IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

On Your Own: Pro-level Autonomous Drone Racing in Uninstrumented Arenas

Michael Bosello†, Flavio Pinzarrone†, Sara Kiade†, Davide Aguiari†, Yvo Keuter†, Aaesha AlShehhi†, Gyordan Caminati†, Kei Long Wong†¶§, Ka Seng Chou†¶§, Junaid Halepota†, Fares Alneyadi†, Jacopo Panerati, and Giovanni Pau†§

***Abstract*— Drone technology is proliferating in many industries, including agriculture, logistics, defense, infrastructure, and environmental monitoring. Vision-based autonomy is one of its key enablers, particularly for real-world applications. This is essential for operating in novel, unstructured environments where traditional navigation methods may be unavailable. Autonomous drone racing has become the *de facto* benchmark for such systems. State-of-the-art research has shown that autonomous systems can surpass human-level performance in racing arenas. However, the direct applicability to commercial and field operations is still limited, as current systems are often trained and evaluated in highly controlled environments. In our contribution, the system's capabilities are analyzed within a controlled environment—where external tracking is available for ground-truth comparison—but also demonstrated in a challenging, uninstrumented environment—where ground-truth measurements were never available. We show that our approach can match the performance of professional human pilots in both scenarios. We also publicly release the data from flights carried out by a world-class human pilot: github.com/tii-racing/drone-racing-dataset.**

Video: youtube.com/watch?v=SNw-zXgv_vA

I. Introduction

Drone technology can now be found in a wide range of industries, such as agriculture, the logistics and delivery sectors, defense, infrastructure inspection, and environmental monitoring. For all these applications, drones can improve efficiency and operational safety.

The development of vision-based state estimation and control, in particular, is at the core of the deployment of autonomous drone systems in real-world applications. Vision-based autonomy is essential for operating in novel, unstructured, and uninstrumented environments, where traditional navigation methods may be unreliable or unavailable [1].

The last two decades of rapid progress in AI have seen machine learning systems challenge, and even defeat, humans in several tasks (from image recognition to the game of Go, to self-driving). Since the 2019 AlphaPilot AI Drone Innovation Challenge [2], [3], drone racing has emerged as the *de facto* benchmark for evaluating vision-based aerial autonomy. In 2023, state-of-the-art research has demonstrated that autonomous racing drones can surpass human-level performance in controlled racing arenas [4]. However, the applicability of fast, vision-based autonomous flight to practical commercial operations is still limited, as current systems are typically trained and evaluated in highly controlled (often externally instrumented) environments.

†Autonomous Robotics Research Center, Technology Innovation Institute, Abu Dhabi, UAE. E-mails: {firstname.lastname}@tii.ae
¶Macao Polytechnic University, Macao SAR, China.
§University of Bologna, Bologna, Italy.

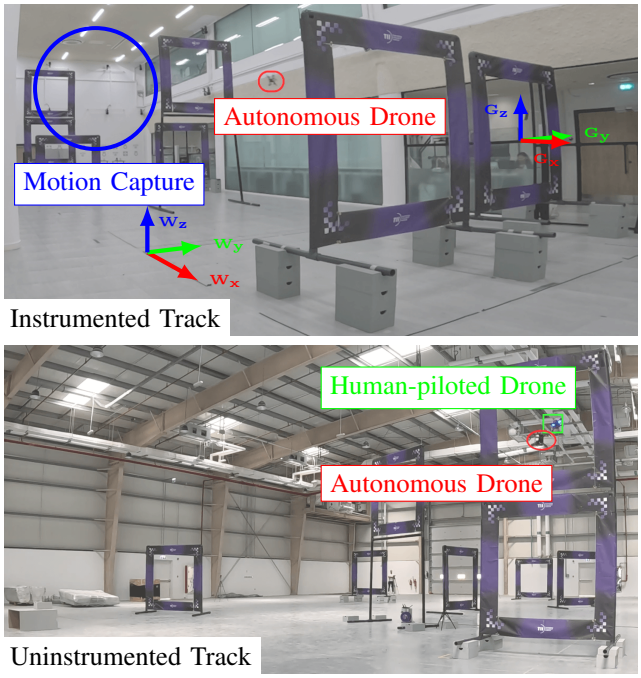


Fig. 1: The two arenas used during the experiments, showing the autonomous drone (red) performing a lap in the instrumented one (top), and a head-to-head versus a human pilot (green) in the uninstrumented track (bottom).

The main contributions of this work are as follows.

- Achieving pro-level autonomous drone racing in both controlled environments—where external tracking is available—as well as in equally challenging, uninstrumented environments—where ground-truth measurements were never available (see Fig. 1).
- Presenting a perception and control stack for autonomous drone racing that does not require fine-tuning with ground truth for residual estimation (and has proven to be resilient to multiple lighting conditions).
- Releasing pro-level piloting data from the instrumented track using the same format as in [5], adding six new flights by a world champion pilot (Sec. VI-D).

II. Related Work

Autonomous drone racing [1] (and the related discipline of autonomous, or self-driving, car racing) is an emerging racing sport characterized by fundamental research and tech-

TABLE I: Comparison of "human v. robot" drone racing literature

Ref.	Year	Vehicle Type	Competition / Contribution	External Sensing		Human Best Speed / Lap	Robot Best Speed / Lap	Head-to-head Winner
				On Track	In Race			
[2]	2019	Quadcopter	AlphaPilot	✗	✗	n/a n/a	9.19m/s 12.00s	n/a
[4]	2023	Quadcopter	Research	✓	✗	21.54m/s 5.19s†	19.44m/s 5.11s†	Robot
[5]	2024	Quadcopter	Dataset	✓	✓	9.58m/s 6.04s	21.83m/s 3.20s	Robot
This	2025	Quadcopter	Research	✗	✗	25.63m/s 5.04s	21.15m/s 5.60s	Human

†Computed from the MoCap recordings released in the extended data of [4], filtered by frequency to account for missing MoCap frames. Speed is computed as the derivative of position and smoothed using an exponentially weighted mean with $\alpha = 0.3$.

nological challenges, in particular the need for fast and robust perception, planning, and control.

In car racing, since the 2004/2005 DARPA Grand Challenges, various leagues and research challenges have been established, including the Indy Autonomous Challenge (IAC) and the Abu Dhabi Autonomous Racing League (A2RL)—the latter organizing "Man vs Machine" events for both drone and car racing [6]—aimed at accelerating progress in performance, research, and entertainment.

In drone racing, competitions like the AlphaPilot AI Drone Innovation Challenge [3] marked significant milestones, with the best teams [2] showing important advances in the use of deep learning for gate detection, Visual-Inertial Odometry (VIO), and dynamics modeling.

At the core of the control loop are often model predictive techniques (like Model Predictive Contouring Control (MPCC) [7] or Perception-Aware Model Predictive Control (PAMPC) [8]) that explicitly consider dynamics and sensor limitations, enabling high-speed, agile flight.

Nonetheless, the latest research developments showed that the use of deep reinforcement learning for control could also make autonomous racing drones capable of outperforming human champion pilots [4], [9], [10]. Table I summarizes the human-piloted and autonomous performance recorded in this work alongside those published in similar recent work.

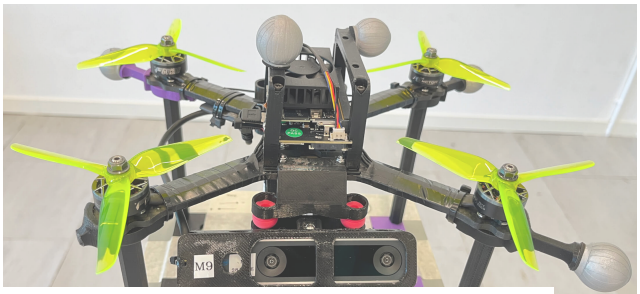
Beyond competitions, research progress is also facilitated by benchmarks and datasets [5], which typically include visual, inertial, and motion capture data from aggressive flights, crucial for developing novel methods and enabling quantitative comparisons.

Still open research questions include: safely and effectively managing mixed human-robot multi-vehicle racing, reliable state estimation at extreme speeds, coordinating multiple autonomous racers, ensuring safety, and improving sim-to-real algorithm transfer.

Insights from autonomous high-speed flight demonstrations, such as this one, and the analysis of pro-level human pilot data, like those released alongside this article, can offer valuable cues for future research.

III. Drone Platform

Racing drones require a combination of agility, durability, and power. In addition to that (and unlike human-piloted drones), autonomy also demands careful integration of onboard compute with high-performance sensors.



Autonomous Drone – Instrumented Track Variant

Piloted Drone – Uninstrumented Track Variant

Fig. 2: Quadrotors used in the experiments: the autonomous drone (including markers for the instrumented track, top); and the piloted drone (with replicas of the autonomy components, without markers for the uninstrumented track, bottom).

The components listed in this section result in an autonomous drone capable of speeds over 25m/s along an agile indoor trajectory and weighing just 665.5g (without battery).

A. Hardware

The hardware components are partially based on the open-design from [5], these include the Flight Controller (FC), motors, electronic speed controller, and battery eliminator circuit. However, the frame and 3D-printed parts have been modified to accommodate the stereo-camera, adopting the same frame (Armattan Chameleon Ti 6") as in [11].

1) Motors and Propellers: The quadrotor features the T-motor F60 PRO V 2020KV motors, as in [5], paired with HQProp HeadsUp R38 propellers to accommodate the professional pilots' unanimous preference for a lower blade pitch. This setup results in a Thrust-to-Weight Ratio (TWR)

of $\sim$7 at full battery capacity, measured using a thrust bench.

2) Camera and Sensors: The platform is equipped with an Intel RealSense T265 stereo-camera, which captures fisheye grayscale images at 30Hz per lens and offers VIO at 200Hz. A custom damping mechanism (Fig. 2) minimizes image blur, improving VIO accuracy during aggressive flight. Additionally, an InvenSense MPU6000 IMU embedded in the FC provides high-frequency acceleration and gyroscopic data.

3) Compute: As a companion board, we use the NVIDIA Orin NX module, installed on the A603 carrier board by Seeedstudio. The Orin NX module comes with JetPack 5.1.2 (i.e., Linux Kernel 5.10, Ubuntu 20.04-based root file system, and CUDA 11.4 support). We select the *MAXN* power mode to maximize core usage and clock frequencies.

4) FPV Replica: Drone replicas of the autonomous model were prepared for the pilots. These replicas were outfitted with an HDZero FPV system (camera, transmitter, and antenna). The autonomy components (i.e., NVIDIA computer and Intel camera) were replaced by 3D-printed replicas filled with lead. This modification ensured that the piloted drones matched the weight and weight distribution of the autonomous one while minimizing the risk of damaging sensitive components. The autonomous and piloted drones are shown in Fig. 2.

B. Software

1) Autopilot: Our drone requires low-latency, real-time reactive control. This is managed by the FC, running Betaflight (BF) 4.3.2 firmware tuned for the platform [12]. The FC is interfaced using the MultiWii Serial Protocol (MSP), enabling efficient bidirectional data exchange between the FC and the onboard computer. This allows for IMU readings at a rate up to 500Hz over a 1MBaud serial connection (much more than the 10Hz obtainable with the SBUS protocol, as reported in [11]). Additionally, the MSP Override feature provides the onboard computer with complete control over the flight channels. A real-time Linux kernel setup—with priority granted to the MSP process—was crucial to achieve the performance reported in this letter.

2) Vison and Autonomy: Our platform uses a hybrid autonomy strategy, integrating *(i) ad hoc* perception and control modules (see Sec. IV), implemented in the Robot Operating System 2 (ROS2), with *(ii)* Intel's proprietary RealSense SDK (`librealsense2`), which provides an estimate of the camera position during the flight, albeit prone to significant drift even over very short time.

IV. Autonomous Drone Racing

The autonomy stack has three main components: vision, state estimation, and control. The vision module provides gate corner detections. The state estimation uses those detections to correct the drift of the Intel T265 VIO, and the FC IMU readings for refining the state. Given the current state estimate and a time-optimal reference trajectory, the control block computes the command to be executed. A schematic of the stack is presented in Fig. 3.

A. Vision Stack

The vision stack detects gates using grayscale images captured from the stereo-camera. For this, only the image frame from the right camera is used, as its position is closer to the drone's center. Every camera is calibrated with the fisheye model to obtain its intrinsics.

A gate is represented by its four inner corners. The pipeline includes two steps—gate detection and corner detection—each using a separate Convolutional Neural Network (CNN)-based model. All models are converted into ONNX graphs with opset v17, and then into TensorRT (v8.5) engines, with half-precision floating-point (FP16), to exploit the Orin NX's Ampere GPU. The measured per-frame latency of the detection stack is 24–30ms, varying with the number of detected gates and the instantaneous CPU load. Fig. 4 illustrates the training procedure and detection flow of the vision stack.

For gate detection, we use You Only Look Once (YOLO), specifically, the YOLOv8n model with 3.2 million parameters, one class (gates), and input size (640×640).

For corner detection, the pixels corresponding to each detected gate region are cropped and forwarded to a custom key-points detection model, MobileNetV3-Small [14] with 1.1 million parameters. The input size of the model is (256×256), and its weights are initially pretrained for a classification task using the ImageNet-1K dataset.

Following the pretraining phase, only the feature extraction layers are kept, while the classification head is replaced with a key-point head composed of five CNN layers. The first three layers employ scaling factors of $[4, 4, 2]$, with convolutional filters of $[256, 128, 64, 32, 4]$.

The model produces heatmaps representing the confidence of four key-point coordinates, yielding an output shape of $(N, 4, 256, 256)$, where N denotes the batch size (or the number of detectable gates at inference time for parallelization). The four channels represent the four key-points, respectively.

1) Data Collection and Training Procedure: The training process consists of fine-tuning pretrained models for both tracks. Gate and corner detection models were pretrained on the grayscale version of the dataset in [5] and fine-tuned on undistorted images using the calibrated intrinsics.

The instrumented track fine-tuning dataset comprises 3,412 frames from autonomous flights. These are initially labeled by the aforementioned pretrained models, then manually corrected using the annotation software CVAT[1].

For the uninstrumented track, we use model distillation to make up for the lack of existing labeled data. A foundational model, Grounding DINO [15], is fine-tuned using 80 manually labeled frames from human-piloted flights, then used to generate gate detection labels for the entire flights, which are used to fine-tune YOLO. For corner detection, the instrumented track model is applied for auto-labeling, with frames exhibiting detection errors manually corrected for fine-tuning.

B. State Estimation Stack

Our state estimation stack leverages VIO from the Intel T265. Although VIO provides highly accurate short-term estimates, its performance degrades over time, leading to significant drift, especially during fast maneuvers that introduce

[1]https://www.cvat.ai/

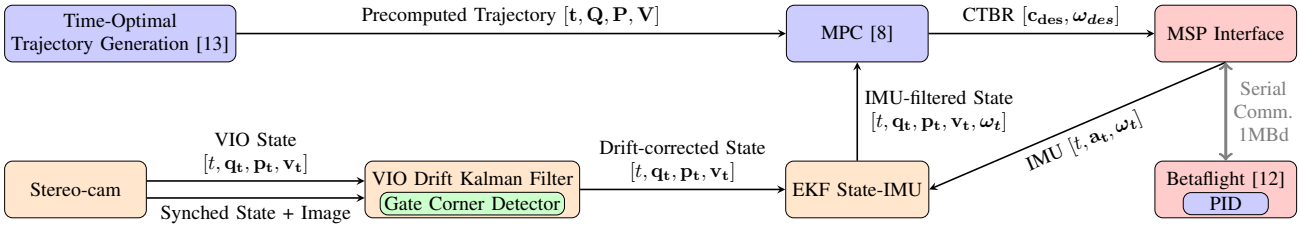


Fig. 3: Architecture diagram illustrating the data flow between the main hardware and software components, including: *vision* (green, Sec. IV-A); *state estimation* (orange, Sec. IV-B); *control* (blue, Sec. IV-C); and *autopilot* (red, Sec. III-B.1). Unlike [4], we introduce a second-stage EKF leveraging the high-frequency FC IMU, which produces a sufficiently smooth state estimate to enable the use of classical control and MPC. High-frequency readings (500 Hz) from Betaflight are supported by our optimized MSP setup (Sec. III-B), offering higher performance than the SBUS protocol adopted in [11].

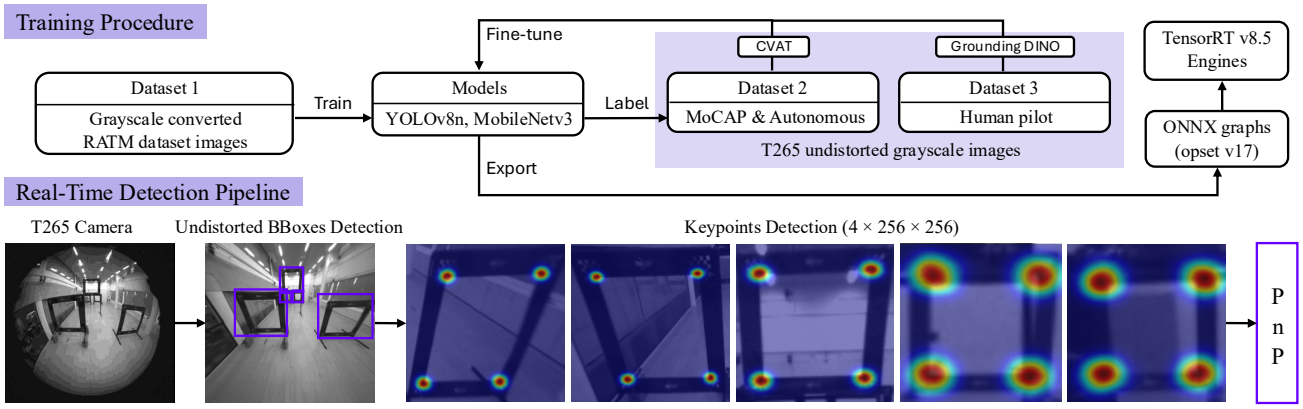


Fig. 4: Vision stack schematics illustrating the training procedure (top) and the detection pipeline (bottom). The training procedure comprises three steps (designed to minimize human supervision while preserving accuracy): *(i)* training on the RATM dataset [5]; *(ii)* auto-labeling, followed by human refinement in CVAT; and *(iii)* labeling track images with Grounding DINO from human examples. By the final stage, fine-tuning for peak performance in new environments requires as few as 80 human-corrected frames. In the detection pipeline, YOLOv8n detects gate bounding boxes, while MobileNetV3-Small [14] estimates their keypoints. Leveraging GPU acceleration (see Sec. IV-A), the stack achieves per-frame latency under 30 ms on 640×640 images. This yields lower delay and larger input resolution compared to [4], resulting in higher pixel-level precision and potentially improved PnP accuracy.

motion blur. To mitigate this drift, we incorporate detections of the only predefined landmarks in drone racing: the gates. By identifying the corners of the gates in the captured images, the drone's pose relative to the gate is computed by solving the Perspective-n-Point (PnP) problem, of which only the positional component is used. The global position of the drone is then refined using the gate's known location in the track layout, whereas the attitude is left uncorrected, as it exhibits a substantially lower drift over time.

Using fixed exposure and carefully tuning the exposure time and gain, we obtain good image brightness with minimal motion blur, ensuring reliable VIO. We enable "mapping", allowing the device to build and update an internal map for loop closure and small drift correction. In contrast, we disable "relocalization", which attempts to re-align the estimate after large drifts, and "pose jumping", which can cause discontinuous estimates. This configuration provided the best balance of VIO accuracy and reliability, while still allowing us to correct larger drifts using gate measurements. The right camera operates at 30Hz, and its frames are synchronized with the corresponding VIO estimates.

Our drift correction strategy starts with the detection of all visible gates in the undistorted image (see Sec. IV-A and Fig. 4). Then, we solve a PnP problem for each detected gate with four visible corners using OpenCV's `SOLVEPNP_ITERATIVE`, which initializes the solution using homography decomposition, as the gates are planar, and refines it with a nonlinear Levenberg-Marquardt minimization scheme. Using these gate measurements and the known track map, we set up a Kalman Filter similar to the one presented in [4] to estimate the translational drift of the VIO. In our case, the state vector is $\mathbf{x} = \mathbf{p}_d^T \in \mathbb{R}^3$, representing the positional drift vector. Unlike [4], we do not estimate the drift velocity as it has a less stable behaviour when loop closure is performed at the camera firmware level. Therefore, the state $\mathbf{x}$ and covariance $\mathbf{P}$ are propagated according to:

$$\mathbf{x}_{k+1} = F\mathbf{x}_k \qquad P_{k+1} = FP_kF^T + Q \tag{1}$$

$$F = \mathbb{I}^{3\times3} \qquad Q = \mathbb{I}^{3\times3}\frac{1}{4}\mathbf{d}t^4\sigma_a^2 \tag{2}$$

The filter state and covariance are initialized to zero, while the process noise is set to $\sigma_a^2 = 8$. The filter is then updated whenever a new measurement $\mathbf{z}_k$ (a position estimate from a gate detection) is available, using the Kalman filter equations:

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + R)^{-1} \tag{3}$$

$$\mathbf{x}_k^+ = \mathbf{x}_k^- + K_k(\mathbf{z}_k - H(\mathbf{x}_k^-)) \tag{4}$$

$$P_k^+ = (I - K_k H_k)P_k^- \tag{5}$$

in which K_k is the Kalman gain, R is the measurement covariance matrix, estimated via Monte-Carlo sampling as described in [4], and H_k is the measurement matrix. When several gates are detected in a single camera frame, all relative pose estimates are stacked and processed in the same Kalman filter update step as multiple simultaneous measurements. The resulting drift estimate is constantly used to correct the VIO position estimate. As a final step, the state estimate is refined by fusing it with IMU data from the FC, which operates at 500Hz. This fusion is performed using an Extended Kalman Filter (EKF) with state vector $\mathbf{x} = [\mathbf{q}^T, \mathbf{p}^T, \mathbf{v}^T, \mathbf{b}_\omega^T, \mathbf{b}_a^T]^T \in \mathbb{R}^{16}$ and error state vector $\mathbf{e} = [\psi, \theta, \phi, \mathbf{p}^T]^T \in \mathbb{R}^6$, in which $\mathbf{q}$ represents the drone's orientation quaternion and ψ, θ, ϕ its Euler angles representation, $\mathbf{p}$ is its position, $\mathbf{v}$ is its linear velocity, $\mathbf{b}_\omega$ is the bias of the gyroscope and $\mathbf{b}_a$ is the bias of the accelerometer. The propagation step follows the equations:

$$\mathbf{x}_{k+1} = F\mathbf{x}_k \qquad P_{k+1} = FP_kF^T + WQW^T \tag{6}$$

Where $\mathbf{P}$ is the state covariance matrix, $\mathbf{F}$ is the state transition matrix, and $\mathbf{W}$ is the Jacobian of the error-state transition model with respect to process noise. The update step is then defined as in 3, 4, 5, but in this case the measurement is $\mathbf{z}_k = [\psi, \theta, \phi, \mathbf{p}^T]^T \in \mathbb{R}^6$, representing attitude and gate-corrected position estimates from VIO, and $\mathbf{R}$ is a static diagonal covariance matrix.

C. Control Stack

Our control stack couples Model Predictive Control (MPC) with a time-optimal trajectory generator.

1) Time Optimal Trajectory Generation: An open-source time-optimal trajectory generator [13] was utilized to create reference trajectories for both tracks. The generator calculates trajectories minimizing the time required to reach specified waypoints, taking into account the full rigid-body drone dynamics and actuator constraints. The linear aerodynamic drag coefficients were not included.

Each gate was assigned two waypoints, positioned at the center of the gate along the y- and z-axes. Along the x-axis, the waypoints were placed at -0.4m and +0.4m (-0.4m and +1.25m for the Split-S). This displacement is defined in gate-frame, and it is transformed in world-frame using the known gate yaw angle (in radians) specified in Fig. 5. The world- and gate-reference frame are shown in Fig. 1.

$$\begin{cases} x_{wp} = x_{gate} \pm 0.4 \times \cos(\theta_{gate}) \\ y_{wp} = y_{gate} \pm 0.4 \times \sin(\theta_{gate}) \end{cases} \tag{7}$$

We center the waypoints in the y- and z-axes for caution and use double waypoints along the x-axis to ensure that the optimized trajectory does not cut through the gate banners. To ensure the feasibility of the trajectories, we set a (conservative) TWR ratio of 3.8 in the generation parameters to mitigate potential modeling errors.

2) MPC Problem: The open-source MPC framework from [8] served as the foundation of our implementation. However, the perception-aware objectives were disabled, and the remaining cost weights were carefully tuned, prioritizing precise trajectory tracking and robustness against noisy

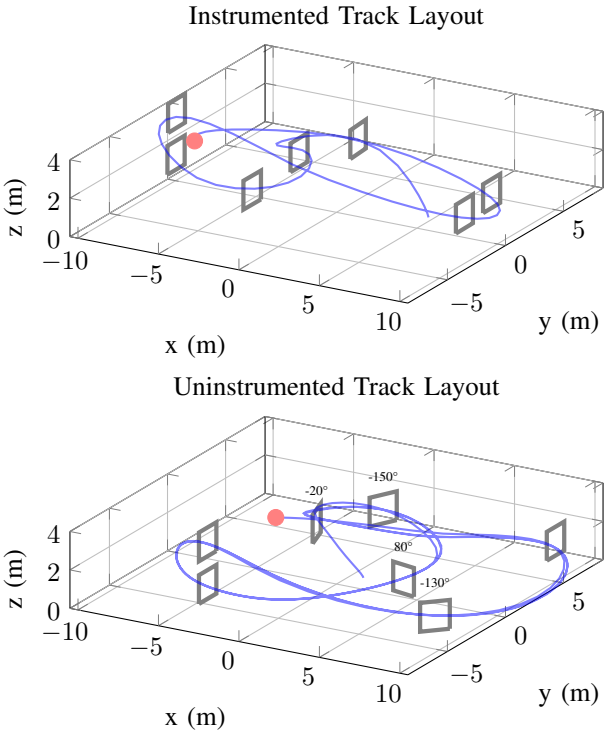


Fig. 5: Gate setup for the instrumented (Sec. V-A, top) and uninstrumented (Sec. V-B, bottom) tracks, including starting point (in red), and the time-optimal reference trajectories (in blue). Yaw is reported for gates with non-zero yaw.

state estimation. To compensate for the command delay—including MPC computation, FC communication, and motor actuation delay—we incorporated a state predictor. This predictor estimates the current state based on point-mass model dynamics and previous commands, compensating for delays before passing the information to the MPC controller, which then computes the optimal control inputs. This modification was crucial to ensure proper trajectory execution.

3) Betaflight PID: The MPC setpoints are defined as Collective Thrust and Body Rates (CTBR), which must be converted into PWM rotor signals to control the drone. This conversion is handled by the internal PID controllers of BF, which utilize gyro feedback to achieve the desired setpoints. The CTBR commands can be directly mapped to BF channels and transmitted to the FC. The PID gains in BF were tuned to optimize the drone's autonomous performance.

V. Racing Tracks

For our experiments, we used two different tracks (Fig. 5). The first track was set up in a flight arena equipped with a motion capture (MoCap) system, providing ground-truth data to analyze speed, gate-to-gate times, traveled distance, state estimation, and trajectory tracking. The second track was prepared in a larger, uninstrumented hangar. In this case, quantitative evaluation was limited to lap and sector times.

The tracks were created using gates made of PVC pipes and covered with fabric banners featuring printed designs. Each gate measures 7-by-7ft (213.36cm) with an inner opening of 5-by-5ft (152.4cm), matching closely the standards set by prominent drone racing leagues[2].

[2]https://www.multigp.com/product/drone-racing-gate-bundle

A. Instrumented Track

1) Technical Specifications: The *Track RATM* from [5] was recreated in an indoor arena measuring 25 (L) by 9.7 (W) by 7 (H) meters. The MoCap system was used for the exact placement of the gates. The track features a sharp hairpin, a spiral segment, and a Split-S maneuver. The gate poses were taken from the public GitHub repository accompanying [5].

2) Motion Capture: The instrumented arena is equipped with a 32-camera Arqus A12 Qualisys MoCap system[3], capable of tracking the 6DoF poses of rigid bodies with millimeter-level precision at 275Hz. The drone presented in Sec. III-A was outfitted with five 25mm markers, three on 3D-printed arm extensions, and two near the center of the frame.

B. Uninstrumented Track

1) Technical Specifications: In the larger, uninstrumented hangar, we reproduced, as faithfully as possible, the *Track Split-S* from [4]. We positioned the gates with the support of a total station, resulting in position errors in the order of a few centimeters. The gates' orientations had errors of at most ten degrees. We should also note that the bottom gate of the Split-S in our track is 20cm lower compared to the original track due to a different gate interlocking system. The gates' placement is similar to that of *Track RATM*. Both tracks feature the Split-S maneuver and similar z-axis variations, although this track has softer curves and lacks the sharp hairpin and spiral segment. The gate poses of the original track are published in the supplementary material of [4].

VI. Instrumented Results

The experiments on the instrumented track (Sec. V-A), where the autonomy components were initially developed, were designed to first compare the performance of the autonomous drone with that of human pilots in an environment where more comprehensive data and statistics could be collected. Both the autonomous and piloted drones used in these flights were equipped with MoCap markers.

A. Participants

Three professional pilots participated in the experiment, divided into two sessions. The first session included two top-ten-ranked pro-pilots from the Drone Champions League (DCL), namely Thomas Kund (`Star23467`) and Krutharth M. C. (`Ion FPV`). The second session featured a world-champion pilot, Minchan Kim (`MCK`), who had earned multiple titles in recent years—in MultiGP Championship, FAI World Drone Racing Championship (WDRC), and Drone Racing League (DRL)—and led his team to victory in the DCL.

B. Runs

In the first session, the two professional pilots had five days to train and record data on the instrumented track. The second session took place four months later, during which the champion pilot spent a total of five days for both (instrumented and uninstrumented) experiments, dedicating three of those days to the instrumented track.

[3]https://www.qualisys.com/cameras/arqus

The pilots were encouraged to train as much as needed to familiarize themselves with the platform and track, reaching their desired peak performance before starting to record timed laps for the experiment. During the training period, they also fine-tuned the Betaflight parameters to suit their preferences. While the flight hardware was fixed, the pilots had full control over the firmware configuration. They were allowed to use their preferred transmitter, cameras, personal radio controllers, and goggles to ensure optimal performance.

The pilots were also allowed to choose how many batteries or hours of practice to use each day, as well as the number of laps to complete with each battery. The autonomous drone always ran one lap per battery instead.

C. Performance

The instrumented evaluation allowed the comprehensive analysis of multiple quantitative metrics, including lap times, average speeds, and path lengths. We recorded data for three human pilots and our autonomous system, the latter using both external state estimation (MoCap) as well as fully on-board autonomy (VIO). The results of these flights are presented in Table II and Fig. 6. The autonomous systems (MoCap and VIO) surpassed the humans in *average* lap time and top speed, although they fell short to one out of the three pilots in terms of best individual results (achieved by `MCK`). In the instrumented arena, the autonomous system showed a clear reliability advantage, completing all runs without any crash, whereas the human pilots suffered multiple incidents.

D. Dataset

Six flights by the fastest pilot, `MCK`, totaling 240.77s of flying time and 2342.98m of traveled distance, at a top speed of 21.29m/s, are added to our `drone-racing-dataset`, using the format described in [5].

VII. Uninstrumented Results

The experiments on the uninstrumented track (Sec. V-B) demonstrate the performance of our autonomous drone when deployed in a new environment where ground-truth data could not be recorded. The systems under scrutiny were the state estimation (Sec. IV-B) and control (Sec. IV-C) modules, as they are the ones that can benefit from fine-tuning using ground-truth data.

The scenario was not entirely unknown, as the basic data required for the vision stack were available. To support the vision module, images of the new location were captured and used to fine-tune the corner detection algorithm. This adjustment allowed for accommodating the significant variations in lighting conditions caused by sunlight streaming through the hangar windows (as tests had to be carried out at all hours of the day). The gate map, essential to support the state estimation module, was created using a total station. Although accurate, this method was considerably less precise than the map generated by the MoCap system.

A. Participants

This uninstrumented track experiment involved only the fastest world-class pilot, Minchan Kim (`MCK`), who competed against our autonomous drone using only on-board sensing and without ground-truth data fine-tuning ("on its own").

TABLE II: Summary of the †instrumented and ‡uninstrumented flights

Pilot	Lap Time (s)				Top Speed (m/s)				Path Length (m)				Laps	Laps / Batteries	Crashes
	Avg.	Std.	Min.	Max.	Avg.	Std.	Min.	Max.	Avg.	Std.	Min.	Max.			
†Star23467	7.71	0.89	6.76	10.74	11.90	0.97	9.13	13.8	55.68	1.74	51.65	62.60	49	6.12	2
†Ion FPV	6.51	0.60	5.70	9.04	15.83	1.57	12.12	18.51	57.37	1.93	52.27	62.85	78	6.5	7
†MCK	4.71	1.25	**3.84**	15.75	20.87	2.95	13.16	**24.96**	51.29	5.87	47.38	124.68	**196**	**6.75**	5
†*Ours* MoCap	**4.44**	**0.11**	4.39	**4.85**	**21.92**	**0.50**	**20.06**	22.28	**47.42**	**0.40**	47.15	**48.90**	21	1	**0**
†*Ours* VIO	4.65	0.22	4.40	**4.85**	20.98	1.29	19.61	22.2	48.93	1.11	**47.11**	50.01	6	1	**0**
‡MCK	**5.80**	0.40	**5.05**	6.89	–	–	–	–	–	–	–	–	**63**	**6.89**	**2**
‡*Ours* "On Its Own"	6.02	**0.06**	5.92	**6.13**	–	–	–	–	–	–	–	–	45	3	4

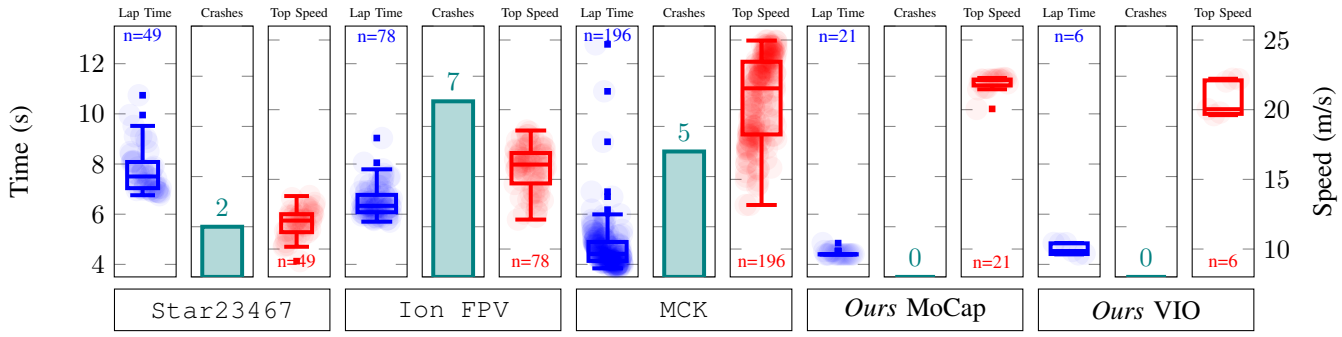


Fig. 6: Results from the instrumented track showing the lap time distributions, the number of crashes, and the top speed distributions for the three pro-level pilots (Star23467, Ion FPV, and MCK), our autonomous system supported by external sensing (*via* motion capture), and our autonomous system using only onboard sensing (VIO).

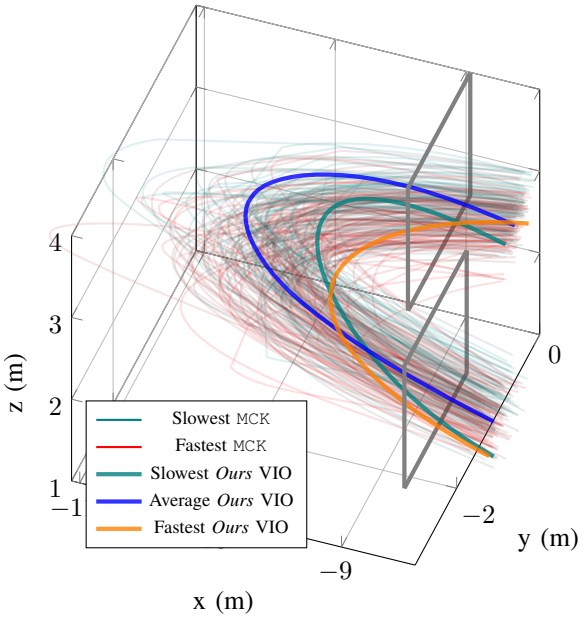


Fig. 7: Comparison of the trajectories followed during the Split-S maneuver by the fastest pilot (MCK) and our autonomous system using only onboard sensing (VIO), sorted by lap time, on the instrumented track.

B. Runs

This experiment was carried out immediately after the second session of the instrumented tests and spanned two days. It included both time trial flights and head-to-head competitions. The time trial flights were similar to the instrumented tests, although we could only record timings. In the head-to-head flights, the autonomous and piloted drones competed simultaneously over three consecutive laps of the track. The start of these races was signaled by an audio countdown. In both scenarios, our analysis was conducted on the basis of lap times. Moreover, the head-to-head flights provided additional insight into the endurance and resilience of the autonomous drone. To measure lap and sector times, we employed GoPro cameras filming at 240fps.

C. Performance

MCK achieved both a lower average and best lap time (see Table II, Fig. 9). The autonomous drone, which was 1.27% faster than MCK on average lap time in the instrumented track, gave up a 3.65% margin to the fastest human. Note that the second and third ranked pro-pilots were ∼40-60% slower than MCK in the instrumented track. MCK had 2 crashes, whereas the autonomous system crashed 4 times—3 of which were due to collisions with the human pilot, who was able to recover in two of these cases.

VIII. Discussion

State-of-the-art research has shown that autonomous drone racing can outperform professional pilots in controlled, instrumented arenas [4]. We set ourselves out to answer the question of whether such a level of performance could also be achieved in uninstrumented settings—as this will be crucial for the real-world deployment of these methods.

Our results were positive and showed pro-level autonomous performance against a champion pilot (MCK), over multiple laps (3), in an uninstrumented arena, with varying lighting conditions. We should also note that the autonomous performance, even in the uninstrumented arena, albeit not always the fastest, was always remarkably more consistent than the humans ("robots [...] are not weary").

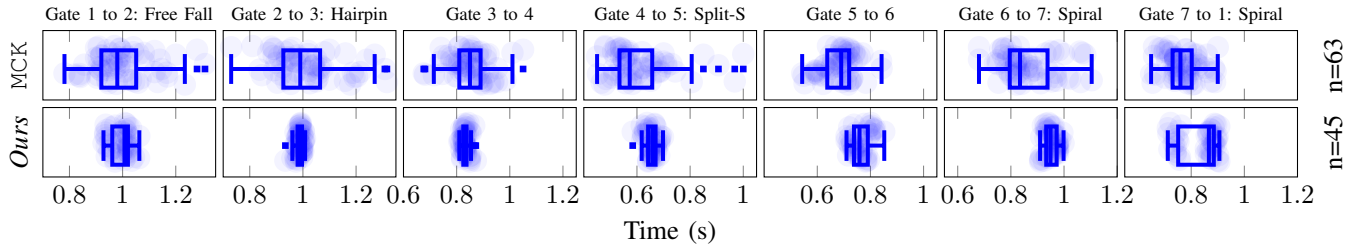


Fig. 8: Distribution of the sector times of MCK (top) and our autonomous drone (bottom) on the uninstrumented track.

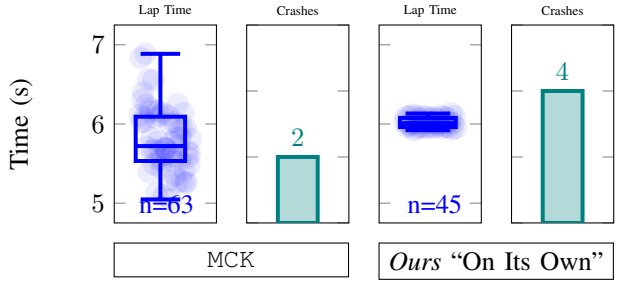


Fig. 9: Lap time distributions and crashes for the fastest pilot and our autonomous system on the uninstrumented track.

The design choices in Sec. IV and their rigorous integration were critical to achieve the demonstrated champion-level performance. Notably, the smooth state estimate produced by the dual-stage filtering (Sec. IV-B) enabled the aggressive MPC tuning required for lap-time optimization; until the introduction of such filtering, the resulting state discontinuities had led to instability and crashes. Furthermore, the state predictor used for delay compensation (Sec. IV-C.2) unlocked 3D maneuvers at more sustained speed. Together, these improvements helped us cross the boundary between achieving pro-level performance and champion-level performance, corresponding to an advantage of approximately 2 seconds on the reference tracks.

In the instrumented arena experiments (Fig. 6), we observed different critical maneuvers for the human pilots and the autonomous drone. Star23467 and Ion FPV reported the Split-S (Fig. 7) maneuver as problematic for human pilots due to the constrained physical space, as the Split-S was placed at the end of the arena, near a wall. All pilots consistently identified the spiral section of the track as a pivotal segment for performance. For the autonomous system, the hairpin turn between gates 2 and 3 was a major limiting factor, as it became a failure point when trying to execute trajectories generated with a less conservative TWR.

In the uninstrumented arena (Fig. 8, 9), the crashes in the head-to-head competition between MCK and the autonomous drone highlighted the autonomous system's consistency (even to a fault) and, on the other hand, the human pilot's greater adaptability. The autonomous system remained much more vulnerable to shared-track interactions. This type of "human v. robot" competition still offers many open research questions that will need to be answered to create robots that can be safely deployed alongside humans.

As a further demonstration of skill transferability, our autonomous system was also deployed in two additional uninstrumented venues—featuring novel race tracks—with only minimal tuning and limited on-site testing. Specifically, we only used 80 additional images for fine-tuning (Fig. 4), we made adjustments to the camera exposure and gain (Sec. IV-B), and we generated new time-optimal trajectories (Sec. IV-C.1). These deployments were showcased in public demonstrations at IROS 2024 and the 2024 Abu Dhabi F1 Grand Prix, where the autonomous drone outperformed the professional pilot Ion FPV in both time-trial and head-to-head races. Notably, the F1 track posed the additional challenge of outdoor lighting conditions. Videos of both demonstrations are included in the multimedia material.

IX. Conclusions

In this letter, we presented a drone racing autonomy stack—comprising vision, state estimation, and control modules—that is capable of pro-pilot performance in uninstrumented arenas, i.e., without access to ground-truth data to fine-tune the state estimation and control modules. In particular, we showed that our drone outperformed professional pilots in an instrumented setting and was competitive against a champion pilot in the uninstrumented one. In our future work, we will replace the stereo-camera with a monocular one (bringing our stack closer to its human counterpart).

Acknowledgments—We thank the pilots, Thomas Kund (Star23467), Krutharth M. C. (Ion FPV), and Minchan Kim (MCK), for their insight and invaluable suggestions.

References

[1] D. Hanover, *et al.*, "Autonomous drone racing: A survey," *IEEE Transactions on Robotics*, vol. 40, pp. 3044–3067, 2024.
[2] C. De Wagter, *et al.*, "The sensing, state-estimation, and control behind the winning entry to the 2019 artificial intelligence robotic racing competition," *Field Robotics*, vol. 2, pp. 1263–1290, 2022.
[3] P. Foehn, *et al.*, "Alphapilot: autonomous drone racing," *Autonomous Robots*, vol. 46, no. 1, pp. 307–320, Jan 2022.
[4] E. Kaufmann, *et al.*, "Champion-level drone racing using deep reinforcement learning," *Nature*, vol. 620, pp. 982–987, Aug 2023.
[5] M. Bosello, *et al.*, "Race against the machine: A fully-annotated, open-design dataset of autonomous and piloted high-speed flight," *IEEE Robotics and Automation Letters*, vol. 9, no. 4, pp. 3799–3806, 2024.
[6] A2RL, "A2rl soars to new heights: Launches autonomous drone racing championship worth us $1mn prize pool," [Online], 2024.
[7] A. Romero, *et al.*, "Model predictive contouring control for time-optimal quadrotor flight," *IEEE Transactions on Robotics*, vol. 38, no. 6, 2022.
[8] D. Falanga, *et al.*, "Pampc: Perception-aware model predictive control for quadrotors," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 1–8.
[9] Y. Song, *et al.*, "Reaching the limit in autonomous racing: Optimal control versus reinforcement learning," *Science Robotics*, 2023.
[10] R. Ferede, *et al.*, "One net to rule them all: Domain randomization in quadcopter racing across different platforms," *arXiv*, 2025.
[11] P. Foehn, *et al.*, "Agilicious: Open-source and open-hardware agile quadrotor for vision-based flight," *Sci. Robot.*, vol. 7, no. 67, 2022.
[12] Betaflight, "The betaflight open source flight controller firmware project," https://github.com/betaflight/betaflight.
[13] P. Foehn, *et al.*, "Time-optimal planning for quadrotor waypoint flight," *Science Robotics*, vol. 6, no. 56, 2021.
[14] A. Howard, *et al.*, "Searching for mobilenetv3," *arXiv*, 2019.
[15] K. Chen, *et al.*, "MMDetection: Open mmlab detection toolbox and benchmark," *arXiv*, 2019.