FalconWing: An Ultra-Light Fixed-Wing Platform for Indoor Aerial Applications

Anonymous Author(s)
Affiliation
Address
email

Abstract

We introduce *FalconWing*, an ultra-light (150 g) fixed-wing platform for indoor, vision-based autonomy. Controlled indoor settings enable year-round, repeatable UAV experiments but impose strict mass and maneuverability limits, motivating a sensor-minimal design. FalconWing couples a lightweight hardware stack (137 g UMX airframe, 9 g analog FPV camera, offboard compute) with a *world model* composed of a photorealistic 3D Gaussian Splat (GSplat) simulation environment and a system-identified nonlinear dynamics. We validate FalconWing on two tasks without IMU or motion capture: leader-follower visual tracking and autonomous visual landing. Policies are trained via imitation entirely *inside the world model*, augmented with domain randomization over appearance and geometry. In simulation, our best learned policy attains 100% success across three unseen leader maneuvers and is robust to appearance shifts. For landing, a policy trained purely in the world model transfers *zero-shot* to hardware, achieving an 80% success rate over ten indoor trials. We will release hardware designs, GSplat scenes, dynamics parameters, and ROS workflows, positioning FalconWing as an open-source benchmark and educational kit for *world-model-driven* vision-based fixed-wing autonomy.

1 Introduction

Figure 1: **Left:** Our ultra-light 150 g fixed-wing aircraft for indoor aerial research, equipped with a FPV camera and ROS-enabled autonomous control. **Middle:** Onboard view for leader-follower visual tracking using a digital camera. **Right:** Onboard view during autonomous landing using an analog camera.

Autonomous fixed-wing aircraft are attractive for delivery [1], navigation [2, 3], and environmental monitoring [4] due to their energy efficiency and long endurance. Vision-based aerial autonomy [5, 6] is important in GPS-denied zones, but it is challenging for fixed-wing aircraft: the vehicle must maintain airspeed to generate lift; it is governed by nonlinear aerodynamics; and the onboard video stream can degrade due to vibration and turbulence.

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

Existing research on fixed-wing UAVs addresses these challenges using relatively large [7, 8, 9, 10] and sensor-rich platforms [2, 3] with GPS/GNSS, lidar, high-resolution cameras, and onboard computation. While these platforms are suitable for large-scale outdoor experiments, such experiments typically require large, regulated airspaces and are constrained by weather and time-of-day limitations, reducing accessibility and experimental throughput. In contrast, indoor spaces, such as our $40\times20\times5$ m flying arena (Figure 3), as well as typical university gyms, auditoriums, and stadiums, provide weather/time-independent controlled environments where experiments can be scheduled year-round, thus enabling more frequent, repeatable, and accessible experiments and controlled perturbations (e.g., fan-generated wind).

Indoor flight, however, imposes tight weight and maneuverability constraints: every additional gram raises the minimum required airspeed and thus increases the minimum turning radius. A 150 g aircraft requires approximately 7 m/s minimum airspeed and an 8.7 m minimum turning radius (derivations in the Appendix); a 300 g aircraft (e.g., adding a 174 g Jetson Orin) would require about 10 m/s and a 17.7 m turning radius. This shows that existing heavy and sensor-rich research platforms [2] with onboard computation are often impractical to maneuver within a small space (e.g., 20 m indoor width) and leaves an important yet under-explored design gap for a lightweight, sensor-minimal fixed-wing UAV suitable for iterative and reproducible indoor experiments.

To address this gap, we introduce *FalconWing*, a vision-based fixed-wing aircraft research platform weighing just 150 g with a 9 g analog camera and off-board computation. FalconWing couples (i) a sensor-minimal hardware stack supporting both manual and autonomous modes (Section 3) with (ii) a world model comprising a photorealistic Gaussian Splat (GSplat) simulator (Section 4.1) and a system-identified nonlinear dynamics model (Section 4.2).

To demonstrate FalconWing's capability, we tackle two challenging aerial tasks *without* using IMU or motion capture: leader–follower visual tracking in simulation of the world model (Section 5) and zero-shot sim-to-real transfer of a vision-based autonomous landing controller in indoor environments (Section 6). In the visual-tracking case study, building on FalconGym [5], we design a vision controller that removes reliance on known target states and IMU by decoupling perception from control and training with domain randomization applied to GSplat renders. Experiments show that our vision policies improve generalization to unseen maneuvers and appearance shifts. In the autonomous-landing case study, a vision policy trained purely in the world model transfers *zero-shot* to hardware, achieving an 80% success rate over ten indoor trials without fine-tuning.

In summary, our contributions are: (i) *FalconWing platform:* we design an ultra-light indoor fixed-wing platform with ROS integration, safety mechanisms, and dual operating modes, paired with a world model consisting of photorealistic GSplat and system-identified dynamics; (ii) *Demonstration on two challenging case studies:* utilizing FalconWing's world model, we train vision-only controllers that (a) track a leader with unseen maneuvers and appearance perturbations in simulation and (b) perform autonomous landing with *zero-shot* transfer achieving 80% success on real hardware.

2 Related Work

Fixed-Wing Autonomy Traditional approaches to fixed-wing autonomy rely heavily on external sensors such as GPS and IMUs for state estimation and control. For example, [2] demonstrated accurate vision-aided navigation using GNSS, IMU-assisted Kalman filtering, while [3] achieved agile maneuvering with high-precision motion capture systems. Some work tries to mitigate reliance on tracking sensors by adding learning-based perception modules like faster R-CNN [9] and YOLO [10] for pose estimation but still requires high-fidelity digital cameras. [11, 12, 13] have attempted vision-based flight for landing, but all of those platforms rely on large, heavy platforms (e.g., 1-5kg) with multi-sensor suites, making them impractical for agile, indoor, or GPS-denied scenarios.

Neural Scene Representation in Robotics Neural scene representations like Gaussian Splat [14] have recently emerged as powerful tools for photorealistic 3D scene reconstruction. Variants and extensions have been applied to diverse robotics tasks, including 3D scene editing [15], pose estimation [16, 17] and navigation [18]. NeRF2Real [19] and RialTo [20] demonstrate the potential real-to-sim-to-real transfer for ground robots and robot arm manipulation, by constructing a simulation based on real images, and deploy simulation-trained model to real world again. [5, 6] achieves zero-shot drone navigation using policies trained in neural scene representation.

3 FalconWing Hardware Stack

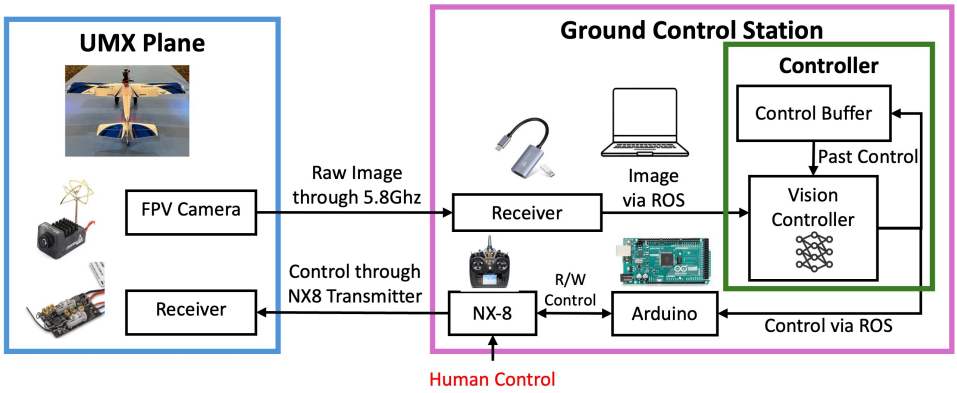

Figure 2: Architecture of FalconWing Hardware: a light 9 g FPV camera mounted on the fixed-wing plane streams images to the ground control station, where images are published to ROS. The controller reads published image plus buffered past controls, computes new flight control, and sends it via ROS to an Arduino. The Arduino writes these commands into the Spektrum NX-8 trainer port, closing the vision-based control loop over radio. The human pilot can instantly reclaim control at any time via a transmitter switch.

Base Airframe The platform builds on the UMX Turbo Timber® (Figure 1 Left), a hobby-grade airframe chosen for its lightweight design (137 g), integrated electronic speed controller (ESC) and flight controller, and off-the-shelf parts availability. Its durable foam fuselage withstands crashes during iterative testing, while the 70 cm wingspan with flaps balances maneuverability and provides extra lift.

Vision System A 9 g RunCam Spotter® analog FPV camera provides onboard vision. Mounted along the fuselage centerline via a custom 3D-printed bracket (5 g), the camera avoids propeller occlusion and preserves the center of gravity. Images are transmitted via a 5.725 GHz analog link to a ground control station, where a diversity receiver forwards the signal to a USB capture card. The card streams 640×480 RGB frames at 100 Hz into a ROS topic, enabling real-time processing. An LC filter is also connected to the camera to reduce high-frequency noise from motor vibrations. We also provide the option to switch to a digital camera, if users value image quality over light-weight.

Control Interface Autonomous flight is enabled through the Spektrum NX-8® transmitter's trainer port (Figure 2). An Arduino Mega 2560 bridges ROS and the transmitter via rosserial-python, translating four-channel pulse-position modulation (PPM) signals (throttle, aileron, elevator, and rudder) between ROS messages (20 Hz) and the transmitter's serial interface.

Operating Modes We configure two modes that support safe experiments:

- **Manual Mode:** A human pilot flies via the NX-8. The Arduino logs pilot commands and time-synchronized images to ROS to build datasets for system identification and controller training. Because we retain the 13.5 g Horizon Hobby receiver (Fig. 1) and its integrated flight controller (with IMU), expert pilots can perform robust manual flight and aerobatics if desired. Switching to Autonomous Mode requires only a single transmitter toggle.
- **Autonomous Mode:** The Arduino subscribes to the ROS topic publishing control commands from the vision-based controller and writes these commands to the trainer port, closing the perception-action loop. The human pilot can instantly take over by flipping the same switch whenever intervention is required.

Safety Mechanisms To mitigate the risk of degraded analog video during Autonomous Mode, we deploy a frame-quality monitor based on the Structural Similarity Index (SSIM) [21]. If the SSIM between consecutive frames falls below an empirical threshold for five consecutive frames, we raise a flag alerting human pilots to take control immediately.

4 FalconWing Software Stack

In this section, we introduce FalconWing's software stack, which include 2 variations of photorealistic simulation environment (Section 4.1), identified airplane dynamics used for simulation training and testing (Section 4.2) and the open-source availability (Section 4.3).

4.1 Photorealistic Simulation via Gaussian Splatting

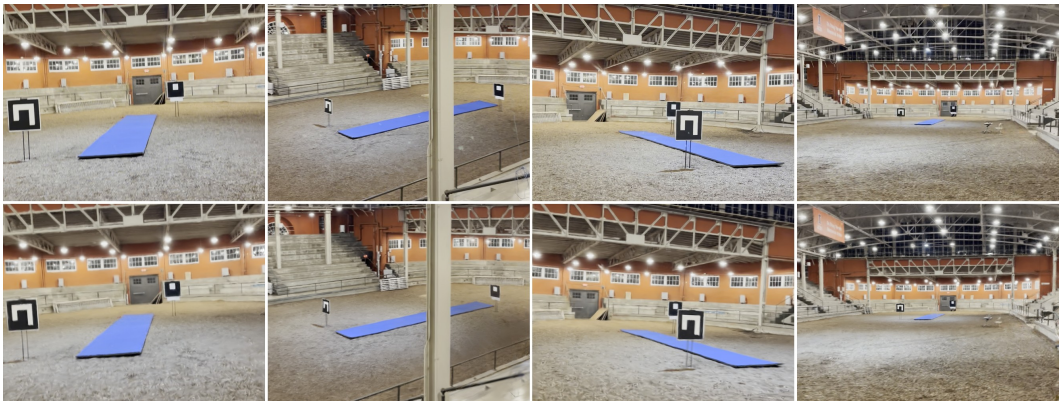

Figure 3: FalconWing's simulation can render photorealistic images using Gaussian Splat from different poses. The top row shows 4 real world images taken by the camera, while bottom row displays the corresponding images rendered by GSplat at the same coordinates.

A photorealistic simulation environment can help mitigate the sim-to-real gap in vision-based control. In FalconWing, we synthesize a photorealistic simulation environment G, that can render a photorealistic image I from any virtual camera pose p in world coordinates, i.e., $I = G(p)$. We achieve this by enhancing the original FalconGym [5] pipeline, replacing Neural Radiance Fields (NeRF)[22] with Gaussian Splat (GSplat) [14] for faster and better rendering, and eliminating the need for a motion capture system.

Data Collection and Calibration A human operator carried the onboard FPV camera to capture approximately 2000 images throughout the indoor arena from diverse viewpoints. We recover camera intrinsics and initial poses using COLMAP [23]. Since COLMAP's coordinate frame is arbitrary, we align it to the real-world frame by placing an 80 cm ArUco marker beside the runway (Figure 3). Using OpenCV's ArUco detector, we identify the marker center in a subset of images and treat it as the global origin. We then compute the rigid transform between COLMAP and world frames via the Kabsch–Umeyama algorithm [24].

GSplat-based Simulation Construction With calibrated poses, we feed the images and transforms into the open-source NeRFStudio Splatfacto pipeline [25]. On an NVIDIA RTX 4090, training converges in approximately 15 minutes. The resulting model supports fast rendering with an average of 0.004s for a 960x720 image.

Digital Camera Variant Simulation To accommodate researchers who favor image quality over minimal mass, we repeat the above procedure using a digital ArduCam RGB camera. This provides an additional digital camera-based simulation environment. Figure 3 qualitatively compares real-world images with renders from simulation environment.

UMX Gaussian Splatting Utilizing the same techniques, we also construct our UMX plane as a GSplat asset. By combining the UMX plane's GSplat model with the flying arena GSplat model, we have the capability to place and render a photorealistic leader aircraft in simulation for the Leader-Follower case study (Section 5) with the learned dynamics (Section 4.2).

4.2 Non-linear System Identification

With the photorealistic simulation established in Section 4.1, we now aim to obtain a reliable dynamics model of our FalconWing aircraft. We adopt a reduced-order kinematic model inspired by standard fixed-wing dynamics formulations [26]. Specifically, we represent the aircraft state as $x = [p_x, p_y, p_z, \theta, \gamma, \phi, v_x, v_y, v_z]$, which captures the aircraft's position, orientation (pitch, yaw, roll), and linear velocity. Note the first 6 state variable is exactly camera (plane) pose p. Our control inputs are defined as $u = [u_T, \delta_a, \theta_c, \gamma_c]$, corresponding to throttle, commanded aileron, elevator and rudder. We model the discrete-time non-linear dynamics as a parametric function f_K, i.e., $f_K(x_t, u_t)$ where K denotes the vector of unknown dynamics parameters we seek to estimate.

Hybrid State Estimation Since no motion capture system is yet available in our flying arena, we must estimate ground-truth states purely from visual input. We propose a hybrid vision-based state-estimation pipeline: when the aircraft is close enough to the ArUco marker and it's detectable by the OpenCV ArUco library, we directly use its estimates; otherwise, we utilize a neural network-based inverse Gaussian Splat (iGSplat) model to infer camera poses from single RGB frames. Although iterative pose-optimization methods such as iNeRF [16] can estimate camera poses accurately, they are computationally expensive. Therefore, we train a single-shot neural network architecture for efficient inference. Specifically, our iGSplat model employs a Vision Transformer (ViT) backbone pretrained on ImageNet-21k [27, 28]. We freeze early transformer layers to leverage general visual feature extraction, adding a trainable regression head for direct camera-pose estimation. To circumvent discontinuities inherent in angular regression, our network predicts sine and cosine values of pitch, yaw, and roll angles, subsequently recovering angular orientations via a trigonometric transformation.

System Parameter Identification through Least Square We collect a dataset $\mathcal{D}_I = \{(I_t, u_t)\}$ by recording images and pilot inputs during Manual Mode landings (Section 3). Applying our hybrid estimator yields pose sequences $\{p_t\}$, which we differentiate to obtain velocity and form state-action pairs $\mathcal{D}_x = \{(x_t, u_t)\}$, where frames corrupted by significant analog noise or yielding clearly implausible pose estimations are manually removed. Using the cleaned dataset, we solve

$$K^* = \arg\min_K \sum_{(x_t,u_t)\in\mathcal{D}_x} \left\| f_K(x_t, u_t) - x_{t+1} \right\|_2^2$$

via nonlinear least squares (SciPy). The optimized parameter set K^* yields a reliable dynamics model for controller design and training in simulation.

4.3 Open-source Software Package

In addition to the hardware part list and user manual, we also plan to open-source the complete FalconWing software stack including: two photorealistic simulation environments (analog and digital camera variants) and system-identified dynamics parameters with everything packed in a conda environment for easy distribution. This digital twin is designed as an open-source reusable benchmark for future research in vision-based fixed-wing control.

We next demonstrate FalconWing's capabilities through two of the most challenging aerial tasks: visual tracking (Section 5) and visual autonomous landing (Section 6). Note that although our FalconWing hardware (Section 3) does carry a self-leveling flight controller, which researchers may choose to employ in their applications, we deliberately disable the autopilot assists for both case studies so as to isolate pure vision-policy performance.

5 Case Study: Leader-Follower Visual Tracking

In this case study, we consider the problem of visual tracking, where the follower UMX aircraft needs to track a leading UMX aircraft using vision. Fixed-wing leader-follower visual tracking is vital for tasks such as search-and-rescue, delivery and aerial navigation. Yet visual tracking is challenging due to the small size of the aircraft (in our case, 70cm×52cm) in the image space, its nonlinear underactuated dynamics, and the lack of ground-truth state feedback.

In the following subsections, we develop three distinct neural visual tracking policies and evaluate their ability to track under three types of leader representative maneuvers: a left-turn S-shape descent,

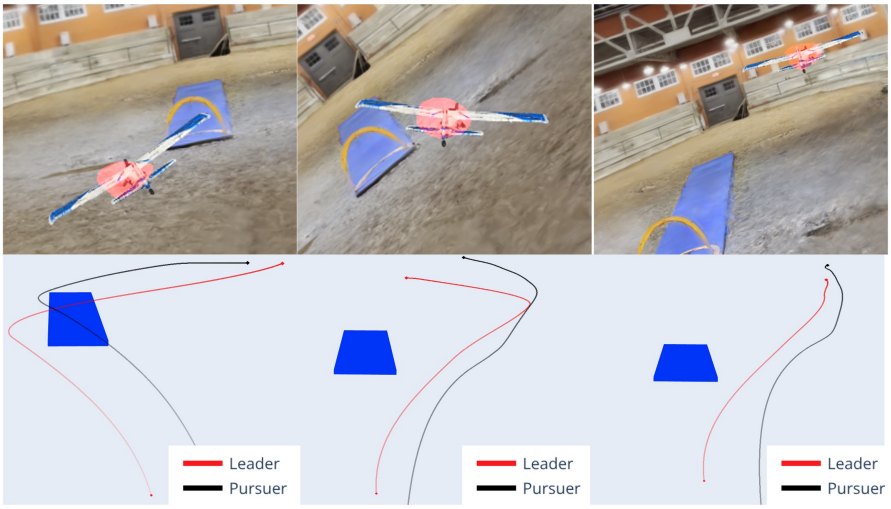

Figure 4: Trajectory plot and onboard perception view for the Leader-Follower Case Study: our vision-based policy on the follower can closely track the leaders with three different leader maneuvers. The annotated red part on the onboard images indicating the mask detection result described in Section 5.2.

a right-turn S-shape ascent, and a right-turn sharp climb, as shown in Figure 4. As shown in Table 1, we measure the tracking performance using 3 key metric: (i). Success Rate (SR), fraction of trials where the follower maintains visual lock on the leader throughout all frames; (ii) Average Tracking Error (ATE), defined as the average displacement between the leader and the follower minus the initial tracking offset; (iii) Average Runtime(ART), per-frame inference time of the control policy.

5.1 Vision Controller: Direct Imitation Learning

Building on FalconGym's [5] success in quadrotor navigation via imitation learning, we started by designing an end-to-end vision policy that maps onboard RGB images directly to fixed-wing control commands, while addressing three key limitations of FalconGym: (i) reliance on known target positions, (ii) dependence on IMU readings, (iii) heavyweight dual-ViT architecture. To overcome these constraints, our network ingests only the current image I_t and a history of the past 30 control inputs $u_{t-30:t-1}$, which implicitly encode temporal state information and fuses them via a lightweight self-attention module (green box, Figure 2), thereby eliminating explicit pose estimation and IMU usage while reducing model size for faster inference time. We empirically try out different length of history and find 30 being a suitable length of history that balances model size and effectiveness.

We train this multi-modal policy $\pi_\psi(I_t, u_{t-30:t-1})$ by learning from an expert. The expert state-based controller π^* is first implemented following standard fixed-wing designs [29]. Then we configure the leader UMX plane to always to use the expert state-based controller to follow predefined maneuvers. And during training data collection, we also configure the follower plane to use the expert state-based controller that has access to both leader's ground truth state $\hat{x}_t$ and its own ground truth state x_t to compute optimal actions $u_t^* = \pi^*(x_t, \hat{x}_t)$ To broaden the state-action distribution and improve robustness, we inject mild Gaussian noise, $u_t = u_t^* + \mathcal{N}(0, \sigma^2)$, into the follower's expert outputs, encouraging slight exploration of off-nominal trajectories without compromising feasibility. We render the image I_t with the leader UMX plane at each state x_t using the UMX asset (Section 4.1) in FalconWing's photorealistic simulation and log the expert history and noisy action to form the dataset $\mathcal{D}_C = \left\{(I_t,\, u_{t-30:t-1},\, u_t^*)\right\}$. This dataset consists of 100 trajectories (with around 8k data pairs) by setting random but dynamically feasible waypoints for leader UMX to explore. We then optimize the policy parameters ψ by minimizing the mean squared error between predicted and expert actions:

$$\mathcal{L}(\psi) = \frac{1}{|\mathcal{D}_C|} \sum_{(I_t, u_{t-30:t-1}, u_t^*) \in \mathcal{D}_C} \left\| \pi_\psi(I_t, u_{t-30:t-1}) - u_t^* \right\|_2^2.$$

We refer to this policy as "RGB" in Table 1, because the visual input to the controller is a RGB image. While "RGB" attains high SR and low ATE on the training trajectories, it fails to generalize to unseen

leader maneuvers. This is consistent with the observation in FalconGym [5] that a single end-to-end vision policy tends to specialize to a single scenario. We believe this overfitting failure mode happens because the leader UMX occupies only a small fraction of image pixels, so the network overfits to background appearance rather than learning a transferable representation of the leader aircraft.

5.2 Vision Controller: UMX Mask Detection

To improve generalization and avoid overfitting, we decouple perception from control: a UNet [30] first predicts a binary mask of the leader from the onboard RGB image; a lightweight ResNet then consumes the mask together with the past control history to output the next action. We denote this approach as "Mask" in Table 1.

Training the UNet also leverages our UMX GSplat assets (Section 4.1). We synthesize the perception training dataset by sampling leader plane poses across the arena workspace and spawning the follower aircraft (camera) at feasible viewpoints (ensuring the leader is roughly front-facing). Because the poses of Gaussians corresponding to the leader's wing tips and nose-tail endpoints are known and the camera matrix is calibrated, we can approximate the aircraft as a 3D ellipsoid and project it onto the image plane to obtain ground-truth masks using standard geometry with traditional computer vision techniques. We collect 3000 pairs of RGB images and masks image to train for this UNet. After training the UNet, we apply the same imitation-learning setup as in Section 5.1 but train the controller to map mask and past controls to the current action. For visualization purposes, we annotate the mask as red, as shown in Figure 4.

This "Mask" approach reduces reliance on background cues and yields a more scalable leader-follower policy than "RGB", as shown in Table 1. However, it is not completely immune to perception errors: in one unseen maneuver, the UNet confuses the leading UMX with a bright window pattern (similar white stripes), producing an incorrect mask and steering the follower toward the background window, as shown in Figure 6.

5.3 Vision Controller: RGB + Mask

To further mitigate such perception failures that causes downstream control problems, we introduce "RGB+Mask," which stacks the predicted binary mask as a fourth channel on top of the RGB image and repeats the imitation-learning procedure. The additional appearance context helps both disambiguate false positives and reduces overfitting, improving both SR and ATE across both training and unseen maneuvers (Table 1). The trade-off is increased inference time due to two sequential networks (mask prediction followed by a 4-channel ResNet), which makes hardware deployment at runtime risky.

5.4 Domain Randomization

Beyond pose and camera sampling, we apply domain randomization to mask detection to enhance robustness and reduce sim-to-real gaps. Specifically, we perturb the leader UMX's gaussians (Section 4.1) by varying leader's scale and color via injecting noise into colors associated with the leader's Gaussians. Figure 8 illustrates the three perturbations (brightness, salt-pepper noise and scaling) used during training.

5.5 Experiment Setup & Tracking Performance Analysis

For safety, all leader-follower experiments are conducted in simulation, and the vision policy is ran on a 4090; sim-to-real validation of FalconWing appears in the next autonomous landing study (Section 6).

Baseline. We first evaluate a state-based follower that has access to ground-truth states of both planes and uses the same fixed-wing controller as the leader. For each of the three unseen leader maneuvers (a left-turn S-shape descent, a right-turn S-shape ascent, and a right-turn sharp climb, shown in 4), we run 10 trials with slight variations in initial conditions and report SR, ATE, and ART in Table 1. As expected, the state-based baseline achieves the best SR and lowest ATE, and its ART is negligible because the computation is basically simple tensor operations.

Direct RGB policy. The "RGB" policy closely imitates the expert on its training trajectory but scales poorly to unseen leader maneuvers. Its key advantage is runtime: ART ≈ 0.02 s, which can potentially fit a 50 Hz control loop and is therefore suitable for actual hardware deployment (Section 6).

Mask-based policy. The "Mask" variant substantially reduces overfitting by conditioning control on the predicted UMX mask and past controls, improving SR and ATE across unseen maneuvers. However, it is susceptible to perception errors and has slightly higher runtime.

RGB+Mask policy. Stacking the predicted mask as a fourth channel ("RGB+Mask") mitigates the rare mask failures while retaining the generalization benefits of the Mask approach. It delivers the strongest overall SR and ATE among learned policies, but at the cost of the highest ART, which complicates hardware deployment.

Robustness. We further probe robustness of our "RGB+Mask" approach to appearance changes using scale and salt-and-pepper perturbations applied at the Gaussian color space. As is shown in Figure 7, across 10 runs per condition, the policy remains reliable over a broad range of apparent sizes; performance degrades only when the leader is reduced to 50% of its nominal size, where detection becomes unreliable. The controller is also robust to salt-and-pepper noise injected on leader-associated Gaussians.

Table 1: Controller Performance for Leader-Follower Case Study

Scenarios	Controller Input	SR% ↑	ATE [cm] ↓	ART [s] ↓
Training	State-based	100%	72	≈ 0.00
	RGB	100%	78	**0.02**
	Mask	100%	91	0.08
	RGB+Mask	**100%**	**73**	0.13
Unseen	State-based	100%	79	≈ 0.00
	RGB	30%	102	**0.02**
	Mask	90%	139	0.08
	RGB+Mask	**100%**	**94**	0.13

6 Case Study: Vision-Based Autonomous Landing

In this section, we tackle another most challenging fixed-wing task: vision-based landing and show success sim-to-real transfer using our FalconWing platform. Landing is the fundamental of all aerial applications and requires precise perception of the runway and tight control of glide slope. Even skilled RC (radio-controlled planes) pilots typically need weeks of practice to master consistent landings. We tackle vision-based landings to mimic landing in a GPS-denied zone where no ground-truth state information is available.

In the rest of this section, we describe our vision policy for landing (Section 6.1), indoor landing setup (Section 6.2) and evaluate vision-based autonomous landing in both simulation and hardware (Section 6.3) using the analog FPV camera (selected for its lower mass relative to a digital unit, enabling higher agility).

6.1 Vision Controller: RGB Approach

We used the "RGB" approach from the previous case study for vision-based control because, as indicated by Table 1, only the "RGB" policy comfortably meets the 20Hz hardware control rates. Although "RGB" suffers from generalization issues, for landing, overfitting to the specific appearance is less problematic because the runway is usually static to the background. We therefore reuse the RGB imitation-learning setup from the leader-follower study (Section 5), but train in the simulation with the leader asset removed and the objective focused on landing.

6.2 Indoor Flying Arena Setup for Autonomous Landing

All hardware trials were conducted in an indoor arena (40m×20m×5m) equipped with a blue landing pad (13m×2m×0.1m) placed at one end, as shown in Figure 3 Right. Each trial began with a human pilot manually piloting the aircraft to an initial position approximately 20 m from the runway and 1.5 m above ground, where the landing pad becomes roughly visible to the onboard analog camera.

Upon reaching this position, control was switched to Autonomous Mode (Section 3), with the pilot instructed to immediately regain manual control if a flag was raised or unsafe behavior was observed.

6.3 Sim2Real Landing Performance

We performed 10 autonomous landing trials using our "RGB" vision-based controller in the real-world environment. Landing performance was evaluated based on two primary metrics: (1) *landing success*, defined as touchdown within the bounds of the landing pad; and (2) *Absolute Lateral Deviation (ALD)* from the runway centerline at touchdown. Given that our runway width is 2 m, deviations less than or equal to 1 m is acceptable. Because our flying arena currently lacks external motion-capture infrastructure, we assessed landing accuracy by coating the landing gear with powder and measuring the resulting touchdown marks on the runway.

For accurate simulation comparisons, we recorded the aircraft's initial Autonomous Mode engagement positions, estimated by our iGSplat model, and subsequently replayed each landing attempt in simulation with both our learned vision-based controller and the state-based expert controller, enabling direct comparison.

Results are summarized in Table 2. In simulation, both the state based and vision-based controllers landed successfully in all ten cases, with mean ALD of 0.15m and 0.37m. Hardware trials achieved eight successful landings; the two failures (Runs 3 and 10) occurred after the aircraft was handed over with a steep nose-down attitude and the analog video suffered some flicker noise. The average ALD (41cm) of the real world trials are slightly larger than in simulation, this is most likely due to the difference between dynamics estimates and difference in image rendering quality. Due to the lack of ground truth states, we could not run the state-based control in real world for comparison. However, additional simulation experiments show the vision policy can handle curved approaches and large initial offsets and different altitudes (Figure 5), but these were not flight-tested because of space constraints and bank-angle safety limits.

Table 2: Controller Performance for Landing Case Study

	Simulation				Real World	
	State-based		Vision-based		Vision-based	
Run #	Success?	ALD (cm) ↓	Success?	ALD (cm) ↓	Success?	ALD (cm) ↓
1	✓	42	✓	54	✓	35
2	✓	1	✓	4	✓	45
3	✓	14	✓	38	✗	N/A
4	✓	1	✓	12	✓	40
5	✓	27	✓	32	✓	40
6	✓	37	✓	74	✓	70
7	✓	13	✓	21	✓	20
8	✓	14	✓	79	✓	80
9	✓	2	✓	32	✓	78
10	✓	2	✓	23	✗	N/A

7 Conclusion

We introduced *FalconWing*, an open-source platform for indoor, vision-based fixed-wing autonomy. FalconWing integrates a lightweight (150g) hardware stack with a world-model suite comprising photorealistic GSplat simulation and system-identified aircraft dynamics, scheduled to release upon publication. We validate FalconWing on two challenging aerial tasks without IMU or motion capture. In leader-follower visual tracking, de-coupled perception and control as well as domain-randomized GSplat training enable vision policies to generalize to unseen maneuvers and visual perturbations. In autonomous landing, an RGB-based policy trained purely in simulation transfers zero-shot to hardware, achieving an 80% success rate over ten indoor trials without fine-tuning. Future work includes: (i) identify richer dynamics models that incorporating wind disturbances, flap/drag effects, and ground effect to narrow residual sim-to-real gaps; (ii) more challenging scenarios: controlled wind/lighting changes, and temporary occlusions in leader-follow visual tracking, sharper landing approaches.

APPENDIX

Standard lift equation is $L = \frac{\rho V^2 SC}{2} \geq mg$ and coordinated-turn relations is $R = \frac{V^2}{gtan(\phi)}$, assume g=9.8, UMX wing area S=0.076 m^2, air density ρ=1.3 $\mathrm{kg/m}^3$, UMX lift coefficient C=0.6, and bank angle $\frac{\pi}{6}$.

Figure 5: Visualization for the autonomous landing case study: **Left** figure shows the gsplat-based simulation onboard view. **Middle** shows the real-world onboard view. **Right** shows different landing trajectories.

Figure 6: An example of failed detection where perception confuses leading aircraft with the background window using "Mask" approach in Section 5.2 that leads to downstream tracking error.

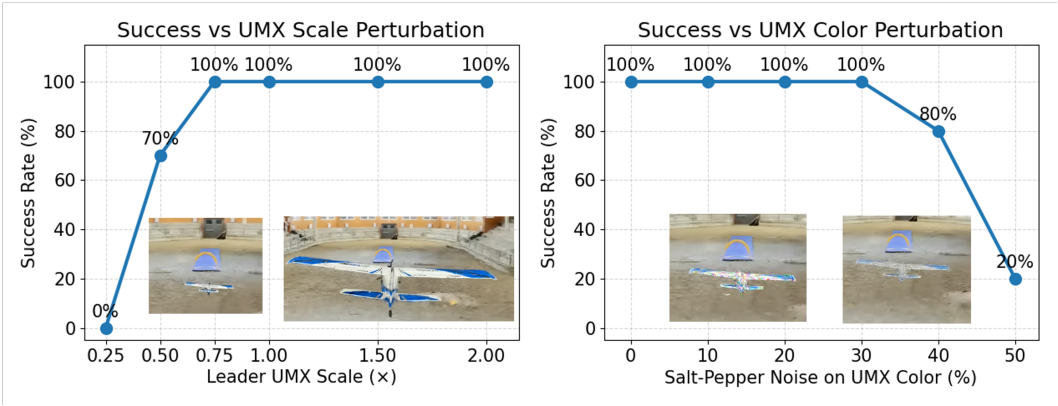

Figure 7: Ablation Studies showing that our "RGB+Mask" vision-policy for Leader-follower tracking is relatively robust to both leader scale and color change in the 3D gaussian space. This shows the potential for a robust sim-to-real transfer.

Figure 8: During training, we enable domain randomization in terms of leader gaussians' color and scale to improve training and minimize the sim-to-real gap.

References

[1] Evan Ackerman and Michael Koziol. "The blood is here: Zipline's medical delivery drones are changing the game in Rwanda". In: *IEEE Spectrum* 56.5 (2019), pp. 24–31. DOI: 10.1109/MSPEC.2019.8701196.

[2] Valentin Wüest et al. "Accurate Vision-based Flight with Fixed-Wing Drones". In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2022.

[3] Ezra Tal and Sertac Karaman. "Global Incremental Flight Control for Agile Maneuvering of a Tailsitter Flying Wing". In: *Journal of Guidance, Control, and Dynamics* 45.12 (2022), pp. 2332–2349. DOI: 10.2514/1.G006645. eprint: https://doi.org/10.2514/1.G006645. URL: https://doi.org/10.2514/1.G006645.

[4] Marinus A Boon, Albert P Drijfhout, and Solomon Tesfamichael. "Comparison of a fixed-wing and multi-rotor UAV for environmental mapping applications: A case study". In: *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 42 (2017), pp. 47–54.

[5] Yan Miao, Will Shen, and Sayan Mitra. *Zero-Shot Sim-to-Real Visual Quadrotor Control with Hard Constraints*. 2025. arXiv: 2503.02198 [cs.RO]. URL: https://arxiv.org/abs/2503.02198.

[6] JunEn Low et al. "SOUS VIDE: Cooking Visual Drone Navigation Policies in a Gaussian Splatting Vacuum". In: *IEEE Robotics and Automation Letters (under review)* (2024). Available on arXiv: https://arxiv.org/abs/2412.16346. arXiv: 2412.16346 [cs.RO].

[7] Abu Bakar et al. "Design of Low Altitude Long Endurance Solar-Powered UAV Using Genetic Algorithm". In: *Aerospace* 8.8 (2021). ISSN: 2226-4310. DOI: 10.3390/aerospace8080228. URL: https://www.mdpi.com/2226-4310/8/8/228.

[8] Azarakhsh Keipour et al. "Visual Servoing Approach to Autonomous UAV Landing on a Moving Vehicle". In: *Sensors* 22.17 (2022). ISSN: 1424-8220. DOI: 10.3390/s22176549. URL: https://www.mdpi.com/1424-8220/22/17/6549.

[9] Junjie Chen et al. "Identification of autonomous landing sign for unmanned aerial vehicle based on faster regions with convolutional neural network". In: *2017 Chinese Automation Congress (CAC)*. 2017, pp. 2109–2114. DOI: 10.1109/CAC.2017.8243120.

[10] Noi Quang Truong et al. "SlimDeblurGAN-Based Motion Deblurring and Marker Detection for Autonomous Drone Landing". In: *Sensors* 20.14 (2020). ISSN: 1424-8220. DOI: 10.3390/s20143918. URL: https://www.mdpi.com/1424-8220/20/14/3918.

[11] Sungsik Huh and David Hyunchul Shim. "A Vision-Based Automatic Landing Method for Fixed-Wing UAVs". In: *J. Intell. Robotics Syst.* 57.1–4 (Jan. 2010), 217–231. ISSN: 0921-0296. DOI: 10.1007/s10846-009-9382-2. URL: https://doi.org/10.1007/s10846-009-9382-2.

[12] Blake Barber, Timothy McLain, and Barrett Edwards. "Vision-based landing of fixed-wing miniature air vehicles". In: *Journal of Aerospace Computing, Information, and Communication* 6.3 (2009), pp. 207–226.

[13] H. Jin Kim et al. "Fully Autonomous Vision-Based Net-Recovery Landing System for a Fixed-Wing UAV". In: *IEEE/ASME Transactions on Mechatronics* 18.4 (2013), pp. 1320–1333. DOI: 10.1109/TMECH.2013.2247411.

[14] Bernhard Kerbl et al. “3D Gaussian Splatting for Real-Time Radiance Field Rendering”. In: *ACM Transactions on Graphics* 42.4 (2023). URL: https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/.

[15] Jingcheng Ni et al. *Efficient Interactive 3D Multi-Object Removal*. 2025. arXiv: 2501.17636 [cs.CV]. URL: https://arxiv.org/abs/2501.17636.

[16] Lin Yen-Chen et al. “iNeRF: Inverting Neural Radiance Fields for Pose Estimation”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2021.

[17] Matteo Bortolon et al. “6DGS: 6D Pose Estimation from a Single Image and a 3D Gaussian Splatting Model”. In: *ECCV*. 2024.

[18] Michal Adamkiewicz et al. “Vision-Only Robot Navigation in a Neural Radiance World”. In: *IEEE Robotics and Automation Letters* 7.2 (2022), pp. 4606–4613. DOI: 10.1109/LRA.2022.3150497.

[19] Arunkumar Byravan et al. “NeRF2Real: Sim2real Transfer of Vision-guided Bipedal Motion Skills using Neural Radiance Fields”. In: *2023 IEEE International Conference on Robotics and Automation (ICRA)* (2022), pp. 9362–9369. URL: https://api.semanticscholar.org/CorpusID:252815541.

[20] Marcel Torne et al. “Reconciling Reality Through Simulation: A Real-to-Sim-to-Real Approach for Robust Manipulation”. In: *Arxiv* (2024).

[21] Zhou Wang et al. “Image quality assessment: from error visibility to structural similarity”. In: *IEEE Transactions on Image Processing* 13.4 (2004), pp. 600–612. DOI: 10.1109/TIP.2003.819861.

[22] Ben Mildenhall et al. “NeRF: representing scenes as neural radiance fields for view synthesis”. In: *Commun. ACM* 65.1 (Dec. 2021), 99–106. ISSN: 0001-0782. DOI: 10.1145/3503250. URL: https://doi.org/10.1145/3503250.

[23] Johannes L. Schönberger and Jan-Michael Frahm. “Structure-from-Motion Revisited”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 4104–4113. DOI: 10.1109/CVPR.2016.445.

[24] W. Kabsch. “A discussion of the solution for the best rotation to relate two sets of vectors”. In: *Acta Crystallographica Section A* 34.5 (Sept. 1976), pp. 827–828. DOI: 10.1107/S0567739478001680. URL: http://dx.doi.org/10.1107/S0567739478001680.

[25] Matthew Tancik et al. “Nerfstudio: A Modular Framework for Neural Radiance Field Development”. In: *ACM SIGGRAPH 2023 Conference Proceedings*. SIGGRAPH ’23. 2023.

[26] “The Kinematics and Dynamics of Aircraft Motion”. In: *Aircraft Control and Simulation: Dynamics, Controls Design, and Autonomous Systems*. John Wiley & Sons, Ltd, 2015. Chap. 1, pp. 1–62. ISBN: 9781119174882. DOI: https://doi.org/10.1002/9781119174882.ch1. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9781119174882.ch1. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119174882.ch1.

[27] Alexey Dosovitskiy et al. “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”. In: *International Conference on Learning Representations*. 2021. URL: https://openreview.net/forum?id=YicbFdNTTy.

[28] Tal Ridnik et al. “ImageNet-21K Pretraining for the Masses”. In: *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*. 2021.

[29] “Aircraft Dynamics and Classical Control Design”. In: *Aircraft Control and Simulation: Dynamics, Controls Design, and Autonomous Systems*. John Wiley & Sons, Ltd, 2015. Chap. 4, pp. 250–376. ISBN: 9781119174882. DOI: https://doi.org/10.1002/9781119174882.ch4.

[30] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. “U-Net: Convolutional Networks for Biomedical Image Segmentation”. In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*. Ed. by Nassir Navab et al. Cham: Springer International Publishing, 2015, pp. 234–241. ISBN: 978-3-319-24574-4.