Robotics: Science and Systems 2025
Los Angeles, CA, USA, June 21–25, 2025

Demonstrating ViSafe: Vision-enabled Safety for High-speed Detect and Avoid

theairlab.org/visafe/

Parv Kapoor*, Ian Higgins*, Nikhil Keetha*, Jay Patrikar*, Brady Moon, Zelin Ye, Yao He, Ivan Cisneros, Yaoyu Hu, Changliu Liu, Eunsuk Kang, Sebastian Scherer

Carnegie Mellon University

*Equal Contribution

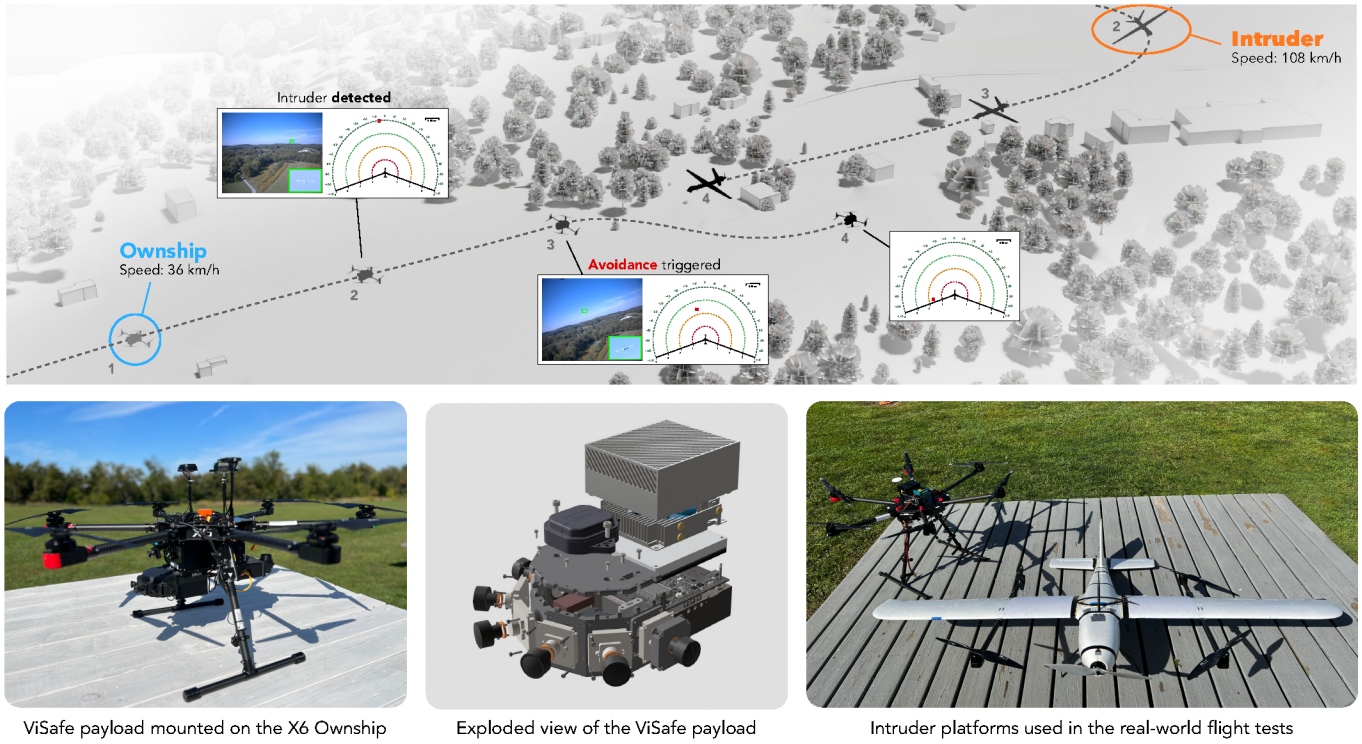

Fig. 1. **We demonstrate *ViSafe*, a high-speed vision-only airborne collision avoidance system.** *Top:* Rendering of a real-world flight test log where the *ViSafe* system detects an incoming intruder with a **144 km/h closure rate** and performs an avoidance maneuver to ensure safe separation. The annotations showcase the detections and multi-camera tracks from the vision-based aircraft detection and tracking system, while the trajectory shows the log of the performed real-world avoidance maneuver. The numbered annotations showcase the different stages of the flight test, from intruder detection to avoidance completion. *Bottom:* ViSafe payload, ownship, and intruder platforms used in the real-world flight tests.

***Abstract*—Assured safe-separation is essential for achieving seamless high-density operation of airborne vehicles in a shared airspace. To equip resource-constrained aerial systems with this safety-critical capability, we present *ViSafe*, a high-speed vision-only airborne collision avoidance system. *ViSafe* offers a full-stack solution to the Detect and Avoid (DAA) problem by tightly integrating a learning-based edge-AI framework with a custom multi-camera hardware prototype designed under SWaP-C constraints. By leveraging perceptual input-focused control barrier functions (CBF) to design, encode, and enforce safety thresholds, *ViSafe* can provide provably safe runtime guarantees for self-separation in high-speed aerial operations. We evaluate *ViSafe*'s performance through an extensive test campaign involving both simulated digital twins and real-world flight scenarios. By independently varying agent types, closure rates, interaction geometries, and environmental conditions (e.g., weather and lighting), we demonstrate that *ViSafe* consistently ensures self-separation across diverse scenarios. In first-of-its-kind real-world high-speed collision avoidance tests with closure rates reaching *144 km/h*, *ViSafe* sets a new benchmark for vision-only autonomous collision avoidance, establishing a new standard for safety in high-speed aerial navigation.**

I. Introduction

Collision avoidance systems are critical to enabling safe operations in shared airspace. The integration of Uncrewed Aerial Systems (UASs) into an already congested National Airspace System raises pressing concerns about ensuring the safe separation of airborne vehicles. Existing solutions, such as Autonomous Collision Avoidance Systems (ACAS) [33] and Unmanned Traffic Management (UTM) [18] frameworks,

have demonstrated effectiveness. However, these systems often depend on multiple active sensor modalities—such as transponders, radars, and ADS-B—which are unsuitable for small UASs due to stringent size, weight, power, and cost (SWaP-C) constraints. Addressing threats posed by cooperative as well as non-cooperative aerial entities like balloons and rogue drones while respecting SWaP-C resource constraints remains an ongoing challenge. As a result, current regulations impose strict line-of-sight requirements on human operators, significantly restricting the utility and scalability of UASs.

Cameras offer a lightweight, cost-effective alternative for enabling safety-critical Detect and Avoid (DAA) capabilities in sUAS. With the growth of data-driven methods, vision-based object detection offers a promising direction for tracking small aircraft in images with low signal-to-noise ratios [17]. However, real-world deployment and integration of these detection systems with downstream provably safe collision avoidance methods remain challenging.

We present *ViSafe*, a vision-only airborne collision avoidance system to impart see-and-avoid capabilities to sUAS. *ViSafe* builds on our prior work AirTrack [17], which uses high-resolution detection and tracking networks to detect aerial objects. We extend AirTrack to the multi-camera setting by using multi-view fusion to track detected intruder positions across multiple cameras. Additionally, we formulate the downstream vision-based collision avoidance problem from a control theoretic perspective using Control Barrier Functions (CBFs) [7], which provide provable guarantees of safety and are suited for runtime monitoring and response.

CBFs have been successfully applied in various domains, including safe navigation [19, 6], robotic manipulation [55, 39], and industrial automation systems [7]. Their deployment for safe collision avoidance in airspaces has also been investigated [16, 38, 42]. While existing work assumes global availability of information and uses proprioceptive information, *ViSafe* removes this assumption in the formulation. We identify key challenges in deploying CBFs in the wild and provide insights for deploying these techniques for high-speed collision avoidance. Moreover, we offer an edge compute-focused solution for real-time deployment on resource-constrained platforms.

Overall, we adopt a "requirement refinement" approach for designing our formulations, where we leverage the empirical performance statistics of our vision inference system (AirTrack [17]) to account for uncertainty in state estimation. This is achieved using a multi-view Kalman Filter and the ASTM F3442/F3442M standard (satisfied by AirTrack) to determine the range profile in which intruder detections are reliable.

Bridging the gap between theoretical formulations and real-world safe behavior requires extensive testing. To address this, we propose a "digital twin" of the system and the operating environment in simulation. Using Nvidia Isaac Sim [5], we render realistic camera feeds for use in our visual detection algorithm, providing large-scale and diverse scenario benchmarking abilities. These tests were run using the same hardware as the real-world payload, thereby minimizing our sim-to-real gap for testing.

Finally, after extensive experimentation in simulation, we test our avoidance algorithms at multiple real-world testing facilities. We created representative configurations in the field to analyze our system's collision avoidance capabilities. We conducted approximately 80 hours of flight testing across two outdoor testing locations to validate our hypotheses. We observe similar collision avoidance performance in both simulation and field testing, which further lends credibility to our hardware-in-the-loop simulation fidelity.

The main contributions of this work are as follows:

1) Multi-view vision-only aircraft detection & tracking and CBF-based collision avoidance system that assumes no global availability of information or communication.
2) Custom-built SWaP-C hardware that simultaneously streams multiple camera inputs, provides state estimation, performs deep learning model edge inference, and computes avoidance maneuvers on board in real time.
3) Digital twin and hardware-in-the-loop simulation to perform DAA benchmarking and performance analysis under different agent types, closure rates, interaction geometries, and environmental conditions.
4) First-of-its-kind real-world flight tests demonstrating that *ViSafe* ensures safe aerial separation in encounter scenarios with a closure rate of up to 144 km/h.

II. Related Work

A. Collision Avoidance logics

One of the seminal works in airborne collision avoidance was the Traffic Collision and Avoidance System (TCAS) [27], which functions on a cooperative surveillance mechanism. The agents communicate their position and intent using Mode S transponders, which must be installed on every aircraft. More recently, the ACAS family of avoidance algorithms has been developed. There are variants of this algorithm for different agent types in different airspaces (ACAS Xa, Xu), etc. The key factor driving the development of ACAS algorithms is the availability of extended surveillance data using ADS-B, which enables aircraft to assess potential collision risks and coordinate maneuvers collaboratively. These logics involve generating cost tables for agent states and possible actions through simulation and optimization [8]. These tables are then used for Resolution Advisories (RAs) during deployment.

However, most existing avoidance logics require special sensors and information to provide RAs. Deploying these sensors on small UAS is challenging, and there is a large push toward sensor miniaturization for this purpose. The investigation of ACASXu with visual inference information is a relatively unexplored area [41]. Additionally, most avoidance logics are designed for large UAS, and hence, resolution advisories tend to be relatively simple. However, small UAS can be in dense airspaces with other UAS where a finer discretization of action space is needed. This can lead to state space explosion, where cost tables can be prohibitively large.

Our proposed technique works with vision inference data and does not depend on radar and ADS-B data. Additionally,

our RAs are generated at runtime using efficient mixed integer linear programming (MILP) solvers, which are suited for more fine-grained control of UAS. Moreover, our technique can factor in nominal control inputs, which incorporate the liveness requirement of reaching a goal. It helps our agent stray not too far from the originally planned trajectory.

B. Control Barrier Functions for Aerial Collision Avoidance

There has been a recent surge in using CBFs to ensure the safe separation of UAS. Squires *et al.*[51] identify key challenges with designing CBFs for collision avoidance and propose a construction technique. However, they assume information availability at all times during an episode and test their formulations in simulation without sensor noise. Likewise, follow-up work [52] investigates the deployment of CBFs in a decentralized setup with message passing to communicate control outputs. However, they do not consider non-cooperative setups in which all agents do not provide information. Our proposed technique assumes a non-cooperative setup in which no information about other agents is explicitly communicated to the controlled agent. Moreover, we deal with information unavailability, where the CBF is only active when intruder information from our vision-based detection and tracking module is available.

C. Aircraft Detection and Tracking

Historically, traditional vision-based aircraft detection and tracking systems have employed modular approaches incorporating established computer vision techniques such as frame stabilization, background-foreground separation, and flow-based object tracking [30, 13, 15, 11, 29, 36]. In these modular approaches, frame stabilization was achieved using optical flow or image registration methods [35, 45, 48], while regression-based motion compensation and morphological operations were utilized to highlight potential intruders [46]. Background subtraction and traditional machine learning algorithms, including SVMs, were also applied to learn descriptors to identify aircraft [46, 13, 44]. Furthermore, temporal filtering was used to reduce the false positives from these morphological operations. Lastly, track-before-detect methods have also been explored, where the tracking is usually managed using methods like Hidden Markov Models, Kalman filters, and Viterbi-based filtering [40, 30, 28, 37]. While our visual detection and tracking system, AirTrack [17], uses deep learning-based modules, the overall sequential pipeline of performing frame alignment, detection, tracking, and false positive filtering is strongly motivated by the success of past modular & traditional approaches.

Recent advancements have leveraged deep learning, particularly convolutional neural networks (CNNs), for object detection [12, 22, 53, 24]. In particular, for aircraft detection, the challenge is detecting small objects within high-resolution images, where keypoint-based architectures prove more effective than traditional anchor-based methods like YOLO and R-CNN [14]. Given the data-centric nature of these methods, this has led to a proliferation of aircraft detection datasets [1, 50, 49, 54, 43]. This insight aligns with findings in related fields such as face detection and object detection in aerial imagery [21, 26, 31]. For vision-based aircraft detection systems where computational efficiency is crucial, fully convolutional networks that predict heatmaps are the most effective [24, 25]. Furthermore, the tracking-by-detection framework is standard for aircraft tracking, using a management system to handle track initiation and termination and associating new detections with existing tracks using algorithms like the Hungarian method [9, 30, 13]. For small objects, where bounding box IoU is unreliable due to sensitivity to positional deviations, integrated detection and tracking approaches offer a superior alternative [57]. Motivated by this trend, our AirTrack [17] employs an anchor-free detection and tracking system.

III. Problem Definition

We model the problem of maintaining safe separation between two aerial agents: the ownship agent and the intruder agent. The ownship refers to the primary controlled agent and the intruder refers to a potentially unknown agent in the configuration space which doesn't have any communication with the ownship agent. Only passive sensing, specifically vision, is used to detect the intruder. Safe separation entails keeping the two agents outside each other's collision volume at all times. We model the collision avoidance problem in the horizontal plane and assume constant velocity vectors for the intruder agent inspired by [33]. The separation requirement can be formally defined as:

$$\forall t \in T_{operation} \bullet d_{ownship-intruder} \geq d_{thresh} \tag{1}$$

where d_{thresh} is the minimum distance threshold and $d_{ownship-intruder}$ is the relative horizontal distance between the agents. For notational simplicity, we will refer to this distance as d in our formulations. Additionally, $T_{operation}$ is the time of operation of the ownship agent, and Eq. (1) formally states the problem of safe separation for the entire operation time.

As can be inferred from (1), this is equivalent to a forward invariance property. In order to encode and enforce this property, we make certain assumptions. First, we assume a simple unicycle dynamics model for our ownship agent:

$$\dot{x_{own}} = \begin{bmatrix} v_{own}\cos(\chi_{own}) \\ v_{own}\sin(\chi_{own}) \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} u \tag{2}$$

$$u^T = [\dot{v}_{own} \;\; \dot{\chi}_{own}] \tag{3}$$

Here, v_{own} denotes the speed of the ownship, and χ_{own} is the heading (yaw) angle. The control input $u \in \mathbb{R}^2$ consists of the rate of change of speed and heading, i.e., $\dot{v}_{\text{own}}$ and $\dot{\chi}_{\text{own}}$. Additionally, we also consider control constraints to closely mimick real world deployment scenarios.

$$u \geq u_{min}, u \leq u_{max} \;\; \forall t \in T_{operation}$$

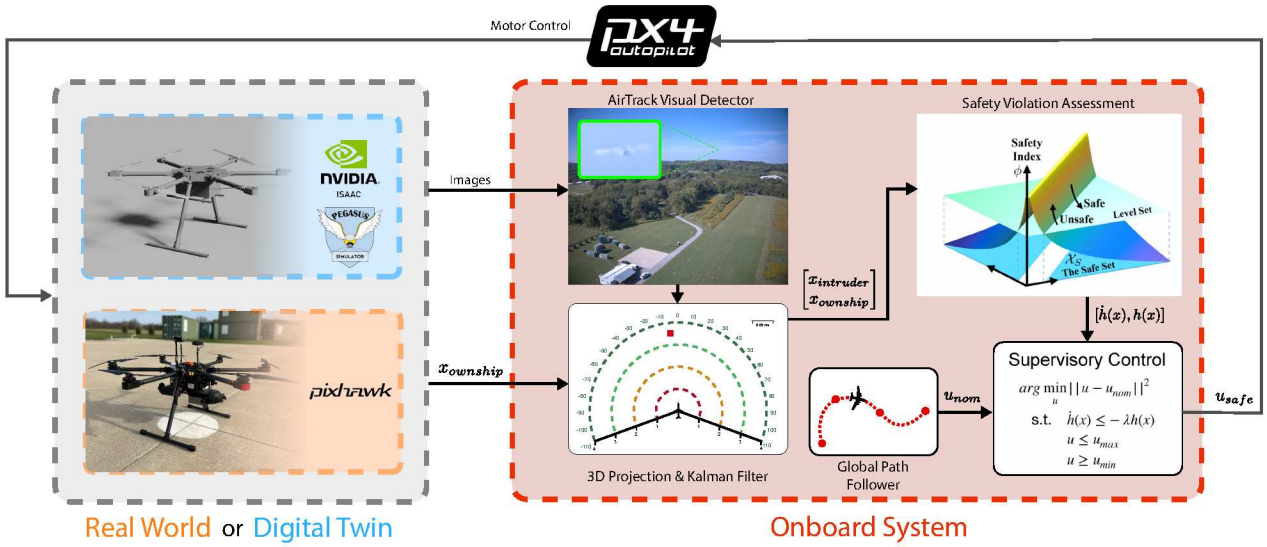

Fig. 2. **Overview of our *ViSafe* framework for real-world testing and hardware-in-the-loop simulation:** Firstly, the onboard sensors or digital twin simulation stream the multi-cam videos to the AirTrack visual detection module, which detects the intruder across multiple views. Then, these detections, along with the ownship state information, are sent to the multi-view fusion and coordinate frame conversion module, which then tracks the intruder and sends the intruder state information along with the ownship state information to the CBF. The CBF uses the nominal global plan and the safety violation assessment to compute modifications to the nominal control input in case of violation. This safe control output is then sent to the drone autopilot system for execution. This loop continues to operate in real-time until the flight test is complete.

These control constraints are particularly important for our use case since, for some aircraft, loiter safety maneuvers are actually impossible and would lead to an aircraft crash.

Lastly, we define the dynamics of the intruder agent:

$$\dot{x_{int}} = \begin{bmatrix} v_{int} \cos(\chi_{int}) \\ v_{int} \sin(\chi_{int}) \end{bmatrix} \tag{4}$$

Here, v_{int} denotes the speed of the intruder, and χ_{int} is the heading (yaw) angle. Overall, we make two assumptions: First, the intruder has constant velocity vectors as discussed earlier. Second, the intruder agent is agnostic to the ownship agent's maneuvers, i.e., there are no interaction effects. This assumption also implies that the township agent possesses no collision avoidance algorithm, since that would lead to interaction effects. These assumptions are made to keep the problem tractable but remain practical in today's airspace.

IV. *ViSafe* Framework

An overview of our *ViSafe* framework (Fig. 2) is as follows: While the visual detection module (Section IV-A) provides an image-level intruder detection, the intruder state information is transformed to the North East Down (NED) coordinate system and broadcasted to the safety controller in its expected input format (Section IV-B). The safety controller (Section IV-C) encodes the desired safe separation constraints as well as our actuation constraints and computes modifications to nominal control input in case of violation. These constraints are derived using our defined CBF. This modified control action is then converted into low-level drone commands and executed.

A. Visual Detection Module

We extend the state-of-the-art aircraft detection & tracking model, AirTrack [17], to detect intruders. Specifically, we modified the AirTrack algorithm (originally focused on single-camera inference) to incorporate multi-camera inputs and enable multi-camera tracking. Furthermore, we also upgraded the SWaP-C hardware, implementing efficient multi-camera image sharing (with zero memory copies) and further optimizing the deep learning model inference to operate at the desired frequency required to facilitate high-speed collision avoidance on the edge. The overall design of our Detection Module comprises Frame Alignment, Detection, Secondary Classification, and Image-level Tracking. The model's inputs are two consecutive grayscale image frames, and it outputs a list of tracked objects with various attributes. The model is trained on the Amazon Airborne Object Tracking (AOT) dataset [1], and its modules are as follows:

Frame Alignment: The goal of this module is to align the successive frames in a video so that the ego-camera motion can be discarded, thereby helping to distinguish foreground objects from the background. This is achieved by predicting the optical flow between two successive input image frames and the confidence of the predicted flow. The input images are cropped from the center-bottom to cover high-texture details below the horizon, and a ResNet-34 backbone architecture with two prediction heads is used for alignment. The prediction is made at $1/32$ scale of the input, and low confidence offset predictions are rejected.

Detection: This module is split into two parts: a primary detection module and a secondary detection module. The primary module takes in two full-resolution (1224×1024)

grayscale image frames that have been aligned, while the secondary module receives a smaller $\approx 1/5$th sized crop of the image around the top k detector outputs (based on confidence) from the primary module. The network is fully-convolutional (HR-Net-W32) and it produces five outputs: a center heatmap indicating the object's center, bounding box size, center offset from the grid to the object's center, track offset of the object's center from the prior frame, and object distance in log scale.

Secondary Classifier: A ResNet-18 module is used as a binary classifier for false-positive rejection. The module takes cropped regions around the bounding boxes from the detection module as input and predicts whether the crop is an aircraft or not. This improves the overall precision.

Image-level Tracking: We use an offset tracking vector-based approach [57]. The predicted track offset vector from the detection module is subtracted from the current object center to find the location in the prior frame. If the location matches existing tracks within a threshold, the detection is associated with an image-level track; otherwise, a new track is initialized.

Overall, the sequential modules of AirTrack work together to provide precise image-level aircraft tracks and distance from the ownship. For *ViSafe*, we run AirTrack on video streams from two cameras at 8 Hz on our Nvidia Orin AGX compute system (described in Section V-E). We encourage the readers to refer to the AirTrack [17] paper for further details.

B. Multi View Fusion & Coordinate Frame Conversion

Once AirTrack provides an image-level intruder track (from any one of the multiple cameras), we unproject the track location in the image to 3D leveraging the predicted distance of the aircraft, center of the predicted bounding box and the calibrated intrinsics of the respective camera. Then, the 3D position of the image-level track in the camera coordinate frame is transformed to the local frame of the 3DM-GQ7 GNS/INSS module by using the multi-camera and IMU extrinsic calibration. Since, the GNSS module provides us the heading of the ownship with respect to the North direction, we can transform the 3D image-level intruder track position to a North-East-Down (NED) coordinate frame centered about the GNSS module of the ownship. Now given the 3D position $[x_t, y_t, z_t]$ of the image-level track, the range d, azimuth θ and elevation ϕ of the intruder in the ownship's NED coordinate frame can be calculated as:

$$\begin{aligned} d &= \sqrt{{x_t}^2 + {y_t}^2 + {z_t}^2} \\ \theta &= \arctan\left(\frac{y_t}{x_t}\right) \\ \phi &= -\arcsin\left(\frac{z_t}{d}\right) \end{aligned} \tag{5}$$

We leverage a multi-view fusion module to track and fuse the intruder positions across multiple cameras. At the start, for the first image-level intruder track, we initialize two Kalman Filters to perform tracking and fusion of new observations for the same intruder. We empirically observe that the range (d) is noisier than the angles (θ & ϕ) since the angular estimates depend solely on the 2D location of the detection

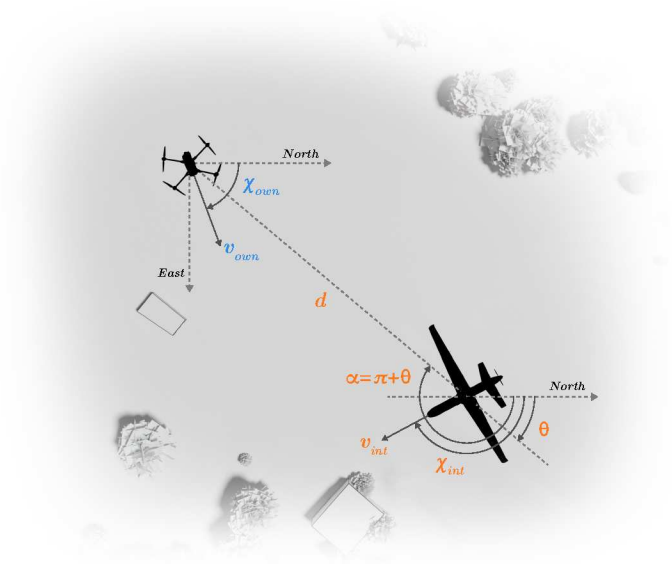

Fig. 3. **Encounter geometry and information required for vision-enabled collision avoidance.** The velocity of the ownship v_{own} and heading with respect to North χ_{own} are obtained from the ownship odometry. The intruder's range d, azimuth θ, velocity v_{int}, and heading with respect to North χ_{int} are obtained using the visual detection module.

while the range is predicted by the monocular deep learning model (AirTrack). It is possible to use a single Kalman Filter by specifying independent process noises; however, we find that the coupling of range measurements degrades the angular estimates, requiring significant joint tuning. Hence, we utilize two Kalman Filters, each with a constant velocity motion model (one for angles and one for range). Through decoupling, the range filter accounts for high uncertainty in range while the angle filter ensures smooth tracking. Subsequently, for each new image-level track across the different cameras, we use the angular distance to determine if the track associates with pre-existing 3D intruder tracks. If it associates, we fuse the new observation with the two pre-existing Kalman Filters. Otherwise, we initiate another new 3D intruder track with two Kalman Filters. In this way, we can use Kalman Filters (unique to each intruder) to initialize, associate, and track different intruders in 3D. Each 3D intruder track is configured to propagate the intruder position and velocity to the downstream supervisory safety controller at a frequency of 24 Hz.

As shown in Fig. 3, the supervisory safety controller described in the subsequent sub-section expects the multi-view fusion module to output the range d, the angle of the line of sight to the NED frame centered about the intruder α, the velocity of the intruder v_{int} along the North and East direction and the heading of the intruder with respect to the North direction χ_{int}. The tracks from the multi-view fusion module already provide the range d, and α can be computed as $\pi + \theta$. Furthermore, using the ownship's odometry information (v_{own} & χ_{own}) and the track information (range d, range rate $\dot{d}$, azimuth θ and azimuth rate $\dot{\theta}$), we can compute the two

components of v_{int} as follows:

$$\begin{aligned} v_{intNorth} &= v_{ownNorth} + \dot{d}\cos\theta + d\dot{\theta}\sin\theta \\ v_{intEast} &= v_{ownEast} + \dot{d}\sin\theta + d\dot{\theta}\cos\theta \end{aligned} \tag{6}$$

Subsequently, the heading of the intruder with respect to North can be obtained as:

$$\chi_{int} = \arctan\left(\frac{v_{intEast}}{v_{intNorth}}\right) \tag{7}$$

Hence, in this way, we transform & convert the image-level intruder detections to the expected input format of the supervisory safety controller.

C. Supervisory Safety Controller

The three key requirements for this controller are:

- **Certified safety**: Enforcement of a strict notion of safety while working with vision inputs
- **Run time safety**: Reactive low latency solution that can make decisions in real-time
- **Performant safety**: Preserving original mission requirements and introducing a minimal modification to the original control input provided by a nominal controller

Existing techniques, such as reachability analysis and regret minimization for MPC, often conflict with these three requirements. For example, reachability analysis-based techniques are certified and can generate controllers that preserve mission outputs. However, they tend to be computationally expensive, compromising the reactive requirement. Considering these three key requirements, we used a Control Barrier Function (CBF) based Quadratic Program (QP) for our supervisory safety controller. CBFs were introduced in [7] to formally certify the safety of robotic control systems. We assume we are given a nonlinear control affine system defined by

$$\dot{x} = f(x) + g(x)u \tag{8}$$

where x $\in \mathscr{X}$, f and g are locally Lipschitz continuous functions and u $\in \mathbb{R}$ is the set of inputs to the system.

Let us say our safe set is defined by $\mathscr{X}_s \subset \mathscr{X}$, then the goal of safe control is to keep this set forward invariant with respect to the dynamics Eq. (8). Mathematically, this forward invariance of the set is described as

$$\forall x(t_0) \in \mathscr{X}_s \bullet x(t) \in \mathscr{X}_s \quad t \geq t_0 \tag{9}$$

Let $\mathscr{C} := \{x \in \mathscr{X} | h(x) \leq 0\}$ be a zero sub-level set of a function $h : X \to \mathbb{R}$. Then h is a valid control barrier function for $\mathscr{C}$ given dynamics Eq. (8) if there exists an extended class $\mathscr{K}_\infty$ function $\alpha : \mathbb{R} \to \mathbb{R}$ $(\alpha(0) = 0)$ such that:

$$L_f h(x) + L_g h(x)u \leq -\alpha(h(x) \quad \forall x \in \mathscr{C} \tag{10}$$

Here, $L_f h(x)$ and $L_g h(x)$ are the Lie derivatives of h along the drift $f(x)$ and control $g(x)$ dynamics, respectively. Note this formulation is slightly different than the one defined in [7] since we define the safe set to be a zero-level subset instead of a superset. However, since we change the inequality sign, we can still ensure forward invariance using this condition. This design choice is inspired by [32].

Next, we outline our CBF formulation and its associated QP-based controller. Our supervisory controller enforces our safety and actuation constraints. We devise this controller using our defined control barrier function. First, let our safe set be defined as C, then a straightforward distance maximization-based CBF would be:

$$h(x) = d_{thresh} - d \tag{11}$$

where

$$\begin{aligned} x \in C : h(x) \leq 0 \\ x \notin C : h(x) > 0 \end{aligned}$$

Here, d represents the distance between the agents, and d_{thresh} represents the minimum distance threshold that should be maintained between the two agents. However, since the relative degree of pure distance maximization-based CBF with respect to u is 2, there would be singularity with the CBF proposed in Eq. (11). Inspired by [32], we propose the following CBF:

$$h(x) = c + d_{thresh}^n - d^n - k\dot{d} \tag{12}$$

where $c > 0, n > 0$ and $k > 0$ are hyperparameters to be tuned.

Our defined CBF takes negative values when the agents are safely separated and positive values when the requirement is violated. The CBF condition we enforce for forward invariance is as follows:

$$\dot{h}(x) \leq -\lambda h(x)$$

We choose the penalty term as $h(x)$ itself as is often done for CBFs. λ is another hyperparameter that controls the rate of change of the barrier function. Note that the non-linear constraint $h(x) \leq 0$ is not necessarily a subset of $d \geq d_{thresh}$ when $\dot{d} > 0$. However, as demonstrated in [32], the system remains forward invariant within the set $h(x) \leq 0 \cap d \geq d_{thresh}$ under the application of the control law $\dot{h}(x) \leq -\lambda h(x)$. Our final QP formulation is defined as follows:

$$\begin{aligned} \arg\min_{u} \quad & ||u - u_{nom}||^2 \\ \text{s.t.} \quad & \dot{h}(x) \leq -\lambda h(x) \\ & u \leq u_{max} \\ & u \geq u_{min} \end{aligned} \tag{13}$$

where u_{nom} is the control input provided by the nominal controller and u_{max} and u_{min} encode the actuation constraints.

Now, to compute the constraints enforced for the QP formulation, we can concretize the various quantities in the formulation using post-processed visual detection information and ownership odometry information (v_{own} & χ_{own}). First, we derive the values of $\dot{h}$ in terms of our control input u using our encounter geometry as illustrated in Fig. 3.

$$\dot{h}(x) = nd^{n-1}\dot{d} - k\ddot{d} \tag{14}$$

where

$$\dot{d} = v_{own}\cos(\alpha - \chi_{own}) + v_{int}\cos(\alpha - \chi_{int}) \tag{15}$$

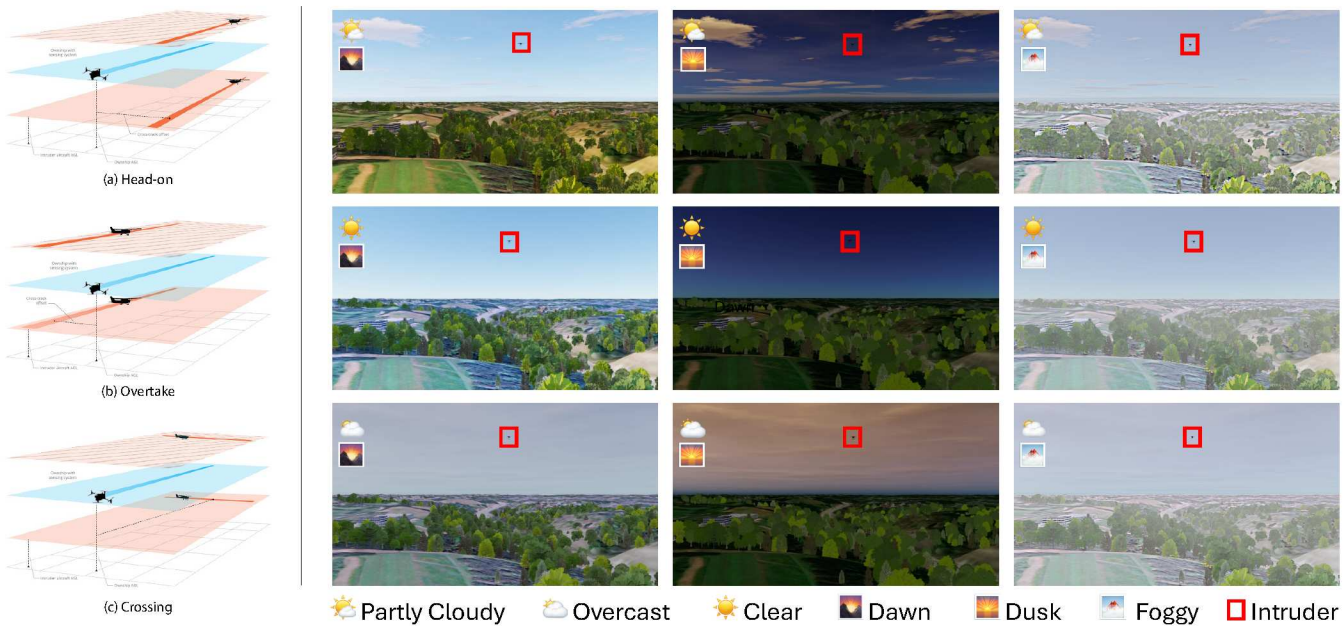

Fig. 4. **Diversity of the airborne collision testing scenarios:** (a) The various encounter geometries used for real-world & simulation flight testing. (b) The diverse weather and lighting conditions that were used to evaluate *ViSafe*'s robustness in simulation. In the simulation, based on the chosen encounter geometry, the intruder position is sampled randomly on the flying corridor.

We compute $\dot{d}$ by taking the projection of v_{own} and v_{int} along the line of sight (LOS). To compute $\ddot{d}$, we differentiate Eq. (15) with our original assumption of constant intruder velocity vector (i.e. $\dot{v}_{int} = 0$ and $\dot{\chi}_{int} = 0$) to get:

$$\ddot{d} = \dot{v}_{own}\cos(\alpha - \chi_{own}) + \dot{\chi}_{own} v_{own}\sin(\alpha - \chi_{own}) + \frac{(v_{own}\sin(\alpha - \chi_{own}) + v_{int}\sin(\alpha - \chi_{int}))^2}{d} \tag{16}$$

Now given $u = \begin{bmatrix} \dot{v}_{own} \\ \dot{\chi}_{own} \end{bmatrix}$, we can rewrite Eq. (16) as

$$\ddot{d} = \begin{bmatrix} \cos(\alpha - \chi_{own}) \\ v_{own}\sin(\alpha - \chi_{own}) \end{bmatrix}^{\top} u + \frac{(v_{own}\sin(\alpha - \chi_{own}) + v_{int}\sin(\alpha - \chi_{int}))^2}{d} \tag{17}$$

As can be observed, the constraints on h are affine in u and fit the form of a QP, as defined in Eq. (13), which can be solved to compute the desired safe control u_{safe}. We use the following tuned hyperparameters for the final formulation of Eq. (12) and 13: k = 0.2, c = 0.01, n = 0.3, and λ = 0.2.

We use a simple PD controller as our nominal controller, where the computed desired safe control u_{safe} is then converted into low-level drone control actions in the form of velocity and yaw setpoints, which are executed on our ownship agent by the flight controller (PX4 in Isaac Sim and Ardupilot on the real-world drone hardware).

V. *ViSafe* Testing

A. Experiment Design

We consider a two-agent interaction scenario. The *ViSafe*-enabled ego agent is tested against an airborne intruder in various collision geometries. We start by identifying three classes of interaction scenarios for our two-agent setup: Head-on, Overtake, and Lateral. A Head-On scenario has the ego agent and the intruder on a single straight-line converging path moving towards each other. An Overtake scenario has the agents on a single straight-line path with a higher-speed ego agent in the trail of a low-speed intruder. A Lateral scenario has the ego and intruder on perpendicular converging paths. Additionally, since our visual detection module can be sensitive to the intruder being above or below the horizon, we investigate both possibilities for our agents, leading to 6 total scenarios. Our six configuration setups are described in Fig. 4. These experiments are performed in both a high-fidelity digital-twin simulation and real-world settings. Table I shows the various agents, collision geometries, commanded ground speeds, and test locations.

B. ViSafe Hardware Prototype

ViSafe uses custom hardware as shown in Fig. 1. The hardware has the following major components:

1) Cameras: 6 x Sony IMX264 cameras providing a total of 220° FOV horizontally and 48° vertically.
2) Compute: NVIDIA AGX Orin Developer Kit
3) Camera Adaptor: Leopard Imaging LI-JXAV-MIPI-ADPT-6CAM-FP

TABLE I
EXPERIMENTAL CONFIGURATIONS

Scenario ID	Intruder	Ownship	Collision Geometry	Hor. Closure Rate	Location
E1	Multirotor (10 m/s)	Aurelia X6 (10 m/s)	Head-On	20 m/s (72 km/hr)	Nardo Airfield
E2	Multirotor (5 m/s)	Aurelia X6 (10 m/s)	Overtake	5 m/s (18 km/hr)	Nardo Airfield
E3	Multirotor (5 m/s)	Aurelia X6 (10 m/s)	Crossing	11.18 m/s (40 km/hr)	Nardo Airfield
E4	VTOL (30 m/s)	Aurelia X6 (10 m/s)	Head-On	40 m/s (144 km/hr)	Nardo Airfield
E5	Multirotor (10 m/s)	Aurelia X6 (10 m/s)	Head-On	20 m/s (72 km/hr)	Leesburg, VA

TABLE II
DIGITAL TWIN & HARDWARE-IN-THE-LOOP BENCHMARKING

	Separation Minima (m) ↑		P(NMAC) ↓		Risk Ratio ↓		Number of violations ↓	
Scenario	Nominal	*ViSafe*	Nominal	*ViSafe*	Nominal	*ViSafe*	Nominal	*ViSafe*
E1	19.9 ± 2.15	35.55 ± 14.00	1.0	0.55	1.0	0.55	4000	2213
E2	22.72 ± 2.63	27.635 ± 6.27	1.0	0.439	1.0	0.439	4000	1759
E3	49.14 ± 0.7	59.04 ± 11.37	1.0	0.5248	1.0	0.5248	4000	2099
Above Horizon - E1, E2, E3	30.58 ± 3.02	50.90 ± 4.72	1.0	0.1	1.0	0.1	6000	605
Below Horizon - E1, E2, E3	28.6 ± 2.73	30.46 ± 3.26	1.0	0.91	1.0	0.91	6000	5466

TABLE III
REAL WORLD BENCHMARKING

	Separation Minima (m) ↑		P(NMAC) ↓		Risk Ratio ↓		Number of violations ↓	
Scenario	Nominal	*ViSafe*	Nominal	*ViSafe*	Nominal	*ViSafe*	Nominal	*ViSafe*
E1	41.28 ± 0.75	51.58 ± 12.76	1.0	0.33	1.0	0.5	4	2
E2	44.25 ± 0.64	56.22 ± 2.88	1.0	0.25	1.0	0.25	4	1
E3	47.45 ± 0.22	71.69 ± 11.65	1.0	0.5	1.0	0.25	4	1
E4	30.54 ± 2.07	46.05 ± 8.29	1.0	0.0	1.0	0.0	2	0
E5	27.39 ± 5.03	38.94 ± 7.83	1.0	0.0	1.0	0.0	3	0
Above Horizon - E1, E2, E3	44.43 ± 3.02	59.43 ± 4.72	1.0	0.0	1.0	0.0	6	0
Below Horizon - E1, E2, E3	44.59 ± 2.73	60.23 ± 18.76	1.0	0.667	1.0	0.667	6	4

4) State Estimation: 3DM-GQ7 GNS/INSS module with a dual-GPS antenna setup.
5) ADS-B In: uAvionix PING™ MAVLink based ADS-B receiver (not used)

The payload is securely mounted below the ownship on a custom mounting plate. The entire payload weighs approx. 1.6 kgs, including the 3D-printed protective casing. The deep-learning pipeline and the collision avoidance algorithm described in Section IV run onboard the payload in real-time. The payload communicates with the ownship autopilot via the MavLink protocol to issue navigation commands in the form of yaw turn rates. We use DroneKit [2] to command the ownship autopilot to execute the avoidance maneuvers. The entire system is agnostic to the ownship platform and is designed to be mounted easily on most systems.

C. Reaction Time Profile for Successful Detect & Avoid

Based on the empirical performance statistics of our vision inference system (AirTrack [17]), we determine the range profile in which the intruder tracks are reliable and the maximum reaction time available for the *ViSafe* system. Leveraging the empirical numbers from the ASTM F3442/F3442M standards (satisfied by AirTrack using a Bell 407 intruder with a frontal diagonal length of 4.27 m), the minimum number of pixels at which AirTrack can reliably detect the object with a 95% probability of track comes out to be 14 pixels. Using the approximate calibrated focal length of our multi-camera payload, the maximum distance d_{max} at which the object can be detected comes out to be $163.2 \times l$, where l is the frontal diagonal length of the intruder. For our real-world intruder platforms (shown in Fig. 1), the reliable maximum distance comes out as $d_{max}^{Hexarotor} = 287m$ and $d_{max}^{VTOL} = 525m$. Thus, for real-world Head-on scenarios, the maximum available reaction time for the M600 intruder is $14.35s$, and for the VTOL intruder, it is $13.12s$. This short maximum reaction time signifies the need for accurate, fast, and on-edge DAA, which we find is empirically satisfied by our *ViSafe* system as shown in both benchmarking (Table II and III) and profiling (Table IV).

D. Simulated Digital-Twin Experiment Setup

We first perform experiments in a digital twin of the test environment. The digital twin enables low-cost, low-risk

TABLE IV
VISAFE SYSTEM PROFILING

Component	Performance
AirTrack Detection & Tracking (GPU)	8 Hz
Multi-View Fusion & State Estimation (CPU)	24 Hz
CBF-based Avoidance Control (CPU)	50 Hz (solving a QP)
Mean CPU Utilization	55.21 %
Mean GPU Utilization	50.02 %
Peak Memory Consumption	15.96 GB

testing in a more diverse and controllable setting.

In order to test the scenario as realistically as possible, we create a digital twin of the real-world testing site. We use the Pegasus Simulator [23], which is a framework built on top of NVIDIA Omniverse and IsaacSim [4]. While designing the assets, we use ArcGIS CityEngine [3], which takes satellite imagery, terrain maps, and structure information from OpenStreetMap to build a 3D model of the test site, including buildings and roads. We also procedurally added trees and a realistic sky with clouds and weather conditions as depicted in Figure 4. The experiments were carried out on a desktop computer with an Nvidia 3080Ti and an Nvidia A6000 running Ubuntu 20.04.

To simulate real-world hardware performance, we transmit the rendered images to the onboard processing units for detection and tracking. This allows us to conduct hardware-in-the-loop (HITL) experiments within the digital-twin framework. Overall, we simulate 10 weather conditions, including time-of-day and cloud covers, three representative collision configurations, and two altitude setups (Above/Below horizon), leading to a total of 60 distinct scenarios. For the above scenarios, we simulated 200 trajectories per scenario, leading to a total of 12000 trajectories.

E. Real World Experiment Setup

We provide details on the real-world experiments to test the *ViSafe* system. Real-world flight tests are conducted at two distinct locations. A dedicated testing facility at the Nardo Flight Test Field [20] (Valentin Vassilev Memorial Field), USA `[FAA Identifier: 77PA]`, `[40-35-00.2420N 79-53-59.1890W]` as well as a secondary test-site in Leesburg, VA, USA are utilized for experiments. All tests are performed under the FAA 14 CFR Part 107 guidelines. The Nardo flight test airfield has an unpaved runway with a total length of 1800ft (550m) and a width of 100ft (30.5m). The Leesburg field is an open test range. Overall, we perform two runs for each configuration shown in Table I. The field experiments use three airborne platforms, as shown in Figure 1. Their details are as follows:

1) Aurelia X6 Standard: A hexa-rotor ownship.
2) Hexarotor: A multi-rotor low-speed intruder.
3) VTOL: A quad-plane vertical takeoff and landing fixed-wing aircraft that acts as our high-speed intruder.

F. Metrics

We compare our technique with a nominal planner without an avoidance strategy. By default, the nominal plans violated the safe separation clause, and we include it solely as a reference point to quantify ViSafe's gains via metrics. Our metrics are inspired by the operational standards proposed for collision avoidance systems by the Radio Technical Commission for Aeronautics (RTCA) [47]. These are the key metrics which we use:

1) *Probability of Near Mid Air Collision P(NMAC)*: This metric measures the probability that two agents come within a predefined unsafe distance. Lower values signify better collision avoidance.
2) *Separation Minima*: This metric captures the minimum distance between agents during an episode. Higher values indicate greater safety margins.
3) *Horizontal Rate of Closure (HROC)*: This metric measures the relative velocity at which two agents approach each other, with negative values indicating impending collisions. Higher values reduce collision risks.
4) *Risk Ratio*: This metric measures the ratio of collisions compared to a baseline (nominal planner). Lower values indicates higher safety improvements over the baseline.

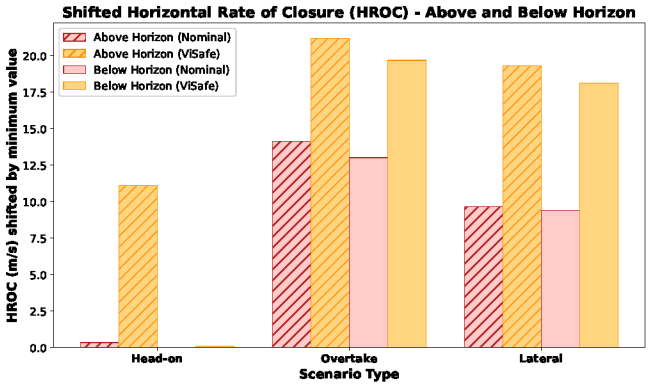

Fig. 5. **Average horizontal rate of closure comparisons across different encounter geometries in real-world testing:** Higher values indicate that agents are moving apart, showcasing diverging & safe trajectories. Under different testing scenarios, it can be seen that *ViSafe* consistently shows a significant boost in average HROC over the nominal plan.

G. Results and Discussion

In this section, we highlight the key findings of comparing *ViSafe* with a nominal planner without an explicit avoidance strategy. The results for simulated digital-twin and real-world experiments are summarized in Tables II and III.

1) ViSafe enables high-speed collision avoidance ***[H1]****:* Our results demonstrate the advantage of *ViSafe* in reducing the collision risk over nominal plan. By design, The nominal planner exhibited high *P(NMAC)* values across Head-On, Overtake, and Lateral configurations for simulation experiments. *ViSafe* drastically reduces the *P(NMAC)* for the same configurations. Our method significantly reduces collision risk and improves safety margins, especially in critical configurations such as head-on and overtaking situations. The total number of loss of separations is reduced significantly by *ViSafe* for all testing configurations in the real world.

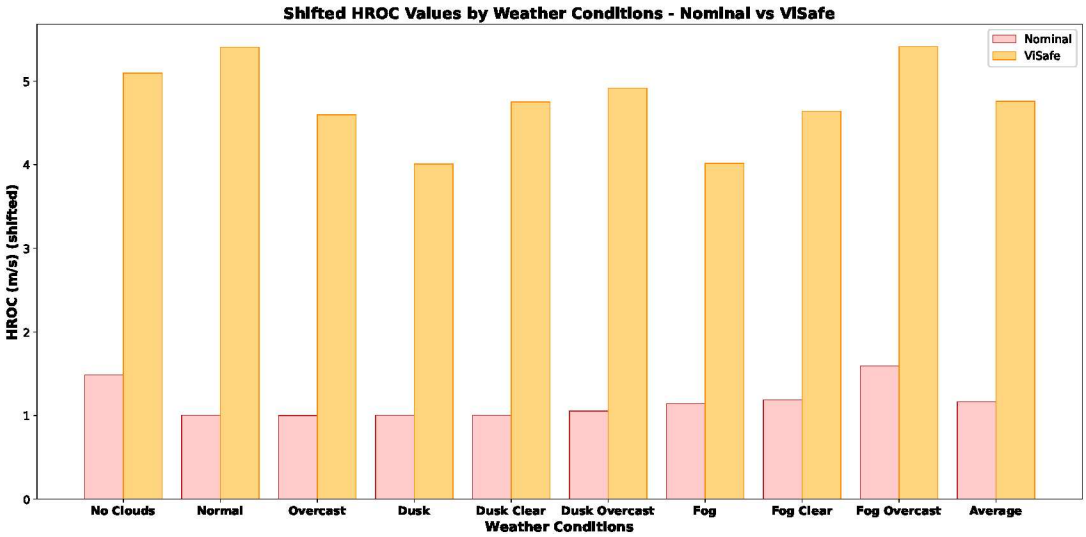

Fig. 6. **Horizontal rate of closure comparisons across different weather conditions in the digital twin:** Higher values indicate that agents are moving apart, showcasing diverging & safe trajectories. Across the different weather scenarios, *ViSafe* showcases consistent boost in HROC over the nominal plan.

The results in Fig. 5 indicate that *ViSafe* improves the HROC values compared to the nominal planner. In the head-on scenario, HROC improved from $-19.25\,\text{m/s}$ (nominal) to $-13.83\,\text{m/s}$ (*ViSafe*), demonstrating that our method facilitates safer approaches by reducing the rate of closure. Similar trends are observed in lateral and overtaking cases, where *ViSafe* maintains a higher and safer HROC, effectively mitigating dangerous approaches.

2) ViSafe is robust to weather and lightning conditions ***[H2]:*** We also test the robustness of *ViSafe* to weather and lighting conditions in simulation. The results are highlighted in II. Our risk ratios increase due to detection and tracking failures for some weather conditions. However, our overall risk ratios with respect to all trajectories are still considerably lower than the nominal values across all configurations. *ViSafe* also achieves higher HROC values than the nominal ones for all weather conditions, indicating safe & divergent trajectories as highlighted in Fig. 6.

3) Digital-twin provides a reliable estimate of real-world performance ***[H3]:*** Our results indicate that the performance metrics observed in the digital-twin environment closely align with those obtained via real-world testing. As summarized in Tables II and III, *ViSafe* demonstrates higher separation minima and low-risk ratios in all configurations, with slightly higher risk ratios in simulation compared to real-world results. We hypothesize that this is due to larger simulated data points that capture more edge cases. In addition, the lower risk ratios observed in real-world tests (indicating higher safety performance) suggest that the digital twin environment provides a conservative estimate of *ViSafe*'s performance. These findings underline the importance of the digital twin as a reliable proxy for predeployment evaluation of safety and finding the lower bound. By identifying and remedying edge case behavior in simulation, *ViSafe* ensures robust real-world operation.

VI. Learned Challenges and Limitations

A. Learned Challenges

There were multiple challenges observed while trying to develop and test *ViSafe*. We list a few of them.

1) **Lack of global information availability:** CBF guarantees invariance when information is available throughout the episode. Since we use vision inference to trigger avoidance, the avoidance window is significantly reduced. Theoretical guarantees provided by traditional CBFs can be compromised without making modifications to the design of the CBF itself.
2) **Actuation constraints and singularities:** Actuation constraints for the ownship drone proved challenging in the CBF design process. We had to theoretically guarantee safety while ensuring we always had a control output that kept us safe. This was achieved by careful hyperparameter tuning for a specific ownship system. In the future, a more principled approach, such as automatic safety index synthesis [56], can be used.
3) **Inaccuracies in vision-based inference:** Vision-based state estimation is not perfect; therefore, false positives can often throw the safety module off. We had to use Kalman filtering to ensure minimal false positives were passed, but in the future, we plan to make it explicitly a part of the CBF to ignore out-of-distribution inputs, leveraging Robust CBFs [10].
4) **Safety guarantees:** We acknowledge that the CBF can only provide a formal safety guarantee given always accurate global perception, i.e., no false negatives, which is not the case in the real world. However, we believe that ensuring compliance with standards and conducting operability studies supported by empirical statistical analysis (similar to our extensive HITL digital twin testing) can help build valid safety cases for DAA systems.

5) **Real world testing constraints:** Testing CBFs on actual drones and collecting data is a time-consuming process, and only a limited number of experiments can be run. Additionally, multiple safety procedures must be put in place before testing, making the process cumbersome. Also, unexpected disturbances like LOS, high wind speeds, and data corruption make testing in the real world challenging. In the future, we plan to use the drone telemetry data we have collected to test our approach and fine-tune it, further exploring ways to improve simulation fidelity.

B. Limitations

Across our wide array of simulation and real-world tests, we find that our current system struggles when the intruder is below the horizon. As acknowledged in the benchmarking of our prior work (AirTrack [17]), the primary reason is that the vision-based inference system fails to establish a consistent track when the background changes fast. Furthermore, as shown in prior work [34], this performance behavior can be attributed to biases or shortcuts learned by the model given the training data distribution. We aim to leverage optical flow and temporal information in future work to mitigate this issue. Lastly, for the design of our CBF, we make the following assumptions: the intruder has a constant velocity vector, the action space of the ownship is constrained to a 2D plane parallel to the ground plane, and the intruder doesn't react to the ownship's actions. Although these are used to make the problem tractable, we aim to investigate interaction-aware avoidance without explicit communication in future work using adaptive control barrier functions, improving the versatility of our *ViSafe* system.

VII. Conclusion

In this work, we present a comprehensive solution for high-speed vision-only airborne collision avoidance. Our full-stack solution, *ViSafe*, deploys perceptual input-focused control barrier functions (CBF) for safe self-separation in dynamic environments. Through simulated hardware-in-the-loop digital twin testing and real-world flight tests, we exemplify *ViSafe*'s reliability in the face of heterogeneous agents, varying encounter geometries, and other environmental factors. *ViSafe*'s successful deployment in real-world high-speed collision avoidance tests (at closure rates of up to 144 km/h) establishes its effectiveness as a cornerstone for autonomous aerial navigation systems.

Acknowledgments

Approved for public release; distribution is unlimited. This research was sponsored by DARPA (W911NF-18-2-0218). The views, opinions, and/or findings expressed are those of the author(s) and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government. The authors thank Rebecca Martin, Helen Wang, Nayana Suvarna, and folks at Near Earth Autonomy for their support with the field testing of ViSafe. We also thank John Keller for helping out with the payload.

References

[1] Airborne object tracking dataset. URL https://registry.opendata.aws/airborne-object-tracking. 3, 4

[2] Dronekit-python: Developer sdk for python to control drones using the mavlink protocol. URL https://github.com/dronekit/dronekit-python. 8

[3] Procedural City Generator — 3D City Maker — ArcGIS CityEngine — esri.com. https://www.esri.com/en-us/arcgis/products/arcgis-cityengine/overview. [Accessed 11-12-2024]. 9

[4] Isaac Sim — developer.nvidia.com. https://developer.nvidia.com/isaac/sim. [Accessed 11-12-2024]. 9

[5] Nvidia isaac sim, 2022. URL https://developer.nvidia.com/isaac-sim. 2

[6] Ayush Agrawal and Koushil Sreenath. Discrete control barrier functions for safety-critical control of discrete systems with application to bipedal robot navigation. In *Robotics: Science and Systems*, 2017. URL https://api.semanticscholar.org/CorpusID:1780280. 2

[7] Aaron D. Ames, Samuel Coogan, Magnus Egerstedt, Gennaro Notomista, Koushil Sreenath, and Paulo Tabuada. Control barrier functions: Theory and applications. In *2019 18th European Control Conference (ECC)*, pages 3420–3431, 2019. doi: 10.23919/ECC.2019.8796030. 2, 6

[8] DM Asmar, MJ Kochenderfer, DM Asmar, and MJ Kochenderfer. Optimized airborne collision avoidance in mixed equipage environments. *Lincoln Laboratory, MIT Report*, 2013. 2

[9] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *2016 IEEE international conference on image processing (ICIP)*, pages 3464–3468. IEEE, 2016. 3

[10] Joseph Breeden and Dimitra Panagou. Robust control barrier functions under high relative degree and input constraints for satellite trajectories, 2021. URL https://arxiv.org/abs/2107.04094. 10

[11] Ryan Carnie, Rodney Walker, and Peter Corke. Image processing algorithms for uav" sense and avoid". In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 2848–2853. IEEE, 2006. 3

[12] Xueyun Chen, Shiming Xiang, Cheng-Lin Liu, and Chun-Hong Pan. Aircraft detection by deep convolutional neural networks. *IPSJ Transactions on Computer Vision and Applications*, 7:10–17, 2014. 3

[13] Debadeepta Dey, Christopher Geyer, Sanjiv Singh, and Matt Digioia. Passive, long-range detection of aircraft: towards a field deployable sense and avoid system. In *Field and Service Robotics*, pages 113–123. Springer, 2010. 3

[14] Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. Centernet: Keypoint triplets for object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*,

pages 6569–6578, 2019. 3

[15] Giancarmine Fasano, Domenico Accardo, Anna Elena Tirri, Antonio Moccia, and Ettore De Lellis. Morphological filtering and target tracking for vision-based uas sense and avoid. In *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 430–440. IEEE, 2014. 3

[16] Junjie Fu, Guanghui Wen, and Jinde Cao. High-order control barrier function based collision avoidance formation tracking of constrained fixed-wing aircraft. In *2023 5th International Conference on Industrial Artificial Intelligence (IAI)*, pages 1–6, 2023. doi: 10.1109/IAI59504.2023.10327509. 2

[17] Sourish Ghosh, Jay Patrikar, Brady Moon, Milad Moghassem Hamidi, and Sebastian Scherer. Airtrack: Onboard deep learning framework for long-range aircraft detection and tracking. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1277–1283, 2023. doi: 10.1109/ICRA48891.2023.10160627. 2, 3, 4, 5, 8, 11

[18] Asma Hamissi, Amine Dhraief, and Layth Sliman. A comprehensive survey on conflict detection and resolution in unmanned aircraft system traffic management. *IEEE Transactions on Intelligent Transportation Systems*, 2024. 1

[19] Marvin Harms, Mihir Kulkarni, Nikhil Khedekar, Martin Jacquet, and Kostas Alexis. Neural control barrier functions for safe navigation, 2024. URL https://arxiv.org/abs/2407.19907. 2

[20] Yao He, Ivan Cisneros, Nikhil Keetha, Jay Patrikar, Zelin Ye, Ian Higgins, Yaoyu Hu, Parv Kapoor, and Sebastian Scherer. Foundloc: Vision-based onboard aerial localization in the wild. *arXiv preprint arXiv:2310.16299*, 2023. 9

[21] Peiyun Hu and Deva Ramanan. Finding tiny faces. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 951–959, 2017. 3

[22] Sunyou Hwang, Jaehyun Lee, Heemin Shin, Sungwook Cho, and David Hyunchul Shim. Aircraft detection using deep convolutional neural network in small unmanned aircraft systems. In *2018 AIAA Information Systems-AIAA Infotech@ Aerospace*, page 2137, 2018. 3

[23] Marcelo Jacinto, João Pinto, Jay Patrikar, John Keller, Rita Cunha, Sebastian Scherer, and António Pascoal. Pegasus simulator: An isaac sim framework for multiple aerial vehicles simulation. In *2024 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 917–922, 2024. doi: 10.1109/ICUAS60882.2024.10556959. 9

[24] Jasmin James, Jason J Ford, and Timothy L Molloy. Learning to detect aircraft for long-range vision-based sense-and-avoid systems. *IEEE Robotics and Automation Letters*, 3(4):4383–4390, 2018. 3

[25] Jasmin James, Jason J Ford, and Timothy L Molloy. Below horizon aircraft detection using deep learning for vision-based sense and avoid. In *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 965–970. IEEE, 2019. 3

[26] Nikhil Varma Keetha, Chen Wang, Yuheng Qiu, Kuan Xu, and Sebastian Scherer. Airobject: A temporally evolving graph embedding for object identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8407–8416, 2022. 3

[27] JE Kuchar and Ann C Drumm. The traffic alert and collision avoidance system. *Lincoln laboratory journal*, 16(2):277, 2007. 2

[28] John Lai, Jason J Ford, Peter O'Shea, and Rodney Walker. Hidden markov model filter banks for dim target detection from image sequences. In *2008 Digital Image Computing: Techniques and Applications*, pages 312–319. IEEE, 2008. 3

[29] John Lai, Luis Mejias, and Jason J Ford. Airborne vision-based collision-detection system. *Journal of Field Robotics*, 28(2):137–157, 2011. 3

[30] John Lai, Jason J Ford, Luis Mejias, and Peter O'Shea. Characterization of sky-region morphological-temporal airborne collision detection. *Journal of Field Robotics*, 30(2):171–193, 2013. 3

[31] Dong Liang, Zongqi Wei, Dong Zhang, Qixiang Geng, Liyan Zhang, Han Sun, Huiyu Zhou, Mingqiang Wei, and Pan Gao. Learning calibrated-guidance for object detection in aerial images. *arXiv preprint arXiv:2103.11399*, 2021. 3

[32] Changliu Liu and Masayoshi Tomizuka. Control in a safe set: Addressing safety in human-robot interactions. In *IEEE/ACM International Conference on Human-Robot Interaction*, 2014. 6

[33] Guido Manfredi and Yannick Jestin. An introduction to acas xu and the challenges ahead. In *2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC)*, pages 1–9, 2016. doi: 10.1109/DASC.2016.7778055. 1, 3

[34] Rebecca Martin, Clement Fung, Nikhil Keetha, Lujo Bauer, and Sebastian Scherer. Targeted image transformation for improving robustness in long range aircraft detection. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10431–10438. IEEE, 2024. 11

[35] Jeffrey W McCandless. Detection of aircraft in video sequences using a predictive optical flow algorithm. *Optical Engineering*, 38(3):523–530, 1999. 3

[36] Luis Mejias, Scott McNamara, John Lai, and Jason Ford. Vision-based detection and tracking of aerial targets for uav collision avoidance. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 87–92. IEEE, 2010. 3

[37] Timothy L Molloy, Jason J Ford, and Luis Mejias. Detection of aircraft below the horizon for vision-based detect and avoid in unmanned aircraft systems. *Journal of Field Robotics*, 34(7):1378–1391, 2017. 3

[38] Tamas G. Molnar, Suresh K. Kannan, James Cunningham, Kyle Dunlap, Kerianne L. Hobbs, and Aaron D.

Ames. Collision avoidance and geofencing for fixed-wing aircraft with control barrier functions, 2024. URL https://arxiv.org/abs/2403.02508. 2

[39] Muhammad Ali Murtaza, Sergio Aguilera, Vahid Azimi, and Seth Hutchinson. Real-time safety and control of robotic manipulators with torque saturation in operational space. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 702–708, 2021. doi: 10.1109/IROS51168.2021.9636794. 2

[40] Andreas Nussberger, Helmut Grabner, and Luc Van Gool. Aerial object tracking from an airborne platform. In *2014 international conference on unmanned aircraft systems (ICUAS)*, pages 1284–1293. IEEE, 2014. 3

[41] Michael P. Owen, Sean M. Duffy, and Matthew W. M. Edwards. Unmanned aircraft sense and avoid radar: Surrogate flight testing performance evaluation. In *2014 IEEE Radar Conference*, pages 0548–0551, 2014. doi: 10.1109/RADAR.2014.6875652. 2

[42] Jay Patrikar, Joao Dantas, Sourish Ghosh, Parv Kapoor, Ian Higgins, Jasmine J Aloor, Ingrid Navarro, Jimin Sun, Ben Stoler, Milad Hamidi, et al. Challenges in close-proximity safe and seamless operation of manned and unmanned aircraft in shared airspace. *arXiv preprint arXiv:2211.06932*, 2022. 2

[43] Jay Patrikar, Joao Dantas, Brady Moon, Milad Hamidi, Sourish Ghosh, Nikhil Keetha, Ian Higgins, Atharva Chandak, Takashi Yoneyama, and Sebastian Scherer. Image, speech, and ads-b trajectory datasets for terminal airspace operations. *Scientific Data*, 12(1):468, 2025. 3

[44] Stavros Petridis, Christopher Geyer, and Sanjiv Singh. Learning to detect aircraft at low resolutions. In *International Conference on Computer Vision Systems*, pages 474–483. Springer, 2008. 3

[45] Vladimir Reilly, Haroon Idrees, and Mubarak Shah. Detection and tracking of large number of targets in wide area surveillance. In *European conference on computer vision*, pages 186–199. Springer, 2010. 3

[46] Artem Rozantsev, Vincent Lepetit, and Pascal Fua. Flying objects detection from a single moving camera. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4128–4136, 2015. 3

[47] RTCA. *Minimum Operational Performance Standards for Airborne Collision Avoidance System X (ACAS X) (ACAS Xa and ACAS Xo), Volume I and Volume II*. DO 385A. RTCA, 2023. Approved by the RTCA Program Management Committee (PMC) as documented in June 2023. 9

[48] Falk Schubert and Krystian Mikolajczyk. Robust registration and filtering for moving object detection in aerial videos. In *2014 22nd International Conference on Pattern Recognition*, pages 2808–2813. IEEE, 2014. 3

[49] Tianjun Shi, Jinnan Gong, Shikai Jiang, Xiyang Zhi, Guangzhen Bao, Yu Sun, and Wei Zhang. Complex optical remote-sensing aircraft detection dataset and benchmark. *IEEE Transactions on Geoscience and Remote Sensing*, 61:1–9, 2023. 3

[50] Elysia Smyers, Sydney Katz, Anthony Corso, and Mykel J Kochenderfer. Avoidds: Aircraft vision-based intruder detection dataset and simulator. *Advances in Neural Information Processing Systems*, 36:8076–8091, 2023. 3

[51] Eric Squires, Pietro Pierpaoli, and Magnus Egerstedt. Constructive barrier certificates with applications to fixed-wing aircraft collision avoidance. In *2018 IEEE Conference on Control Technology and Applications (CCTA)*, pages 1656–1661, 2018. doi: 10.1109/CCTA.2018.8511342. 3

[52] Eric Squires, Pietro Pierpaoli, Rohit Konda, Samuel Coogan, and Magnus Egerstedt. Composition of safety constraints with applications to decentralized fixed-wing collision avoidance. *CoRR*, abs/1906.03771, 2019. URL http://arxiv.org/abs/1906.03771. 3

[53] Vladan Stojnić, Vladimir Risojević, Mario Muštra, Vedran Jovanović, Janja Filipi, Nikola Kezić, and Zdenka Babić. A method for detection of small moving objects in uav videos. *Remote Sensing*, 13(4):653, 2021. 3

[54] Bo Yang, Dongjian Tian, Songliang Zhao, Wei Wang, Jun Luo, Huayan Pu, Mingliang Zhou, and Yangjun Pi. Robust aircraft detection in imbalanced and similar classes with a multi-perspectives aircraft dataset. *IEEE Transactions on Intelligent Transportation Systems*, 2024. 3

[55] Mingxin Yu, Chenning Yu, M-Mahdi Naddaf-Sh, Devesh Upadhyay, Sicun Gao, and Chuchu Fan. Efficient motion planning for manipulators with control barrier function-induced neural controller, 2024. URL https://arxiv.org/abs/2404.01184. 2

[56] Weiye Zhao, Tairan He, Tianhao Wei, Simin Liu, and Changliu Liu. Safety index synthesis via sum-of-squares programming, 2023. URL https://arxiv.org/abs/2209.09134. 10

[57] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Tracking objects as points. In *European Conference on Computer Vision*, pages 474–490. Springer, 2020. 3, 5