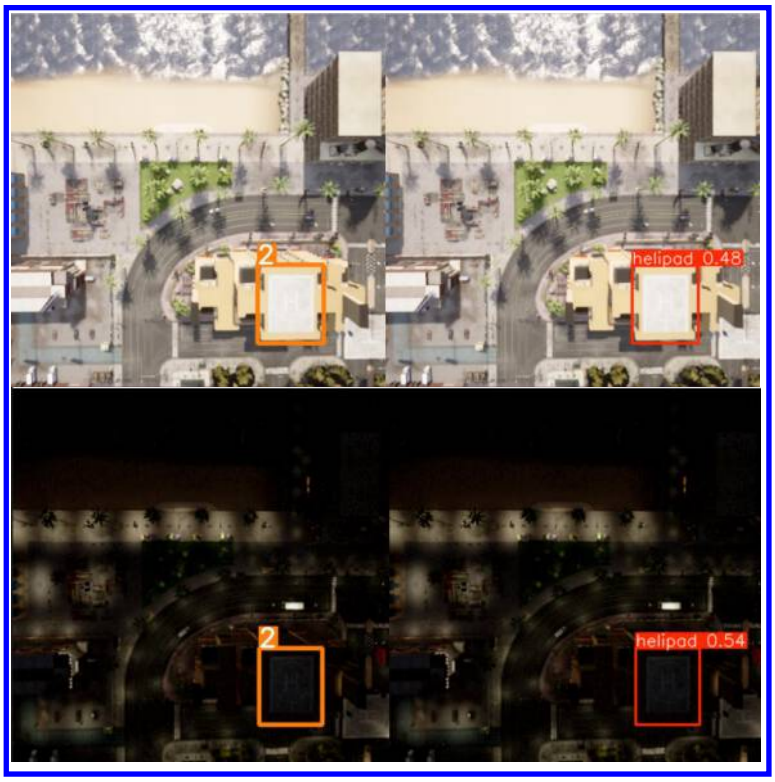


Fig. 5 Examples of helipad detection under different lighting conditions, photos are taken in CARLA from a downward-facing RGB camera attached to the belly of the aircraft.

1. Perception - Landing Zone (Helipad) Detection

To ensure the safety and reliability of autonomous landing systems in hybrid VTOL UAVs, this work integrates a vision-based detection algorithm, using a YOLO (You Only Look Once) [44] deep neural network (DNN) for helipad detection [45]. The system ensures precise identification and localization of the potential landing area and forms the basis for path planning.

The detection system is built on a dataset of annotated helipad images, which includes real-world images of helipads captured from aerial views from Google Earth images [46]. Helipad locations are marked in each image with bounding boxes, and the synthetic data augmentation includes varying lighting conditions (e.g., night) and weather conditions (e.g., heavy rain), enhancing model robustness to real-world variations. The YOLOv8 model [47] is trained on this dataset to produce bounding boxes indicating helipad locations and the detection performance is evaluated using mean Average Precision at various Intersection-over-Union thresholds. Robustness and reproducibility are further ensured by disabling random data augmentations during controlled training cycles so that it allows for analysis of hyper-parameter effects. A Bayesian optimization framework is employed to ensure algorithm's robustness in dynamic conditions. The helipad detection algorithm serves as an input to the path planning module. The center of the detected bounding box is utilized as the target end point for the planned trajectory to ensure accurate alignment of the UAV with the landing zone in the descending phase.

In this work, to accurately evaluate the effects of the perception on performance, we assume three perception setups: 1) a synthetic ideal perception system (Scenario1), 2) a synthetic perception system with bounded behavior (scenarios 2 and 3), and 3) YOLOv8 based perception system (scenarios 4 and 5). These setups and scenarios are detailed in Section V.B.

Finally, while we evaluate the impact of obstacles between the aircraft and the landing pad, the detection of the obstacles itself is not a focus of this work. In this work, the ideal obstacle information is assumed to be available. Reliable obstacle detection and collision avoidance is an active area of research with significant prior solutions [13, 48–51].

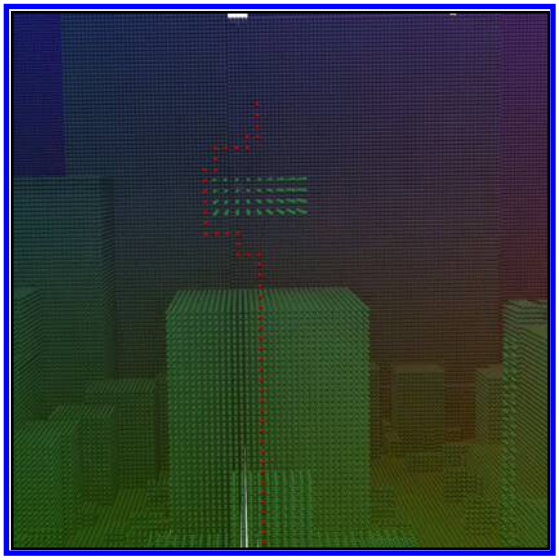


Fig. 6 Visualization of A* path planning in a 3D environment. The red dots indicate the planned trajectory. All other dots highlight obstacles.

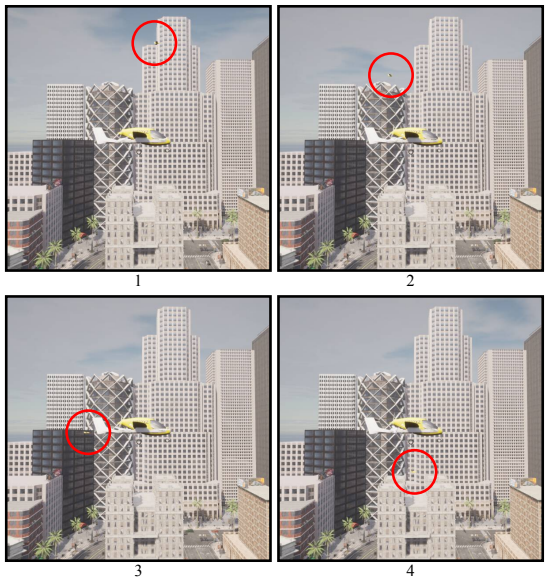


Fig. 7 Simulation of the A path planning algorithm to avoid another hovering aerial vehicle during landing. MiniHawk follows the trajectory depicted in Fig. 6. The red circles indicate its position at different frames.

2. Path Planning - A planner*

In this work, the A* algorithm is used to plan the path of VTOL aircraft navigating in a three-dimensional environment. A* is a graph-based search algorithm that determines the shortest path by minimizing the total cost function:

$$f(n) = g(n) + h(n), \tag{3}$$

where $g(n)$ represents the actual cost from the start node to the current node n, and $h(n)$ denotes the heuristic estimate of the cost from n to the goal. The algorithm guarantees optimality under specific conditions. First, the heuristic $h(n)$ must be admissible, meaning it does not overestimate the actual cost to the goal ($h(n) \leq h^*(n)$, where $h^*(n)$ is the true cost). Second, $h(n)$ must be consistent, satisfying the inequality $h(n) \leq c(n, n') + h(n')$ for any edge from n to n', where $c(n, n')$ is the cost of the edge. These properties ensure the correctness and efficiency of the algorithm [52].

A* operates on a discretized 3D representation of the environment, where nodes correspond to spatial points, and edges define feasible transitions between them [53]. The combination of the cost-to-go ($g(n)$) and the heuristic ($h(n)$) allows A* to effectively balance exploration and exploitation during the search process. However, the computational complexity of A* increases significantly with the dimensionality and density of the search space, which poses challenges for real-time implementations, especially on resource-constrained systems.

To mitigate these challenges, the action space of the VTOL is limited to six discrete motions: forward, backward, left, right, upward, and downward. By constraining the available actions, the branching factor of the graph is reduced, thereby decreasing the computational burden of the algorithm. This makes the implementation feasible for onboard systems with limited processing power. However, this simplification comes at the cost of path quality, as the resulting trajectories are often less smooth and may include abrupt transitions.

To address the difficulties in tracking such paths, robust control strategies are required to compensate for the lack of continuity in the planned trajectories. Despite these limitations, the proposed approach achieves a balance between computational efficiency and path-planning performance. This makes A* suitable for real-time applications in autonomous VTOL navigation, demonstrating its ability to provide efficient and reliable path planning under tight resource constraints.

Fig. 6 illustrates the operation of the A* algorithm in a 3D environment. The grid represents the discretized obstacles in the search space. The red points indicate the waypoints defining the trajectory generated by the path planner. All other points represent obstacles. Fig. 7 shows the resulting motion of the MiniHawk-VTOL around an existing

obstacle — another aerial vehicle.

3. Controller

For this case study, we utilize the ArduPilot flight stack as the primary controller for the MiniHawk in the simulation. The same ArduPilot has been implemented on the real MiniHawk using the mRo PixRacer Pro flight controller. This hardware-software combination ensures consistency in interface and behavior between simulation and physical platforms, significantly reducing the effort required to transplant algorithms verified in simulators onto the real vehicle. This streamlined transition facilitates an efficient and systematic V&V process. ArduPilot employs a PID-based control system that processes waypoint commands to manage motor inputs in a closed-loop manner. This conservative control approach prioritizes safety and robustness, particularly during outdoor experiments where environmental uncertainties can pose risks. Additionally, the tiltrotor functionality of the MiniHawk is natively supported by ArduPilot, enabling smooth transitions between hover and forward flight modes while maintaining control stability.

Future advancements may involve the integration of modern control strategies, such as the $\mathcal{L}_1$ adaptive control architecture [54], to enhance the adaptability and performance under uncertain conditions. However, the primary challenges arise not from the algorithms themselves but from the risks inherent in hardware experiments. Low-level interfaces needed to control motor thrusts directly and bypass the safety constraints of waypoint-based system, demanding precise calibration and increasing the risk of instability during outdoor tests, where the margin for error is small. By utilizing the PID-based controller in ArduPilot, safety and reliability are prioritized, providing a baseline for experiments. This choice enables us to focus on demonstrating the full V&V framework while maintaining flexibility for integrating advanced control strategies in future work.

B. Scenario Setting

We consider emergency scenarios where MiniHawk must land on a helipad located on the top of a building. The landing procedure in scenarios 1–3 begins immediately, with varying levels of prior knowledge about the target landing point from external sources (e.g., GPS), as described below. In scenarios 4 and 5, we assume no such information is available, relying solely on perception (§V.A.1) to detect the landing zone on the rooftop, with only the height of the landing pad known. To evaluate the performance, correctness, and robustness of the autonomy algorithms under different conditions, we consider the following scenarios aimed at verifying the landing system's ability to handle uncertainties and land safely. In each scenario, the MiniHawk vehicle starts above the landing zone and navigates to the landing pad. The center point of the landing pad on top of the building is $(0, 0, 29)$, with all dimensions in meters along the X, Y, and Z axes, respectively.

Scenario 1: *uncertainties in initial conditions with fixed landing point.* In this scenario, the MiniHawk initiates the landing procedure from varying initial positions. The landing point on the helipad is known and fixed at $(0, 0, 29)$. The initial position is within a bounded region above the landing pad $(\pm 5, \pm 5, 75 \pm 5)$. Within this bounded region the initial position is distributed uniformly along all axes. The uncertainty in initial position represents the variations in the trajectories with which the aerial vehicle may approach the landing pad. The fixed landing target mimics the conditions when an HD map of the landing location and error free GPS or GNSS[†] sensors are available to localize the aerial vehicle with respect to the landing target.

Scenario 2: *uncertainties in both initial conditions and landing points.* Building on the first scenario, this scenario introduces uncertainties in the location of the landing point. While the initial conditions remain the same as in Scenario 1, the target landing point is now within a region around the helipad's center, at a fixed height. The X and Y coordinates follow a normal distribution with $\mu = (0, 0, 29)$ and $\sigma = (1.5, 1.5, 0)$. The variance was chosen to ensure that the sampled target remains within the actual helipad. This mimics real-world conditions where the exact placement of the landing zone may vary. Specifically, this emulates the uncertainty in the landing pad position estimation by the sensors used to localize the vehicle, especially in the urban environments where such air taxi services are designed to operate in [55, 56].

Scenario 3: *shared airspace with intruder aircraft.* In this scenario, in addition to the conditions outlined in Scenario 2, a larger aircraft intrudes into MiniHawk's landing path, simulating a shared airspace environment with other aerial vehicles. The intruding aircraft is static at $(0, 0, 60)$, and has the following approximate dimensions: $\Delta x = 6.4$ m, $\Delta y = 10.8$ m, $\Delta z = 2.2$ m. The initial position of MiniHawk is sampled from a uniform distribution within the range: $x \in [-5.5, -3.0]$, $y \in [-1.5, 1.5]$, $z \in [73.0, 77.0]$. The range of initial positions is intentionally reduced to

[†]Global Positioning System (GPS), Global Navigation Satellite System (GNSS)

enforce consistent behavior by MiniHawk when navigating around the obstacle, specifically ensuring it flies behind the intruding vehicle. This scenario highlights the integration of collision avoidance and safe landing capabilities in a complex, shared airspace environment. The reasons for the change in initial position generation, for this scenario and scenario 5, is discussed in Section V.D.

Scenario 4: *uncertainties in initial conditions with vision-based landing pad detection.* This scenario is similar to Scenario 2, but the landing target is determined using vision-based landing pad detection, as described in Section V.A.1. The landing pad has a fixed height, which is assumed to be known. The X and Y coordinates of the landing target are defined as the midpoint of the bounding box detected by the perception system. Using the camera's intrinsic and extrinsic matrices, we convert this center point from camera coordinates to global coordinates and set it as the target. This scenario highlights the importance and impact of using real perception to detect the landing pad in the absence of ground-truth knowledge.

Scenario 5: *intruder aircraft with landing pad detection.* This scenario combines elements of Scenarios 3 and 4. Starting from scenario 3, the landing target is now determined by the perception system, while the same intruder aircraft from Scenario 3 obstructs the Minihawk ego vehicle's landing path. Notably, the intruder aircraft occludes the landing pad from the Minihawk's perspective.

C. Reachability Analysis

We perform reachability analysis of all the scenarios described in Section V.B using hybrid system verification tool Verse and the results are discussed in this section. For all scenarios, Verse computes an over-approximation of the reachable set, which we term a *reachtube*, up to a time horizon of $t_f = 100$s. The reachable sets are computed from 10 trajectories simulated with a 0.25s time step. Despite the high fidelity of the Minihawk simulation in Gazebo, inevitably, some differences exist. Rarely, the Minihawk simulation can destabilize, likely due to numerical error accumulation in the physics simulation. Such instances were manually identified and removed from the reachability analysis, as they do not reflect the behavior of the real vehicle in identical conditions. With the computed reachtube, *Verse can verify that the MiniHawk will land within the landing pad while avoiding obstacles*. We now present and discuss the results for each scenario.

Scenario 1: *uncertainties in initial conditions with fixed landing point.* The simulated trajectories, reachable sets for each dimension, and a 3D visualization of the reachable set for Scenario 1 are shown in Fig. 8. It can be observed that all simulated trajectories are contained within the reachable set, providing evidence that the reachable set computed by Verse over-approximates the actual reachable set of the system. Despite initial uncertainty, the reachable set converges to a small region contained within the goal area. Additonally, the reachable set does not intersect with any of the gray boxes representing environmental obstacles, such as trees and buildings. These verification results confirm that MiniHawk can land safely without colliding with obstacles.

Scenario 2: *uncertainties in both initial conditions and landing points.* The results for Scenario 2 are shown in Fig. 9. From the plot, it is clear that uncertainties in the endpoint of the planned path, particularly those associated with the landing zone, cause the final portion of the reachable set in Scenario 2 to be larger than in Scenario 1. Despite this, the reachable set still converges toward the target region, demonstrating that the MiniHawk can successfully land on the targeted helipad on the building, even with uncertainty in the target location.

Scenario 3: *shared airspace with intruder aircraft.* This figure demonstrates that the reachable set of the MiniHawk does not intersect with the unsafe space occupied by the intruder (depicted in red in the 3D plot), indicating that the MiniHawk's path planner effectively avoids collisions with the aircraft intruding into the landing path. The reachable set for each dimension is also provided in Fig.10. From these plots, we observe that the collision avoidance behavior introduces greater uncertainties in the MiniHawk's motion, resulting in a reachable set with expanded volume, particularly along the y-axis.

Scenario 4: *uncertainties in initial conditions with vision-based landing pad detection.* The results for Scenario 4 are shown in Fig. 11. These results reveal that Verse computes an over-approximation of the actual reachable set, despite the uncertainties introduced by the perception pipeline. The outcome is similar to that of Scenario 2. However, in Scenario 4, unlike Scenario 2 where the radius of the reachable set in the z-dimension converges to zero, uncertainties persist across all x, y, and z dimensions at t_f due to the uncertainty in the predicted landing point. The 3D plot further confirms that the reachable set does not intersect with any obstacles, demonstrating that the landing process is collision-free.

Scenario 5: *intruder aircraft with landing pad detection.* The reachability analysis for Scenario 5 is shown in Fig. 12. From these results, we observe that, despite the presence of an intruder and uncertainties from the perception

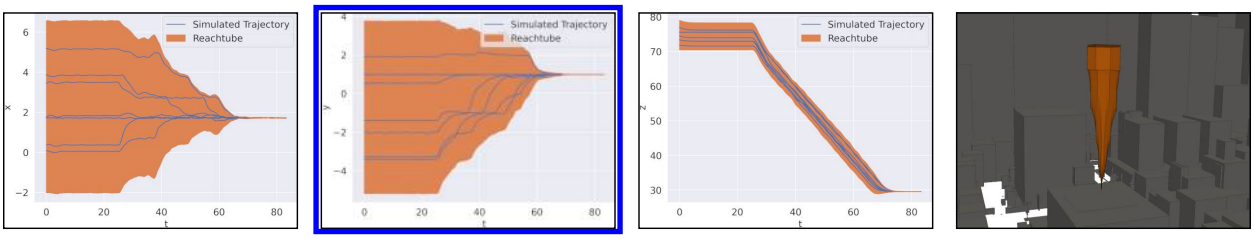


Fig. 8 Reachability analysis for Scenario 1: uncertainties in initial conditions with fixed landing point.

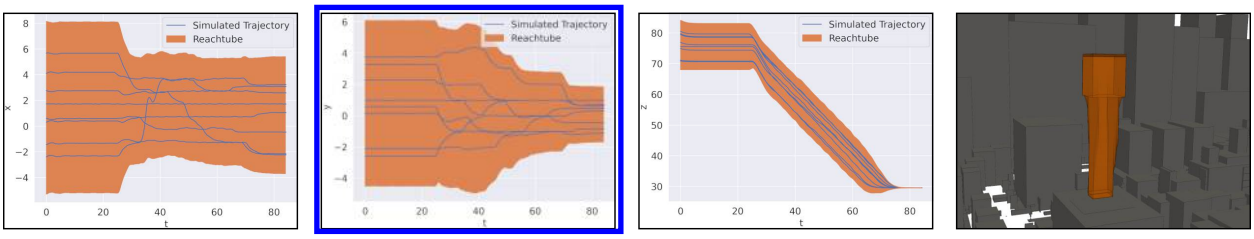


Fig. 9 Reachability analysis for Scenario 2: uncertainties in both initial conditions and landing points.

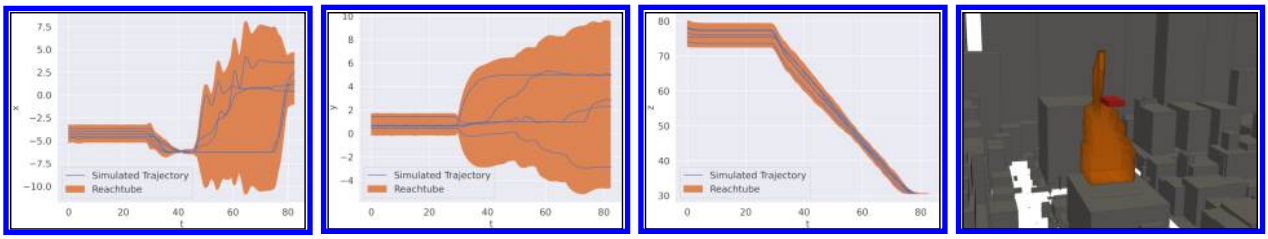


Fig. 10 Reachability analysis for Scenario 3: shared airspace with intruder aircraft.

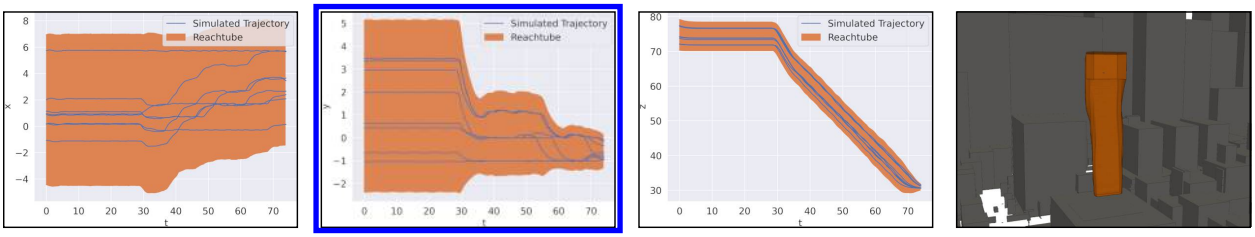


Fig. 11 Reachability analysis for Scenario 4: uncertainties in initial conditions with vision-based landing pad detection.

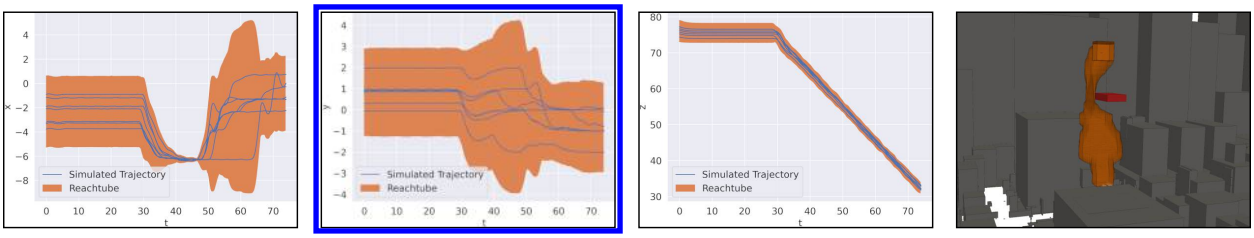


Fig. 12 Reachability analysis for Scenario 5: intruder aircraft with landing pad detection.

algorithm, Verse successfully computes an over-approximation of the reachable set that covers the randomly simulated trajectories, with its volume remaining unchanged. Compared to the results from Scenario 3, the uncertainty in the z-dimension is larger at t_f. However, despite the additional uncertainty from perception, the radius of the reachable set in the y-dimension is smaller in Scenario 5 than in Scenario 3. This reflects that the error distribution of the centers of the detected bounding boxes is narrower than the normal distribution used in Scenario 3. In the final subfigure, we see that the reachable set avoids the intruder, demonstrating that MiniHawk can safely land without colliding with the intruding aircraft, even in the presence of perception uncertainties.

D. Discussion

Reachability Analysis. The key result from the reachability analysis is that the goals of successful landing (1) and collision avoidance (2) are met. Within the range of conditions and constraints described across the evaluation scenarios, the autonomous landing system is safe and reliable.

Dataset Size. The fineness and reliability of the reachable set, currently derived from 10 trials per scenario, could be improved by expanding the dataset to capture more trials and a broader range of conditions. While this would enhance confidence in the results, it would also demand increased computational resources. Given the limited performance of the onboard companion computer, optimizing the efficiency of the autonomy algorithms is crucial to support such expansions without compromising real-time capabilities.

Simulator Fidelity. Enhancing the simulation environment is equally important for improving validation. Efforts such as refining the digital dynamic model to better represent vehicle-specific characteristics and integrating more realistic environmental interactions can further narrow the sim-to-real gap. However, simulation-only V&V faces inherent limitations. These challenges highlight the need for hardware-in-the-loop experiments and outdoor testing to identify failure modes that simulations may overlook.

Vision-based Refinement. The changes in reachability in scenarios 4 and 5 (Fig. 11 and 12), as compared to scenarios 2 and 3 (Fig. 9 and 10), motivate the inclusion of vision-based refinement to landing target position, especially in environments that cause GPS or GNSS sensor's accuracy to degrade. While not a focus of this work, the vision-based refinement benefits from task aware learning and optimization [45].

Over-approximation in V&V Framework. A limitation of the V&V framework is that the reachability analysis is necessarily a conservative overapproximation. In this work, in Scenarios 3 and 5 (§V.B), had the Minihawk spawn logic been the same as other scenarios, the Minihawk would diverge all around the obstacle. While these trajectories are safe, the resultant reachability analysis would erroneously include the region occupied by the obstacle. This is a known limitation. The common practice to addressing this challenge is partitioning the initial condition set, where the overall reachable set is obtained as the union of the reachable set from each partition. By refining the initial conditions, we can expect the resulting reachable set to avoid obstacles, thereby ensuring safety. However, in this paper, it is not achievable since the trajectories are pre-sampled. Therefore, to present the reachability analysis for scenarios with obstacles, by modifying the spawn logic, the Minihawk is made to favor a subset of all possible trajectories around the obstacle.

VI. Conclusion and Future Work

This work presents a Verification and Validation framework for vision-based landing systems in hybrid VTOL UAVs, focusing on the MiniHawk platform. By integrating Verse tool and high-fidelity photorealistic simulation environment built on CARLA, this work evaluates the safety and reliability of an autonomous landing system in cluttered urban environments. Formal verification through reachability analysis provides mathematical guarantees for safe operation, while scenario-based validation assesses algorithm performance and robustness under various conditions. The findings highlight the effectiveness of combining high-fidelity simulation with formal verification tools in narrowing certain aspects of the sim-to-real gap and evaluating system performance. However, limitations emphasize the need for additional approaches such as hardware-in-the-loop testing and outdoor experiments, to refine and validate the algorithms.

The contribution of this study offers a foundation for advancing the reliability of autonomy algorithms in hybrid VTOL UAVs. In future work, we would like to enhance the complexity of test scenarios by introducing dynamic intruders and complex environments. We want to enable the autonomy system to perform online planning using integrated perception and path planning algorithms. Additionally, we plan to address the heterogeneity of hybrid VTOL vehicle by designing new path planning and control algorithms that account for the distinct dynamics in different flight modes. We also aim to develop an interface for direct motor control, allowing us to implement and evaluate state-of-the-art controllers. Beyond the current focus on verifying the landing procedure, we intend to validate complete sequence

including takeoff, forward flight and landing, to ensure performance across all mission phases. Furthermore, we plan to conduct real-world validation using the MiniHawk platform.

Acknowledgments

This material is based upon work supported by the National Aeronautics and Space Administration (NASA) under the cooperative agreement 80NSSC20M0229 and University Leadership Initiative grant no. 80NSSC22M0070, and the National Science Foundation (NSF) under grant no. CNS 1932529 and ECCS 2311085. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsors.

We also thank Michael Acheson from NASA for sharing the GUAM simulator [36].

References

[1] Massachusetts Institute of Technology, "Autonomous Electric VTOL Transport Aircraft," https://roadmaps.mit.edu/index.php/Autonomous_ElectricVTOL_Transport_Aircraft, n.d. Accessed: 2024-11-24.

[2] Wei, H., Lou, B., Zhang, Z., Liang, B., Wang, F.-Y., and Lv, C., "Autonomous navigation for eVTOL: Review and future perspectives," *IEEE Transactions on Intelligent Vehicles*, 2024.

[3] Feary, M. S., Kaneshige, J., Lombaerts, T., Shish, K., and Haworth, L., "Evaluation of Novel eVTOL Aircraft Automation Concepts," *AIAA Aviation Forum and Exposition*, 2023.

[4] Tepylo, N., Straubinger, A., and Laliberte, J., "Public perception of advanced aviation technologies: A review and roadmap to acceptance," *Progress in Aerospace Sciences*, Vol. 138, 2023, p. 100899.

[5] Chen, C., Wang, Z., Gong, Z., Cai, P., Zhang, C., and Li, Y., "Autonomous Navigation and Obstacle Avoidance for Small VTOL UAV in Unknown Environments," *Symmetry*, Vol. 14, No. 12, 2022, p. 2608.

[6] Ducard, G. J., and Allenspach, M., "Review of designs and flight control techniques of hybrid and convertible VTOL UAVs," *Aerospace Science and Technology*, Vol. 118, 2021, p. 107035.

[7] Bauersfeld, L., Spannagl, L., Ducard, G. J., and Onder, C. H., "MPC flight control for a tilt-rotor VTOL aircraft," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 57, No. 4, 2021, pp. 2395–2409.

[8] Sargent, R. G., "Verification and validation of simulation models," *Proceedings of the 2010 Winter Simulation Conference*, IEEE, 2010, pp. 166–183.

[9] Jakobi, N., Husbands, P., and Harvey, I., "Noise and the reality gap: The use of simulation in evolutionary robotics," *Advances in Artificial Life: Third European Conference on Artificial Life Granada, Spain, June 4–6, 1995 Proceedings 3*, Springer, 1995, pp. 704–720.

[10] Salvato, E., Fenu, G., Medvet, E., and Pellegrino, F. A., "Characterization of modeling errors affecting performances of a robotics deep reinforcement learning controller in a sim-to-real transfer," *2021 44th International Convention on Information, Communication and Electronic Technology (MIPRO)*, IEEE, 2021, pp. 1154–1159.

[11] Epic Games, "Unreal Engine 4," https://www.unrealengine.com/, 2014.

[12] Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., and Koltun, V., "CARLA: An open urban driving simulator," *Conference on Robot Learning*, PMLR, 2017, pp. 1–16.

[13] Bansal, A., Zhao, Y., Zhu, J., Cheng, S., Gu, Y., Yoon, H. J., Kim, H., Hovakimyan, N., and Sha, L. R., "Synergistic Perception and Control Simplex for Verifiable Safe Vertical Landing," *AIAA SCITECH 2024 Forum*, 2024, p. 1167.

[14] Li, Y., Zhu, H., Braught, K., Shen, K., and Mitra, S., "Verse: A python library for reasoning about multi-agent hybrid system scenarios," *International Conference on Computer Aided Verification*, Springer, 2023, pp. 351–364.

[15] Qin, T., Zhang, G., Yang, L., and He, Y., "Research on the Endurance Optimisation of Multirotor UAVs for High-Altitude Environments," *Drones*, Vol. 7, No. 7, 2023, p. 469.

[16] Keane, A. J., Sóbester, A., and Scanlan, J. P., *Small Unmanned Fixed-wing Aircraft Design: A Practical Approach*, John Wiley & Sons, 2017.

[17] Lopez, M. J., Duffy, C. S., Tischler, M. B., and Ruckel, P., "Bell V-280 Hover Flight Dynamics Model Validation and Update with Flight Test Data," *Vertical Flight Society 77th Annual Forum*, 2021.

[18] McSwain, R. G., Glaab, L. J., Theodore, C. R., Rhew, R. D., and North, D. D., "Greased Lightning (GL-10) Performance Flight Research: Flight Data Report," Tech. rep., Langley Research Center, NASA, 2017.

[19] Carlson, S. J., and Papachristos, C., "The Minihawk-VTOL: Design, Modeling, and Experiments of a Rapidly-prototyped Tiltrotor UAV," *2021 International Conference on Unmanned Aircraft Systems (ICUAS)*, IEEE, 2021, pp. 777–786.

[20] Carlson, S., "MiniHawk-VTOL," https://github.com/StephenCarlson/MiniHawk-VTOL, 2022.

[21] RoboWork, "Aerial Robotics," https://github.com/robowork/aerial_robotics, 2023. Accessed: 2024-11-18.

[22] Cheng, H., Lin, L., Zheng, Z., Guan, Y., and Liu, Z., "An Autonomous Vision-Based Target Tracking System for Rotorcraft Unmanned Aerial Vehicles," *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2017, pp. 1732–1738.

[23] Wu, S., Li, R., Shi, Y., and Liu, Q., "Vision-Based Target Detection and Tracking System for a Quadcopter," *IEEE Access*, Vol. 9, 2021, pp. 62043–62054.

[24] Kong, W., Zhang, D., and Zhang, J., "A Ground-Based Multi-Sensor System for Autonomous Landing of a Fixed Wing UAV," *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, IEEE, 2015, pp. 1303–1310.

[25] Tian, P., Chao, H., Rhudy, M., Gross, J., and Wu, H., "Wind Sensing and Estimation Using Small Fixed-Wing Unmanned Aerial vehicles: A survey," *Journal of Aerospace Information Systems*, Vol. 18, No. 3, 2021, pp. 132–143.

[26] Cai, Y., Xi, Q., Xing, X., Gui, H., and Liu, Q., "Path Planning for UAV Tracking Target Based on Improved A-star Algorithm," *2019 1st International Conference on Industrial Artificial Intelligence (IAI)*, IEEE, 2019, pp. 1–6.

[27] Zhou, Q., and Liu, G., "UAV Path Planning Based on the Combination of A-star Algorithm and RRT-star Algorithm," *2022 IEEE International Conference on Unmanned Systems (ICUS)*, IEEE, 2022, pp. 146–151.

[28] ArduPilot Development Team, "ArduPilot: Open Source Autopilot," , 2024. URL http://www.ardupilot.org/, accessed: 2024-11-29.

[29] Redfield, S. A., and Seto, M. L., "Verification Challenges for Autonomous Systems," *Autonomy and Artificial Intelligence: A Threat or Savior?*, 2017, pp. 103–127.

[30] Bak, S., and Duggirala, P. S., "Hylaa: A Tool for Computing Simulation-Equivalent Reachability for Linear Systems," *Proceedings of the 20th International Conference on Hybrid Systems: Computation and Control*, ACM, 2017, pp. 173–178.

[31] Fan, C., Qi, B., Mitra, S., and Viswanathan, M., "DryVR: Data-Driven Verification and Compositional Reasoning for Automotive Systems," *International Conference on Computer Aided Verification*, Springer, 2017, pp. 441–461.

[32] Althoff, M., "An Introduction to CORA 2015," *Proc. of the Workshop on Applied Verification for Continuous and Hybrid Systems*, 2015, pp. 120–151.

[33] Li, Y., Yang, B. C., Jia, Y., Zhuang, D., and Mitra, S., "Refining Perception Contracts: Case Studies in Vision-Based Safe Auto-Landing," *arXiv preprint arXiv:2311.08652*, 2023.

[34] Song, L., Cheng, S., Mitra, S., and Hovakimyan, N., "Verification of Design Specifications in $\mathcal{L}_1$ Adaptive Control," *AIAA SCITECH 2024 Forum*, 2024, p. 1165.

[35] Chance, G., Ghobrial, A., McAreavey, K., Lemaignan, S., Pipe, T., and Eder, K., "On Determinism of Game Engines Used for Simulation-based Autonomous Vehicle Verification," *IEEE Transactions on Intelligent Transportation Systems*, Vol. 23, No. 11, 2022, pp. 20538–20552.

[36] Acheson, M. J., Cook, J. W., Simmons, B. M., Derry, S., Bacon, B., and Britton, T., "Generic-Urban-Air-Mobility-GUAM," , June 2024. URL https://github.com/nasa/Generic-Urban-Air-Mobility-GUAM.

[37] Tahir, Z., and Alexander, R., "Intersection focused situation coverage-based verification and validation framework for autonomous vehicles implemented in Carla," *International Conference on Modelling and Simulation for Autonomous Systems*, Springer, 2021, pp. 191–212.

[38] Won, M., and Kim, S., "Verification and Validation Utilizing Carla Simulator for Autonomous Driving Development," *International Conference on Simulation and Modeling Methodologies, Technologies and Applications*, Springer, 2022, pp. 66–85.

[39] Zhang, Y., Carballo, A., Yang, H., and Takeda, K., "Perception and Sensing for Autonomous Vehicles Under Adverse Weather Conditions: A Survey," *ISPRS Journal of Photogrammetry and Remote Sensing*, Vol. 196, 2023, pp. 146–177.

[40] Chen, L., Wu, P., Chitta, K., Jaeger, B., Geiger, A., and Li, H., "End-to-End Autonomous Driving: Challenges and Frontiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[41] Goyal, S., Griggio, A., and Tonetta, S., "System-level Simulation-based Verification of Autonomous Driving Systems with the VIVAS Framework and CARLA Simulator," *Science of Computer Programming*, 2024, p. 103253.

[42] Koenig, N., and Howard, A., "Design and Use Paradigms for Gazebo, An Open-Source Multi-Robot Simulator," *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, Vol. 3, 2004, pp. 2149–2154 vol.3. https://doi.org/10.1109/IROS.2004.1389727.

[43] Open Robotics, "ROS Noetic," https://wiki.ros.org/noetic, 2020.

[44] Redmon, J., and Farhadi, A., "YOLO9000: Better, Faster, Stronger," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7263–7271.

[45] Rasul, A. E., Tasnim, H., Yoon, H.-J., Bansal, A., Wang, D., Hovakimyan, N., Sha, L., and Voulgaris, P., "Bayesian Data Augmentation and Training for Perception DNN in Autonomous Aerial Vehicles," *AIAA SciTech 2025 Forum*, 2025.

[46] Bitoun, J., and Winkler, S., "HelipadCat: Categorised Helipad Image Dataset and Detection Method," *2020 IEEE REGION 10 CONFERENCE (TENCON)*, IEEE, 2020, pp. 685–689.

[47] Ultralytics, "YOLOv8: You Only Look Once version 8," https://github.com/ultralytics/ultralytics, 2023.

[48] Masci, F., "Method of detecting moving objects," , Oct. 13 2022. US Patent App. 17/616,326.

[49] Bansal, A., Kim, H., Yu, S., Li, B., Hovakimyan, N., Caccamo, M., and Sha, L., "Verifiable Obstacle Detection," *2022 IEEE 33rd International Symposium on Software Reliability Engineering (ISSRE)*, IEEE, 2022, pp. 61–72.

[50] Shaikh, M. Y., Petrunin, I., and Zolotas, A., "Self-supervised Obstacle Detection during Autonomous UAS Taxi Operations," *AIAA SCITECH 2023 Forum*, 2023, p. 2672.

[51] Bansal, A., Kim, H., Yu, S., Li, B., Hovakimyan, N., Caccamo, M., and Sha, L., "Perception simplex: Verifiable collision avoidance in autonomous vehicles amidst obstacle detection faults," *Software Testing, Verification and Reliability*, 2024, p. e1879.

[52] Hart, P. E., Nilsson, N. J., and Raphael, B., "A Formal Basis for the Heuristic Determination of Minimum Cost Paths," *IEEE transactions on Systems Science and Cybernetics*, Vol. 4, No. 2, 1968, pp. 100–107.

[53] LaValle, S., "Planning Algorithms," *Cambridge University Press*, Vol. 2, 2006, pp. 3671–3678.

[54] Cao, C., and Hovakimyan, N., "Design and Analysis of a Novel $\mathcal{L}_1$ Adaptive Control Architecture With Guaranteed Transient Performance," *IEEE Transactions on Automatic Control*, Vol. 53, No. 2, 2008, pp. 586–591.

[55] Cui, Y., and Ge, S. S., "Autonomous Vehicle Positioning With GPS in Urban Canyon Environments," *IEEE Transactions on Robotics and Automation*, Vol. 19, No. 1, 2003, pp. 15–25.

[56] Kos, T., Markezic, I., and Pokrajcic, J., "Effects of Multipath Reception on GPS Positioning Performance," *Proceedings ELMAR-2010*, IEEE, 2010, pp. 399–402.