Nature Communications

https://doi.org/10.1038/s41467-026-73283-x

Article in Press

An autonomous single-actuator UAV with omnidirectional field of view, high agility, and collision resistance

Received: 31 May 2025

Accepted: 27 April 2026

Cite this article as: Chen, N., Li, H., Ren, Y. *et al.* An autonomous single-actuator UAV with omnidirectional field of view, high agility, and collision resistance. *Nat Commun* (2026). https://doi.org/10.1038/s41467-026-73283-x

Nan Chen, Haotian Li, Yunfan Ren, Guozheng Lu, Fangcheng Zhu, Jiarong Lin & Fu Zhang

An autonomous single-actuator UAV with omnidirectional field of view, high agility, and collision resistance

Nan Chen[1], Haotian Li[1], Yunfan Ren[1], Guozheng Lu[1], Fangcheng Zhu[1], Jiarong Lin[1], Fu Zhang[1*]

[1]Department of Mechanical Engineering, The University of Hong Kong, Pokfulam Road, Hong Kong, China.

*Corresponding author(s). E-mail(s): fuzhang@hku.hk;
Contributing authors: cnchen@connect.hku.hk; haotianli@connect.hku.hk; renyf@connect.hku.hk; gzlu@connect.hku.hk; zhufc@connect.hku.hk; zivlin@connect.hku.hk;

Abstract

A single-actuator unmanned aerial vehicle (UAV) named PULSAR II is driven by one motor and achieves three-dimensional maneuverability and natural self-rotation. This motion enables an omnidirectional field of view (FoV) for light detection and ranging (LiDAR), thereby eliminating unobserved areas. To enhance agility, we first built an accurate propulsion model using a data-driven method and compensated its nonlinearities. Then, we designed model predictive controller on the $\boldsymbol{S^2}$ manifold for minimal parameterization and aggressive tracking performance. We also derived the UAV's differential flatness to simplify trajectory optimization. To ensure collision resistance, we designed a protective solid frame structure around the UAV and developed a disturbance observer to reject collision-induced disturbances in a fictional coordinate frame that decouples self-rotation, with parameters optimized by $\boldsymbol{\mu}$ synthesis to address model uncertainty. By integrating all modules, we established an autonomous UAV system that demonstrated agile, robust, and fully autonomous flight in real-world environments.

Keywords: Autonomous UAV, single actuator, self-rotation, omnidirectional field of view, collision resistance, model predictive control, disturbance observer

1 Introduction

Unmanned aerial vehicles (UAVs) have become increasingly indispensable platforms for real applications. For example, UAVs are used for environmental mapping [1–3], inspection [4–7], and agriculture [8–10] based on onboard sensors such as cameras or light detection and ranging (LiDAR). In these applications, the efficiency of mapping, the stability of localization, and the observation of obstacles rely on the range of the sensor's field of view (FoV) [2, 11–13]. Meanwhile, key aspects of UAV flight efficiency, agility, and safety also affect overall performance in real applications [14].

Firstly, to deal with the challenges due to the limitation of FoV, some UAVs are directly equipped with sensors with relatively large FoV, such as fisheye cameras [15, 16] or 360° LiDAR [17, 18]. However, there are still about half of the areas that the FoV cannot cover. Using multiple sensors in one system, such as multiple common

cameras [19, 20], fisheye cameras [12, 13], stereo cameras [21, 22], and LiDARs [23, 24], is another solution to achieve a larger FoV. Whereas, it increases the weight, power consumption, and computational load, which are all obviously limited resources of the UAV system. Introducing a gimbal system [25–27] or a swing mechanism is a valid attempt, but beyond the drawback of additional weight and power consumption, the time-variant extrinsic parameters and potential actuator vibrations still lead to extra challenges to the UAV's localization and mapping.

Inherent self-rotation motion of UAVs is a promising solution to achieve sensor FoV far beyond the original one without increasing the weight, power consumption, and computational load [28–30]. Compared with the self-rotating UAVs based on multiple actuators [29–38], the single-actuator designs [28, 39–46] effectively reduce the components' weight and structure complexity through using a minimum number of actuators. However, payload capacity and localization problems during high-speed changes of FoV are still challenging, such that most of the existing single-actuator UAVs [39–46] do not consider extending FoV via their self-rotation. An exceptional single-actuator UAV is our previous prototype [28] that extends the LiDAR FoV from its original conical shape to cover all surrounding areas via self-rotation, but obvious unobserved areas still remain in the top and bottom directions.

Secondly, challenges remain in agility. Existing controllable single-actuator UAVs [42–46] cannot perform agile motions such as aggressive trajectory tracking mainly due to two reasons. The first one concerns actuation. The lift forces of these single-actuator UAVs are generated by a propeller installed on a position away from the center of mass (CoM) of the UAV body [42–44] or by a wing whose lift force direction does not pass through the UAV body's CoM [45, 46], such that the lift force can generate the moment for attitude control. To generate a moment in the desired direction, the lift force has to be regulated once per revolution of self-rotation. Therefore, the regulation frequency must match the self-rotation frequency, resulting in a low control effort frequency since the self-rotation speed is only several hertz. In our previous prototype [28], the regulation frequency of the control efforts matches the motor speed, which can reach hundreds of hertz. However, the nonlinearity of the propulsion system is not well modeled or compensated in the control system, resulting in insufficient accuracy for aggressive trajectory tracking. The second reason concerns control. These single-actuator UAVs are controlled by open-loop controller with passive stability [42], proportional-integral-derivative (PID) controllers [28, 44–46], or linear quadratic regulator (LQR) [43]. These control methods cannot utilize the future information from planned reference trajectories, resulting in limited tracking performance for aggressive trajectories.

Lastly, the safety of all single-actuator UAVs [28, 39–46] is still challenging. The designs of these UAVs do not consider collision resistance in the mechanical structure and disturbance rejection in the control system, significantly reducing flight safety, especially when operating in dense environments. Specifically, these UAVs have many fragile structures, such as unprotected propeller, long wing, or thin damping vanes, making them vulnerable to collisions. In addition, there is no dedicated disturbance rejection module in their control systems to handle disturbances induced by collisions, which obviously increases attitude errors and even the risk of crashes during collisions.

In this work, we introduce an autonomous, single-actuator UAV named PULSAR II with omnidirectional LiDAR FoV, high agility, and collision resistance, as shown in Fig. 1. Leveraging structural design and inherent self-rotation, this LiDAR-equipped UAV can perceive all directions without any unobserved areas. With a more accurately identified model of the propulsion system and an advanced tracking control framework of on-manifold model predictive control (OMMPC) [47], PULSAR II shows superior tracking performance of aggressive trajectories. To ensure collision resistance, a solid and lightweight outer frame design is introduced to a single-actuator UAV to protect the entire UAV structure during external impacts. Additionally, in a fictional coordinate frame that decouples self-rotation, a disturbance observer optimized by μ synthesis is integrated into the control system to compensate for collision-induced moment disturbances [48–50]. Overall, these measures improve the flight safety of PULSAR II. With the integrated system comprising modules of perception, planning,

and control, PULSAR II demonstrates high-speed autonomous navigation in unknown, cluttered, and Global Navigation Satellite System (GNSS)-denied environments.

2 Results

2.1 Overview of PULSAR II

PULSAR II is a UAV with a single-actuator design that uses only a single actuator (i.e., one motor) to achieve full 3D motion. With payloads of an onboard computer and a LiDAR sensor, it can perform fully autonomous navigation in GNSS-denied and cluttered environments (e.g., dense woods in Fig. 1a (i)) without any other external instruments. PULSAR II also has high agility to support its high-speed autonomous navigation in a parking lot shown in Fig. 1a (ii) and tracking of aggressive trajectories (e.g., flip trajectory in Fig. 1a (iii)). Furthermore, PULSAR II has a unique ability of collision resistance among all existing single-actuator UAVs, leading to much safer flights even with collisions in environments with obstacles, as shown in Fig. 1a (iv).

The inherent self-rotation motion of PULSAR II is a natural way to achieve a larger sensor FoV without additional sensors, actuators, and mechanisms. As shown in Fig. 1b (i), the limited original LiDAR FoV (blue area) is substantially extended to an omnidirectional FoV (yellow area). The point cloud maps of an indoor environment before and after take-off are shown in Fig. 1b (ii) and (iii). As can be seen, before take-off, the point cloud map is obviously incomplete due to the limited original LiDAR FoV. After take-off, a complete point cloud map of the indoor environment can be built without any unobserved area, including the ceiling and the floor, which are hard to map at the same time. This demonstrates the advantage of omnidirectional FoV. Furthermore, omnidirectional FoV is also crucial for autonomous navigation in complex and narrow environments. It can promote safety since the UAV does not need to move in a horizontal direction to identify the unknown space that is located in the unobserved area of the original LiDAR FoV. Instead, the UAV can directly plan a suitable path just in a hover state.

2.2 UAV Design

2.2.1 Structure of UAV body

The structures of PULSAR II and its separated components are shown in Fig. 2a, with weight distribution and description of the components. PULSAR II weighs 1122.3 g, which is mainly contributed by LiDAR (22.4%), protection frame (20.8%), and battery (19.5%). With only one motor, the whole 3D motion of PULSAR II can be controlled. Compared with the previous version of the single-actuator UAV proposed in [28], there are mainly four evolutions in design aspects: (i) a solid protection frame is designed to keep the UAV safe during collision; (ii) four damping vanes are naturally installed on the protection frame without extra supporting structures, which reduces the total weight and possibility of damage; (iii) a 360° LiDAR is mounted vertically, achieving a fully omnidirectional FoV without any unobserved area, which outperforms the previous one that has a large unobserved area at the top and bottom; (iv) lighter structural weight and larger capacity battery lead to longer flight duration and agility. See Supplementary Figs. 14-17 for more details about the UAV structure.

2.2.2 Structure of propulsion system

The structure of the propulsion system is shown in Fig. 2b, comprising two blades, a motor, and a swashplateless mechanism (SLM) [28, 51]. The SLM can generate both thrust in the motor shaft direction and moments in the propeller disk plane at the same time. Unlike the swashplate mechanism which is widely adopted in helicopters and utilizes a set of servos to adjust the blades' pitch angles, the SLM can adjust them by two passive hinge mechanisms through modulating the rotational speed during each rotational cycle of the motor. Since the axes of the two hinges are asymmetrical, the changes in the two blades are always opposite. For example, when the motor accelerates, the pitch angle of one blade increases while the other one decreases. This

leads to unbalanced thrust of the two blades such that a net moment can be generated in the propeller disk plane with a direction perpendicular to the blades. Likewise, when the motor decelerates, the SLM still generates the net moment, but its direction will be opposite. More detailed SLM working principles are introduced in the Supplementary Note 1.

To reduce the friction of hinge rotation and alleviate the deadzone phenomenon of the net moment, as shown in Fig. 2b, we added axial bearings, rolling bearings, and shoulder screws to the design of the SLM. To achieve accurate rotor speed modulation and moment generation, we attached a magnet to the motor shaft such that a magnetic encoder can measure the rotor's angle and speed accurately.

2.2.3 Electronics

The electronic components of PULSAR II are shown in Fig. 2a and their internal connection is shown in Fig. 2c, which is divided into five parts. (i) The entire system is powered by the power part containing a six-cell 53.7-Wh (2420-mAh) battery and a DC-DC converter. (ii) The computation part has two separate computing platforms, an onboard computer and a flight control unit (FCU), which are connected via the USB. The type of onboard computer is the iKOOLCORE R2 (shell removed) with a 4-core 3.7-GHz Intel N200 x86 processor, and the type of FCU is the Nxt-FC V1.2 [52] with an ARM STM32H7 chip. (iii) The sensing part comprises LiDAR (Livox Mid360), inertia measurement unit (IMU), and magnetic encoder. The LiDAR measures 3D point clouds at a frequency of 200,000 points per second via a FoV of 360° in the horizontal direction and -7° to 52° in the vertical direction. The used IMU locates on the FCU, which has better performance than the one contained inside the LiDAR. Besides LiDAR and IMU, a magnetic encoder (ams AS5147U) is installed on the bottom of the motor stator, measuring the rotor's angular position (i.e., rotor angle) and rotational speed, both at 2000 Hz. The measured data is transmitted to the FCU via the Serial Peripheral Interface (SPI) to ensure high data bandwidth and low latency. This 2000-Hz measuring frequency can generate about 24 measurements per rotor revolution when the UAV is in hover (i.e., motor speed is about 5000 rpm or 83.3 r/s). (iv) The actuation part contains a motor (T-MOTOR MN5006 KV450) driven by an electronic speed controller (ESC) of type T-MOTOR MINI F45A. The motor command is computed by the FCU and then sent to the ESC via the DShot 600 protocol which enables up to 33.3-kHz command transmission. (v) The communication part has a WiFi antenna and a radio control receiver. The former enables remote login of the onboard computer and the latter is used for human manual control through a radio controller.

2.3 Control system overview

As shown in Fig. 3a, the control system has a hierarchical structure consisting of two parts. The top-level part (orange area) is implemented on the ROS system running on the onboard computer with an x86 CPU, while the low-level part (green area) is realized on PX4 autopilot software [53] running on the ARM chip of the FCU. In this section, we briefly introduce the modules in these two parts for better understanding the comparison experiments below. More details are provided in the Section 4. Meanwhile, for performance comparison of trajectory tracking control, we additionally designed a cascade PID control framework with feedforward (see Supplementary Fig. 13 for detailed system structure of cascade PID).

2.3.1 Top-level control

The modules of the top-level part include LiDAR odometry (LIO), OMMPC, offline trajectory generator, and online trajectory planner, running in a frequency range from 50 to 200 Hz. The LIO is essentially a filter-based multi-sensor fusion framework [54] estimating all states of the UAV based on measurements of a 50-Hz point cloud from LiDAR and a 200-Hz IMU that is transmitted from the FCU to the onboard computer using MAVROS [55]. Since the measurements come from different components, we used LI-Init [56] to estimate their time difference and extrinsic parameters, and finally,

they were well compensated. Although the 50-Hz point cloud output can ensure a 50-Hz state update frequency of LIO, it is not high enough for the controller. Hence, we used the state that is generated by the 200-Hz prediction process of IMU measurement, resulting in 200-Hz state feedback which is sufficient to support the 100-Hz OMMPC. Details are described in the Supplementary Notes 4.

The system contains two trajectory sources. One is the offline trajectory generator, which can publish the planned offline polynomial trajectory with time $[\mathbf{p}_d^{\mathcal{I}}, \mathbf{v}_d^{\mathcal{I}}, \mathbf{a}_d^{\mathcal{I}}, \mathbf{j}_d^{\mathcal{I}}]$ as a command. Another is an online trajectory planner. We adopted our previous work SUPER [57] which can plan and optimize the trajectory directly on the point cloud map with high efficiency. To maintain a point cloud map, SUPER uses a uniform grid structure to spatially downsample the point cloud from LiDAR odometry. Similarly to the offline trajectory generator, the output of the online trajectory planner is also polynomial trajectories. However, the planned polynomial trajectory cannot be tracked directly by the OMMPC. We derived the property of differential flatness of PULSAR II to convert the polynomial trajectory to the reference trajectory of state and input, which is detailed in Methods (Section 4.2).

2.3.2 Low-level control

The structure of low-level control is depicted in Fig. 3b. The low-level control is implemented on the FCU with an operating frequency of 1000 Hz, which is much higher than the top-level part since it handles the rotational dynamics and drives the actuator. The low-level control receives control efforts $a_{T,d}$ and $\boldsymbol{\omega}_{\mathcal{F},d}^{\mathcal{I}}$ from OMMPC as desired inputs.

There are many modules implemented in the low-level control, such as actuator compensation, disturbance observer (DOB), attitude delay compensation, and $\mathcal{F}$ frame calculation. (i) The actuator compensation module is used to dynamically compensate for the nonlinear characteristics of the actuator, such as the coefficient of moment amplitude and the lag angle of moment direction, according to the input magnitude and measured motor speed in real time. Details are described in the Supplementary Notes 2. (ii) The attitude delay compensation module compensates for the delay in the attitude feedback due to the communication between the FCU and the onboard computer and the software processing time. In addition, it can increase the frequency of attitude feedback through the prediction process of high-frequency IMU data. Details of delay compensation are depicted in the Supplementary Notes 5. (iii) With high-frequency attitude feedback $\mathbf{R}^{\mathcal{IB}}$, the attitude related to $\mathcal{F}$ frame (i.e., $\mathbf{R}^{\mathcal{IF}}$ and $\mathbf{R}^{\mathcal{FB}}$) can be obtained in the module of $\mathcal{F}$ frame calculation. Details of the calculation principle can be found in the Supplementary Notes 6. (iv) The DOB module is designed to estimate external moment disturbances such that the UAV can maintain its attitude better during collisions. See details in Methods (Section 4.6) and Supplementary Notes 7.

2.4 Setup of real-world experiments

To evaluate the designed control framework, including actuator compensation, OMMPC, and DOB, as well as the entire system, we performed a series of indoor ablation experiments. We further evaluated PULSAR II autonomy through high-speed autonomous navigation in two complex outdoor environments. All experiments use full onboard sensing and computation, without any external computer, motion capture system, or GNSS.

2.5 Actuator compensation

The experiments in this section evaluate actuator compensation in low-level control to ensure actuator linearity over a wide range of motor speed changes (see Supplementary Video 1 and 2). The UAV tracks two trajectories with actuator compensation enabled or disabled. For fair ablation experiments, except for actuator compensation, all other configurations are the same, with OMMPC in the top-level control and $\mathcal{F}$-frame PI control with DOB in the low-level part.

First, the helix trajectory is a 3D trajectory with a constant radius of 0.6 m in the $y-z$ plane and a displacement of 2.4 m in the x axis. We specified 9 waypoints and optimized the trajectory using MINCO [58]. The optimized trajectory has two periods and a total duration of 4.8 s, and overlaid photos are shown in Fig. 4a.

As shown in Fig. 4b, the UAV can track the helix trajectory with or without compensation, with the highest velocity norm about 2 m s^{-1}. However, the experiment with compensation shows much smaller position errors in all three axes, especially in the second period. For the velocity norm, the experiment without compensation shows a significant overshoot. This is because the helix trajectory requires a wide thrust range due to large altitude change. Without compensation, the mapping from desired moment to generated moment varies obviously, degrading tracking performance. With compensation, the UAV tracks the trajectory well, showing that actuator compensation substantially improves tracking performance.

Second, we use a tilted figure 8 trajectory with displacements of 4 m on the x axis, 2 m on the y axis, and 1 m on the z axis. It is optimized by MINCO [58] with 17 waypoints. The optimized trajectory has a highest reference velocity norm of 3.57 m s^{-1} and two periods, with each period of 5 s, as shown in Fig. 4c.

The tracking results are shown in Fig. 4d. With compensation enabled, the UAV can track the trajectory with small position error. Without compensation, the UAV cannot finish the tracking and crashed in the second period, with oscillating and diverging position error. Compared with the helix trajectory, the tilted figure 8 has higher reference velocity norm and more aggressive attitude changes, so a constant actuator mapping is not valid due to nonlinearity from wide-range motor speed and desired moment variation. Hence, actuator compensation is indispensable for tracking aggressive trajectories of PULSAR II.

2.6 Aggressive trajectory tracking

To evaluate aggressive trajectory tracking of PULSAR II, we conducted two experiments, a high-speed tilted figure 8 trajectory and a flip trajectory (see Supplementary Video 3 and 4). We compared two top-level controllers, OMMPC and cascade PID with feedforward, which is widely used for UAV tracking control (see Supplementary Fig. 13 for the PID structure). Both OMMPC and the PID feedforward use the reference state trajectory obtained via differential flatness conversion (see Fig. 3), providing complete prior trajectory information and ensuring a fair comparison. The parameters of OMMPC and cascade PID were first set in simulation and then well tuned in real-world flight. The low-level control is identical for all cases, with actuator dynamic compensation enabled and PI control in the $\mathcal{F}$ frame with DOB.

2.6.1 Tilted figure 8 trajectory

The tilted figure 8 trajectory used for tracking experiments has the same shape as the trajectory in the actuator compensation experiments in Fig. 4c, and still has two periods. We used the same method to generate the trajectory with a higher time penalty in the optimization, so it has a shorter period of 4 s instead of 5 s and a higher reference velocity norm of 4.04 m s^{-1} such that it can better evaluate the high-speed tracking performance, as shown in Fig. 5a.

We conducted tracking experiments three times for each controller. The position error norm of all 6 experiments is shown in Fig. 5b, where OMMPC has an obviously smaller error than PID with feedforward. For the 3-axis position in Fig. 5c (i), the tracked trajectory has almost no phase shift from the reference due to the reference of OMMPC and the feedforward of PID. OMMPC tracks all 3-axis positions with almost no amplitude attenuation, while the cascade PID shows obvious amplitude attenuation in the y axis due to its higher signal frequency, indicating a higher equivalent control bandwidth for OMMPC. For the position error in Fig. 5c (ii), OMMPC has smaller errors in all 3 axes, with maxima of 0.202 m versus 0.414 m in x axis, 0.188 m versus 0.364 m in y axis, and 0.148 m versus 0.242 m in z axis. The attitude vector, attitude error vector, and attitude angle error are shown in Fig. 5c (iii-iv). Both controllers track the reference attitude well, and OMMPC shows smaller error values and error bands.

In Fig. 5c (v), the first figure shows that OMMPC follows the reference velocity norm more closely than PID. The second and third figures show control efforts of angular velocity norm and thrust. The cascade PID output follows the reference closer because it depends on feedforward and current errors. OMMPC sometimes deviates from the reference because it uses more control effort to minimize state error in the prediction horizon. With modeling errors and external disturbances, OMMPC tends to sacrifice control-effort tracking to maintain smaller state error.

2.6.2 Flip trajectory

Another aggressive trajectory, the flip trajectory, is adopted to evaluate the tracking performance of PULSAR II. We specify 5 waypoints to define the trajectory, with x-axis displacement of 2.2 m, z-axis height of 3.1 m, and no movement in y axis. To achieve the flip motion, we constrain the norm of the UAV's total acceleration at the top point of the flip trajectory to 22 m s^{-2}, in the same direction as gravity. Since this norm is larger than gravity and the propeller cannot invert thrust direction, the trajectory optimization forces the UAV to flip its attitude to satisfy the total acceleration requirement, as shown in Fig. 6a (ii).

The UAV successfully tracked the flip trajectory, as shown in the overlaid photo in Fig. 6a (i). We conducted tracking experiments three times for each controller, with position error norm shown in Fig. 6b. OMMPC always shows a smaller position error norm than PID with feedforward. Tracking results are shown in Fig. 6c. Compared with the tilted figure 8 trajectory, the flip trajectory has aggressive attitude changes, higher maximum velocity norm of 4.51 m s^{-1}, a thrust range from 9.4 to 17.3 m s^{-2}, and rapid angular velocity changes from 50 to 560 deg/s. In the mean tracking results, OMMPC shows slightly smaller position errors in x and y axes and a much smaller error in the z axis, indicating better performance for aggressive motion. For attitude error, both controllers have small errors, while OMMPC has relatively larger errors because it reduces position error by relaxing attitude tracking when position weights are much higher.

For the z-axis position error, although OMMPC surpasses PID with feedforward, the tracked trajectory in Fig. 6c (i) remains below the reference, and thrust saturation under constraints in Fig. 6c (v) finally prevents the UAV from reaching the top point. Possible reasons are: (i) thrust response dynamics is not considered in the OMMPC model, so the throttle is assumed instantaneous although there is delay; (ii) closed-loop low-level control dynamics is not modeled for OMMPC, so tracking fast attitude changes may cause phase delay; (iii) state errors from unmodeled dynamics and external disturbances lead to larger control efforts than the reference and cause saturation, as shown in Fig. 6c (v), where thrust is bounded from 3 to 19 m s^{-2} and x- and y-axis angular velocities are bounded from -15 to 15 rad s^{-1} during OMMPC solving for safety.

2.7 Collision resistance

To evaluate the UAV protection frame and the DOB design, we carried out two types of experiments. In the first type, the UAV collides with a ball dropping from a fixed height during hover flight to evaluate endurance against dynamic obstacles (see Supplementary Video 5). In the second type, the UAV collides with an unfixed box during forward flight and pushes through the box to maintain movement, which evaluates robustness against static obstacles (see Supplementary Video 6).

Angular velocity control was implemented in two coordinate frames, the body-fixed frame $\mathcal{B}$ with a z-direction angular velocity due to self-rotation and the fictional frame $\mathcal{F}$ with zero z-direction angular velocity. This yields four configurations: (i) $\mathcal{B}$ frame control without DOB; (ii) $\mathcal{F}$ frame control without DOB; (iii) $\mathcal{B}$ frame control with DOB; and (iv) $\mathcal{F}$ frame control with DOB.

2.7.1 Collision with dropping ball

In this experiment, the UAV first hovers at a fixed position in the inertial frame $\mathcal{I}$. A soft ball with a small iron sheet is held at the tip of a rigid arm by an electromagnet to

ensure consistent conditions, including the same height, zero initial velocity, no rotation of the ball, and a consistent collision point on the UAV. When the experiment begins, we disable the electromagnet to release the ball by manual remote control. The ball drops freely from 60 cm above the UAV and collides with one side of the protection frame. After impact, the UAV experiences a momentary disturbance that causes attitude and position errors, as shown in Fig. 7a. We performed three experiments for each configuration. The results of each experiment are shown in Fig. 7b, and the data distributions are shown in Fig. 7c.

Because the ball release is manually controlled, it is difficult to synchronize the three experiments, so we plot them as separate lines in Fig. 7b. We select a 2.5 s window that covers the entire collision process. For the x axis, the $\mathcal{F}$-frame configuration with DOB achieves the smallest errors. Since the collision point is on the left side of the UAV (see Fig. 7a, positive direction of y axis of $\mathcal{I}$ frame), the consistently small x-axis errors indicate good decoupling in the $\mathcal{F}$-frame control, while other configurations show obvious x-axis errors caused by the y-axis collision, indicating coupling between the x and y axes. For the y and z axes, the $\mathcal{F}$-frame configuration with DOB generally has smaller errors because DOB compensates for the momentary disturbance and stabilizes the attitude more quickly, reducing the position error. The attitude angle error data show the same trend, with the $\mathcal{F}$-frame configuration with DOB having much smaller errors than others. Comparing the two DOB configurations, the one in the $\mathcal{F}$ frame performs better because self-rotation is decoupled in the $\mathcal{F}$ frame, allowing DOB to handle the low-frequency disturbance signal better than the relatively high-frequency signal coupled with self-rotation.

For each configuration, we merge data from the three experiments into one violin plot in Fig. 7c to illustrate the distributions. The red point is the mean, and the black error bar shows the range from 25% to 75%. In the x and z axes, the $\mathcal{F}$-frame configuration with DOB is more concentrated near zero with a shorter vertical extent. In the y axis, its error magnitude is also smaller than the others. The attitude error violin plot is more significant, as the $\mathcal{F}$-frame configuration with DOB has a lower central area and shorter height, indicating more stable attitude during the ball collision. Overall, the $\mathcal{F}$-frame control with DOB has the best performance among all configurations, showing strong disturbance rejection, especially in the $\mathcal{F}$ frame.

2.7.2 Collision with unfixed box

Unlike the previous ball-dropping experiment, the collision in this experiment does not occur during hover flight. We designed a straight-flight trajectory that drives the UAV forward (i.e., opposite y axis) to the designated goal position, with an unfixed soft box placed along the trajectory. After colliding with the box, the controller maintains forward movement so the UAV pushes the box aside and reaches the goal position, as depicted in Fig. 7d.

We performed three experiments for each configuration, synchronized the data by the reference trajectory, and show the results in Fig. 7e. Besides the reference signal, the other four lines are the mean values of the four configurations. When a collision occurs, there is an x-axis position deflection and a z-axis altitude drop, where the $\mathcal{F}$-frame configuration with DOB achieves the smallest amplitude. For the moving direction (y axis) and the velocity norm, all configurations show similar trends and converging times. When the reference trajectory ends (after 1.5 s), position-error convergence is slow because the reference position and integration of position error drive the UAV to the goal position, while other reference states tend to keep the UAV stopped at the current position. This phenomenon is due to OMMPC and has no relation to DOB. Overall, the DOB in the $\mathcal{F}$ frame shows better disturbance rejection performance than other configurations in the forward-flight box-collision experiment.

2.8 Autonomous navigation in unknown environments

To validate the complete control framework and autonomous system of the single-actuator UAV, we conducted high-speed autonomous navigation experiments in complex unknown environments, including dense woods (Fig. 8) and an underground parking lot (Fig. 9). In each experiment, we specified a start point, an end point that

coincides with the start point, and a set of guide points, marked as purple and blue stars on the point cloud maps in Fig. 8a and Fig. 9a. These maps are built online by the LiDAR odometry module during flight. Each guide point serves as a local goal that shapes the overall trajectory. When the UAV enters a predefined radius around a guide point, it is considered reached and the next guide point becomes the local goal. We use our previous work SUPER [57] for online trajectory planning, running at 50 Hz. If obstacles lie between the current position and the next goal, the planner computes a collision-free corridor and optimizes a dynamically feasible trajectory within it. The trajectory tracking controller then ensures accurate tracking of the planned trajectory, which is essential for high-speed autonomous navigation of a single-actuator UAV.

2.8.1 Dense woods

The first experimental environment is dense woods, with trees, branches, and weeds scattered throughout the space. The flight area has a size of 48 m $\times$ 43 m, and we set two series of guide points to shape two trajectories, a loop trajectory and a star trajectory (see Supplementary Video 7 and 8).

The first autonomous navigation is a loop shaped by 9 guide points (one start point and points P1-P8). The actual flight trajectory is shown in Figs. 8a-b and the flight data are shown in Fig. 8c. Fig. 8a shows that the UAV avoided obstacles and returned to the starting point. With the reaching threshold set to 1 m, the UAV almost passed through all guide points. The yellow arrows indicate zoom-in directions for detailed trajectories, and the corresponding views of the data visualization tool and real-world snapshots are shown in Fig. 8b. The UAV performed smooth turns and flew through gaps between trees without collisions. In Fig. 8c, the position error in the z axis is very small, while the errors in the x and y axes are larger but bounded within 0.5 m. For attitude and velocity norm, the UAV tracked the reference well and reached a maximum velocity norm of 6 m s^{-1} in this dense environment. The velocity norm decreased to zero twice because the planner spent more time finding a trajectory due to dense obstacles. The UAV decelerated more when approaching each guide point because the reaching threshold is small. The two control efforts followed their references most of the time, with peaks in the angular velocity norm during aggressive sharp turns.

The second autonomous navigation experiment uses similar configurations, but the trajectory is a more complicated star shape with sharper turns at the vertices. The trajectory is shaped by 7 guide points (one starting point and points P1-P6), and the actual flight trajectory is shown in Fig. 8d-e. At a star vertex, as shown in Fig. 8e(ii), the planner generated a smooth high-curvature trajectory, and the UAV followed it without colliding with nearby trees. In the flight data in Fig. 8f, the position error in the z axis remains the smallest, while those in the x and y axes are slightly larger than 0.5 m, and large errors mainly appear during acceleration after a sharp turn. The maximum velocity norm is also 6 m s^{-1}, which is high speed for a single-actuator UAV's autonomous navigation. There were two stops at two star vertices (P2 and P4) because the planned turns are acute-angle turns rather than arc turns, so the UAV had to decrease the velocity to zero first.

2.8.2 Underground parking lot

The second experimental environment is an underground parking lot (see Supplementary Video 9). As shown in Fig. 9a-b, there are numerous obstacles, including dense pillars, multiple doors, and complex ceiling pipes, which are very challenging for autonomous navigation. We specified eight guide points (one start point and points P1-P7) to shape the trajectory, and set a higher reaching threshold of 3 m for each guide point. As shown in Fig. 9a, the UAV successfully performed autonomous navigation, but the actual trajectory did not pass through all guide points, which meets expectations. This setting allows the planner to optimize the trajectory earlier before arriving at a guide point. As a result, although decelerations occur in turns, the velocity norm remains more than 6 m s^{-1} and never drops to an approximate zero value, as shown in Fig. 9c. For position tracking, the maximum horizontal error is 0.77 m,

which is larger than previous results but sufficient to ensure safe flight in this environment. Large position errors occurred during straight flight at high speed because: (i) a conflict between position error and attitude error occurs since no air drag model is considered in OMMPC, causing the UAV to tilt more unexpectedly to resist air drag, and (ii) the penalty weight of OMMPC for the state of position error integration is not large enough to attenuate the error over a short period. Fig. 9b shows zoom-in views, including passing through a door and turning around pillars, demonstrating the agility and autonomy of the single-actuator UAV.

2.9 Comparisons

Compared with common quadrotors, many advantages, such as a simpler structure, reduced non-payload weight, higher efficiency, and inherent FoV extension, have been demonstrated in PULSAR [28]. In this section, we mainly focus on comparing PULSAR II with PULSAR [28] as well as other state-of-the-art self-rotating UAVs [29–46], highlighting the contributions in the conceptual realization, disturbance rejection framework, module refinement, and system-level enhancement.

2.9.1 LiDAR omnidirectional FoV with minimal components

Fully omnidirectional FoV is very beneficial for trajectory planning in narrow areas because the UAV does not need to adjust its FoV to observe the unobserved regions without horizontal movement, enabling generation of collision-free trajectories even in hover. The design of PULSAR II completely realizes the concept of fully omnidirectional FoV on a UAV with a minimal set of components, that is a single LiDAR sensor and no additional actuators, such as gimbal motors. This realization has not been fully achieved in PULSAR or, to our knowledge, in any other existing UAV platforms.

In prior systems, extending UAV's single LiDAR FoV typically fell into one of three categories. First, even a UAV with state-of-the-art LiDAR sensors offering a hemispherical FoV (e.g., HESAI JT128 [59] with 360° horizontal and 189° vertical coverage) still leaves half of the full spherical volume unobserved. Second, some UAVs incorporate a LiDAR with an additional rotating mechanism (e.g., a UAV from Exyn Technologies adopts a motor to rotate the LiDAR [60]). While this approach expands FoV coverage, it adds extra weight and still suffers from obvious occlusion by the UAV's own body (estimated occlusion is vertically 70° and horizontally 30°), preventing a fully unoccluded, omnidirectional FoV.

Third, extends FoV via self-rotation. Most self-rotating UAVs cannot carry visual sensors due to limited payload and compute, with a few exceptions [28–30]. PULSAR [28] extends FoV through self-rotation but still leaves blind regions, especially vertically. Similarly, Lockheed Martin's dual-actuator self-rotating UAV [29] retains unobserved areas in its camera FoV. The self-rotating quadrotor with 2D LiDAR in [30] achieves omnidirectional horizontal coverage yet still has vertical unobserved regions due to its narrow vertical FoV and structural occlusions.

In contrast, PULSAR II achieves a fully omnidirectional FoV with minimal additional components. This is enabled not only by LiDAR selection (i.e., adopting a modern commercial-off-the-shelf LiDAR with 360° FoV), but also by deliberate structural co-design: (i) the LiDAR is designedly mounted with a 90° tilt such that its native FoV sweeps a full spherical volume through self-rotation; (ii) the structural design eliminates the FoV occlusion: the airframe totally avoids blocking the region beneath the UAV, while gaps between rotating propeller blades allow observation above; (iii) damping vanes are placed to minimize their projection in the LiDAR FoV, preventing beam refraction and reflection.

2.9.2 Disturbance rejection framework for self-rotating UAVs

Previous self-rotating UAVs were largely proof-of-concept systems because of limited actuation, agility, and payload capacity, and they often did not address external disturbances such as wind or collisions. In this work, we introduce a DOB design that accounts for the coupled characteristics of angular velocity induced by self-rotation, resulting in an effective disturbance-rejection control framework for self-rotating

UAVs. First, we define a non-rotating virtual coordinate frame using IMU integration and derive the dynamic model in this frame. We prove that accumulated IMU integration error does not affect the equivalence between the dynamics in the self-rotating body-fixed frame ($\mathcal{B}$) and the non-rotating virtual frame ($\mathcal{F}$). Second, based on the $\mathcal{F}$-frame model, we design and implement a multi-input multi-output (MIMO) DOB in the low-level controller to separate high-frequency components induced by self-rotation from low-frequency disturbances, enabling more effective rejection of true low-frequency disturbances. Third, we optimize the disturbance observer parameters via μ-synthesis to account for model uncertainties while balancing disturbance rejection and noise suppression. Finally, extensive real-world comparative experiments show that the proposed framework improves disturbance rejection performance.

2.9.3 Module refinement and system-level enhancement

As autonomous UAVs comprise multiple modules, overall performance depends on both individual module capability and system integration. With refinements of each module and more thorough system integration, PULSAR II shows substantial performance enhancements over the previous prototype [28] and other self-rotating UAVs [29–46].

Mechanically, PULSAR II is equipped with a protective frame that improves collision resilience and enables mounting fragile wind deflectors inside the frame, removing the extra support structures used in [28]. In propulsion, model identification and dynamic compensation address the nonlinearities of swashplateless mechanism and improve agility. For trajectory generation, we derive the differential flatness of PULSAR II. Because self-rotating UAVs control only three DoF and do not require yaw control, we choose 3D position as the flat outputs, avoiding redundant yaw planning and reducing optimization complexity. In control, we incorporate OMMPC to handle states on S^2, improving tracking accuracy over the traditional PID with feedforward.

System integration remains challenging. Modules must be assigned to the onboard computer or FCU and integrated to support high agility and high-speed autonomy, and each improved module must be validated in the full system. This process can expose new issues. For instance, communication between two computing platforms introduced a small estimation delay that was amplified by high-speed rotation and degraded aggressive-flight tracking. We addressed this by adding a delay-compensation module that predicts states from the latest estimates using recorded IMU data. As a result, PULSAR II achieves more precise state estimation, enabling aggressive tracking of high-speed figure-8 and flip trajectories and autonomous navigation with speed exceeding 6 m s^{-1}, whereas PULSAR flew conservatively at 1 to 2 m s^{-1}.

2.9.4 Energy efficiency improvement

Compared with common quadrotor UAVs, single-actuator UAV demonstrates much higher energy efficiency due to the reduced structural weight and higher energy conversion efficiency of the propulsion system, which are quantitatively analyzed in [28]. Beyond that, PULSAR II further improves the overall energy efficiency by integrating modern components and better design: (i) the AVIA LiDAR is replaced with the more suitable Mid-360 LiDAR, reducing LiDAR mass by 233 g; (ii) the battery is replaced with a higher energy-density one (i.e., 12 Wh increased and 50 g reduced); despite PULSAR II is equipped with the protective frame and a more powerful computer, its total mass still decreases by 112 g; (iii) the damping vanes are installed fully parallel with the propeller downwash airflow to reduce air drag. Hence, PULSAR II achieves an average hover power of 147 W (132 W of propulsion system and 15 W of electronics), leading to an efficiency of 7.63 g W^{-1} and a hover duration of 15 min 50 s (see Supplementary Video 10). These results are higher than those of PULSAR [28] (i.e., 6.7 g W^{-1} and 12 min 30 s).

3 Discussion

Extensive real-world experiments validated PULSAR II from four aspects: (i) an ablation study based on trajectory tracking verified the effectiveness of actuator compensation; (ii) aggressive trajectory tracking, including high-speed figure-8 and flip flights, demonstrated precise and stable control; (iii) collision and disturbance tests, including impacts during hover and forward flight, confirmed strong collision resilience and robustness; (iv) high-speed autonomous navigation in complex unknown environments demonstrated reliable system-level autonomy and robust performance. These experiments also verified long-term mechanical reliability. The sturdy carbon-fiber protection frame with a stable overall structure prevents damage to the propeller and damping vanes during collisions or crashes, preserving structural integrity over extended operation. Adding bearings to the swashplateless mechanism resulted in very slight wear during frequent flights, indicating consistent and reliable performance.

PULSAR II also has strong potential for diverse applications due to its omnidirectional FoV, collision resistance, high agility, and high-speed autonomous navigation. It is suitable for weak or unavailable GNSS environments such as dense forests, caves, tunnels, and underground parking lots, and for search and rescue, disaster relief, and terrain mapping tasks requiring autonomous navigation, obstacle avoidance, and high mapping efficiency. High energy efficiency and extended flight duration further improve endurance. State estimation currently relies on LiDAR-inertial odometry, which may fail in open spaces with insufficient LiDAR returns. In most obstacle-rich environments, the omnidirectional FoV observes a larger portion of the surroundings and provides sufficient points for stable state estimation in mapping. In future work, we plan to integrate additional sensors such as GPS and a compass to ensure robust state estimation even in fully open areas with few LiDAR points.

4 Methods

4.1 UAV dynamic modeling

4.1.1 Definition of coordinate frames

As shown in Fig. 10a, three coordinate frames are defined for UAV modeling: (i) an inertial (ground-fixed) coordinate frame $\mathcal{I}$ (basis vectors $[\mathbf{x}^{\mathcal{I}}, \mathbf{y}^{\mathcal{I}}, \mathbf{z}^{\mathcal{I}}]$) whose z-axis is the same as the direction of the gravity vector, (ii) a body-fixed coordinate frame $\mathcal{B}$ (basis vectors $[\mathbf{x}^{\mathcal{B}}, \mathbf{y}^{\mathcal{B}}, \mathbf{z}^{\mathcal{B}}]$) attached to the UAV body and whose origin coincides with the UAV's CoM, and (iii) a fictional coordinate frame $\mathcal{F}$ (basis vectors $[\mathbf{x}^{\mathcal{F}}, \mathbf{y}^{\mathcal{F}}, \mathbf{z}^{\mathcal{F}}]$) whose origin and z axis coincide with $\mathcal{B}$ and has no z-axis rotation on $\mathcal{F}$ (see Supplementary Note 6 for more details).

Defining $\boldsymbol{\omega}^{\mathcal{B}}_{\mathcal{B},xy} = [\omega^{\mathcal{B}}_{\mathcal{B},x}, \omega^{\mathcal{B}}_{\mathcal{B},y}, 0]^{\mathsf{T}}$, $\boldsymbol{\omega}^{\mathcal{B}}_{\mathcal{B},z} = [0, 0, \omega^{\mathcal{B}}_{\mathcal{B},z}]^{\mathsf{T}}$, then $\boldsymbol{\omega}^{\mathcal{B}}_{\mathcal{B}} = \boldsymbol{\omega}^{\mathcal{B}}_{\mathcal{B},xy} + \boldsymbol{\omega}^{\mathcal{B}}_{\mathcal{B},z} = [\omega^{\mathcal{B}}_{\mathcal{B},x}, \omega^{\mathcal{B}}_{\mathcal{B},y}, \omega^{\mathcal{B}}_{\mathcal{B},z}]^{\mathsf{T}}$ is the angular velocity of $\mathcal{B}$ and is represented in $\mathcal{B}$. Since frame $\mathcal{F}$ has no rotation on z axis, the rotation between $\mathcal{F}$ and $\mathcal{B}$ satisfies

$$\dot{\mathbf{R}}^{\mathcal{FB}} = \mathbf{R}^{\mathcal{FB}} \lfloor \boldsymbol{\omega}^{\mathcal{B}}_{\mathcal{B},z} \rfloor \tag{1}$$

According to the relation of coordinate transformation $\mathbf{R}^{\mathcal{IB}} = \mathbf{R}^{\mathcal{IF}}\mathbf{R}^{\mathcal{FB}}$, we have

$$\dot{\mathbf{R}}^{\mathcal{IB}} = \dot{\mathbf{R}}^{\mathcal{IF}}\mathbf{R}^{\mathcal{FB}} + \mathbf{R}^{\mathcal{IF}}\dot{\mathbf{R}}^{\mathcal{FB}}. \tag{2}$$

Also, the rotation kinematics of frames $\mathcal{I}$ and $\mathcal{F}$ are

$$\dot{\mathbf{R}}^{\mathcal{IB}} = \mathbf{R}^{\mathcal{IB}} \lfloor \boldsymbol{\omega}^{\mathcal{B}}_{\mathcal{B}} \rfloor \tag{3}$$

where $\lfloor \cdot \rfloor$ represents the skew-symmetric matrix. Then, we can further obtain

$$\begin{aligned} \dot{\mathbf{R}}^{\mathcal{IB}} &= \mathbf{R}^{\mathcal{IB}} \lfloor \boldsymbol{\omega}^{\mathcal{B}}_{\mathcal{B},xy} \rfloor + \mathbf{R}^{\mathcal{IB}} \lfloor \boldsymbol{\omega}^{\mathcal{B}}_{\mathcal{B},z} \rfloor \\ &= \mathbf{R}^{\mathcal{IB}} \lfloor \boldsymbol{\omega}^{\mathcal{B}}_{\mathcal{B},xy} \rfloor + \mathbf{R}^{\mathcal{IB}} (\mathbf{R}^{\mathcal{FB}})^{\mathsf{T}} \dot{\mathbf{R}}^{\mathcal{FB}} \\ &= \mathbf{R}^{\mathcal{IB}} \lfloor \boldsymbol{\omega}^{\mathcal{B}}_{\mathcal{B},xy} \rfloor + \mathbf{R}^{\mathcal{IF}} \dot{\mathbf{R}}^{\mathcal{FB}}. \end{aligned} \tag{4}$$

Combining Equations (2) and (4), we have

$$\mathbf{R}^{\mathcal{IB}}\lfloor\boldsymbol{\omega}^{\mathcal{B}}_{\mathcal{B},xy}\rfloor = \dot{\mathbf{R}}^{\mathcal{IF}}\mathbf{R}^{\mathcal{FB}} \tag{5}$$

and then it yields

$$\dot{\mathbf{R}}^{\mathcal{IF}} = \mathbf{R}^{\mathcal{IF}}\mathbf{R}^{\mathcal{FB}}\lfloor\boldsymbol{\omega}^{\mathcal{B}}_{\mathcal{B},xy}\rfloor(\mathbf{R}^{\mathcal{FB}})^{\mathsf{T}} = \mathbf{R}^{\mathcal{IF}}\lfloor\mathbf{R}^{\mathcal{FB}}\boldsymbol{\omega}^{\mathcal{B}}_{\mathcal{B},xy}\rfloor = \mathbf{R}^{\mathcal{IF}}\lfloor\boldsymbol{\omega}^{\mathcal{F}}_{\mathcal{F}}\rfloor \tag{6}$$

where $\boldsymbol{\omega}^{\mathcal{F}}_{\mathcal{F}} = \mathbf{R}^{\mathcal{FB}}\boldsymbol{\omega}^{\mathcal{B}}_{\mathcal{B},xy}$ is a new symbol introduced to represent the angular velocity of the $\mathcal{F}$ frame and is represented in the $\mathcal{F}$ frame. This result indicates that the attitude kinematics of $\mathbf{R}^{\mathcal{IF}}$ is purely determined by the angular velocity of $\boldsymbol{\omega}^{\mathcal{F}}_{\mathcal{F}}$. The z-axis component of $\boldsymbol{\omega}^{\mathcal{F}}_{\mathcal{F}}$ is zero because $\boldsymbol{\omega}^{\mathcal{B}}_{\mathcal{B},xy}$ has a zero z component and $\mathbf{R}^{\mathcal{FB}}$ is a rotation only along the z axis of $\mathcal{B}$. Additionally, due to the pure z rotation of $\mathbf{R}^{\mathcal{FB}}$, we have $\boldsymbol{\omega}^{\mathcal{F}}_{\mathcal{B},z} = \boldsymbol{\omega}^{\mathcal{B}}_{\mathcal{B},z}$. For simplicity, we define a new symbol $\boldsymbol{\Omega}^{\mathcal{F}}_{B} = \boldsymbol{\omega}^{\mathcal{F}}_{\mathcal{B},z}$ to represent the angular velocity of the rotor part (that is, the self-rotation of the UAV) with respect to the $\mathcal{F}$ frame.

Equation (6) indicates that the fictional frame $\mathcal{F}$ implicitly defined by $\mathbf{R}^{\mathcal{FB}}$ in Equation (1) also satisfies a rotational kinematics of $\mathbf{R}^{\mathcal{IF}}$ whose angular velocity can be easily calculated by $\boldsymbol{\omega}^{\mathcal{F}}_{\mathcal{F}} = \mathbf{R}^{\mathcal{FB}}\boldsymbol{\omega}^{\mathcal{B}}_{\mathcal{B},xy}$. This property is the foundation for establishing the dynamic rotational model in the $\mathcal{F}$ frame to achieve self-rotation decoupling, which enables better disturbance rejection performance (see the next section for modeling and the experimental results of collision resistance).

4.1.2 Rotational dynamics in $\mathcal{F}$ frame

The total angular momentum $\mathbf{L}^{\mathcal{I}}$ in $\mathcal{I}$ frame is

$$\mathbf{L}^{\mathcal{I}} = \mathbf{R}^{\mathcal{IF}}(\mathbf{I}^{\mathcal{F}}_{O}\boldsymbol{\omega}^{\mathcal{F}}_{\mathcal{F}} + \mathbf{I}^{\mathcal{F}}_{B}\boldsymbol{\Omega}^{\mathcal{F}}_{B} + \mathbf{I}^{\mathcal{F}}_{R}\boldsymbol{\Omega}^{\mathcal{F}}_{R}), \tag{7}$$

where $\mathbf{I}^{\mathcal{F}}_{O}$, $\mathbf{I}^{\mathcal{F}}_{B}$, $\mathbf{I}^{\mathcal{F}}_{R}$ are the inertia matrices of the whole UAV, UAV's body (body part), and UAV's propeller and motor rotor (rotor part), respectively, which are all diagonal and the first two elements on the main diagonal are the same. $\boldsymbol{\Omega}^{\mathcal{F}}_{B}$ and $\boldsymbol{\Omega}^{\mathcal{F}}_{R}$ are the angular velocities of the body part and the rotor part represented in the $\mathcal{F}$ frame, respectively, and their directions are illustrated in Fig. 10a.

According to the angular momentum theorem $\dot{\mathbf{L}}^{\mathcal{I}} = \mathbf{M}^{\mathcal{I}}_{t}$ and $\mathbf{M}^{\mathcal{I}}_{t} = \mathbf{R}^{\mathcal{IF}}\mathbf{M}^{\mathcal{F}}_{t}$, we can obtain

$$\mathbf{M}^{\mathcal{F}}_{t} = \lfloor\boldsymbol{\omega}^{\mathcal{F}}_{\mathcal{F}}\rfloor(\mathbf{I}^{\mathcal{F}}_{O}\boldsymbol{\omega}^{\mathcal{F}}_{\mathcal{F}} + \mathbf{I}^{\mathcal{F}}_{B}\boldsymbol{\Omega}^{\mathcal{F}}_{B} + \mathbf{I}^{\mathcal{F}}_{R}\boldsymbol{\Omega}^{\mathcal{F}}_{R}) + (\mathbf{I}^{\mathcal{F}}_{O}\dot{\boldsymbol{\omega}}^{\mathcal{F}}_{\mathcal{F}} + \mathbf{I}^{\mathcal{F}}_{B}\dot{\boldsymbol{\Omega}}^{\mathcal{F}}_{B} + \mathbf{I}^{\mathcal{F}}_{R}\dot{\boldsymbol{\Omega}}^{\mathcal{F}}_{R}) \tag{8}$$

where $\mathbf{M}^{\mathcal{F}}_{t}$ is the total moment vector in $\mathcal{F}$ frame. With $\lfloor\boldsymbol{\omega}^{\mathcal{F}}_{\mathcal{F}}\rfloor\mathbf{I}^{\mathcal{F}}_{O}\boldsymbol{\omega}^{\mathcal{F}}_{\mathcal{F}} = \mathbf{0}$ due to $I^{\mathcal{F}}_{O,x} = I^{\mathcal{F}}_{O,y}$, we have

$$\mathbf{M}^{\mathcal{F}}_{t} = \lfloor\boldsymbol{\omega}^{\mathcal{F}}_{\mathcal{F}}\rfloor(\mathbf{I}^{\mathcal{F}}_{B}\boldsymbol{\Omega}^{\mathcal{F}}_{B} + \mathbf{I}^{\mathcal{F}}_{R}\boldsymbol{\Omega}^{\mathcal{F}}_{R}) + (\mathbf{I}^{\mathcal{F}}_{O}\dot{\boldsymbol{\omega}}^{\mathcal{F}}_{\mathcal{F}} + \mathbf{I}^{\mathcal{F}}_{B}\dot{\boldsymbol{\Omega}}^{\mathcal{F}}_{B} + \mathbf{I}^{\mathcal{F}}_{R}\dot{\boldsymbol{\Omega}}^{\mathcal{F}}_{R}) \tag{9}$$

Isolating the xy and z directions yields the rotational dynamic model in $x-y$ plane and in z axis as

$$\begin{aligned}\mathbf{M}^{\mathcal{F}}_{t,xy} &= \lfloor\boldsymbol{\omega}^{\mathcal{F}}_{\mathcal{F}}\rfloor(\mathbf{I}^{\mathcal{F}}_{B}\boldsymbol{\Omega}^{\mathcal{F}}_{B} + \mathbf{I}^{\mathcal{F}}_{R}\boldsymbol{\Omega}^{\mathcal{F}}_{R}) + \mathbf{I}^{\mathcal{F}}_{O}\dot{\boldsymbol{\omega}}^{\mathcal{F}}_{\mathcal{F}}\\ \mathbf{M}^{\mathcal{F}}_{t,z} &= \mathbf{I}^{\mathcal{F}}_{B}\dot{\boldsymbol{\Omega}}^{\mathcal{F}}_{B} + \mathbf{I}^{\mathcal{F}}_{R}\dot{\boldsymbol{\Omega}}^{\mathcal{F}}_{R},\end{aligned} \tag{10}$$

where $\mathbf{M}^{\mathcal{F}}_{t,xy}$ contains the x and y components of $\mathbf{M}^{\mathcal{F}}_{t}$ with a zero component in z, and $\mathbf{M}^{\mathcal{F}}_{t,z}$ has the z component of $\mathbf{M}^{\mathcal{F}}_{t}$ with zero components in x and y.

Note that the total moments $\mathbf{M}^{\mathcal{F}}_{t,xy}$ consist of the controllable moment $\mathbf{M}^{\mathcal{F}}_{xy}$ and the aerodynamic moment caused by air drag $\mathbf{M}^{\mathcal{F}}_{a,xy}$. We assume that the aerodynamic moments are proportional to the angular velocity, which is $\mathbf{M}^{\mathcal{F}}_{a,xy} = -\mathbf{K}_{a}\boldsymbol{\omega}^{\mathcal{F}}_{\mathcal{F}}$ where $\mathbf{K}_{a} = \mathrm{diag}(k_{a})$ is a diagonal coefficient matrix of aerodynamic moments. In the z axis, there is no controllable moment and only the damping moment caused by the air drag exerted on the body and propeller of the UAV, so we directly simplify $\mathbf{M}^{\mathcal{F}}_{t,z} = \mathbf{M}^{\mathcal{F}}_{z}$. Finally, the rotational dynamic model in the $x-y$ plane is

$$\mathbf{M}^{\mathcal{F}}_{xy} = \lfloor\boldsymbol{\omega}^{\mathcal{F}}_{\mathcal{F}}\rfloor(\mathbf{I}^{\mathcal{F}}_{B}\boldsymbol{\Omega}^{\mathcal{F}}_{B} + \mathbf{I}^{\mathcal{F}}_{R}\boldsymbol{\Omega}^{\mathcal{F}}_{R}) + \mathbf{K}_{a}\boldsymbol{\omega}^{\mathcal{F}}_{\mathcal{F}} + \mathbf{I}^{\mathcal{F}}_{O}\dot{\boldsymbol{\omega}}^{\mathcal{F}}_{\mathcal{F}} \tag{11}$$

and that in the z axis is

$$\mathbf{M}_z^{\mathcal{F}} = \mathbf{I}_B^{\mathcal{F}}\dot{\boldsymbol{\Omega}}_B^{\mathcal{F}} + \mathbf{I}_R^{\mathcal{F}}\dot{\boldsymbol{\Omega}}_R^{\mathcal{F}}. \tag{12}$$

4.1.3 Dynamics of translation and attitude

Defining the system state as

$$\mathbf{x} = [(\mathbf{p}^{\mathcal{I}})^{\mathsf{T}}, (\mathbf{v}^{\mathcal{I}})^{\mathsf{T}}, (\mathbf{n}^{\mathcal{I}})^{\mathsf{T}}]^{\mathsf{T}}, \tag{13}$$

where $\mathbf{p}^{\mathcal{I}}$, $\mathbf{v}^{\mathcal{I}}$, $\mathbf{n}^{\mathcal{I}}$ are the position, velocity, and direction of $\mathcal{F}$ frame's z axis (i.e., a unit vector on S^2 manifold), all represented in $\mathcal{I}$ frame. The system input is

$$\mathbf{u} = [a_T, (\boldsymbol{\omega}_{\mathcal{F}}^{\mathcal{I}})^{\mathsf{T}}]^{\mathsf{T}}, \tag{14}$$

where $a_T \in \mathbb{R}^1$ is the acceleration generated by the propeller's thrust and $\boldsymbol{\omega}_{\mathcal{F}}^{\mathcal{I}} = \mathbf{R}^{\mathcal{I}\mathcal{F}}\boldsymbol{\omega}_{\mathcal{F}}^{\mathcal{F}} \in \mathbb{R}^3$ is the angular velocity of $\mathcal{F}$ frame represented in $\mathcal{I}$ frame. Then, the dynamic model of translation and attitude can be expressed as

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{\mathbf{p}}^{\mathcal{I}} \\ \dot{\mathbf{v}}^{\mathcal{I}} \\ \dot{\mathbf{n}}^{\mathcal{I}} \end{bmatrix} = \begin{bmatrix} \mathbf{v}^{\mathcal{I}} \\ a_T\mathbf{n}^{\mathcal{I}} + \mathbf{g}^{\mathcal{I}} \\ \lfloor \boldsymbol{\omega}_{\mathcal{F}}^{\mathcal{I}} \rfloor \mathbf{n}^{\mathcal{I}} \end{bmatrix} \tag{15}$$

4.2 Differential flatness

Differential flatness means that all the system states and inputs can be expressed as functions of selected flat outputs and their multi-order derivatives. This characteristic reduces the complexity of trajectory optimization. Generally, the number of selected flat outputs should be equal to the DoF of the system input [61, 62]. Although the system input $\boldsymbol{\omega}_{\mathcal{F}}^{\mathcal{I}}$ is a $\mathbb{R}^3$ vector, it only has two DoFs since its direction is always orthogonal to $\mathbf{n}^{\mathcal{I}}$ according to the definition (i.e., $\omega_{\mathcal{F},z}^{\mathcal{F}} = 0$). As a result, the total DoF of inputs is three; hence, we can only choose the flat outputs as the position in the inertial frame $\mathbf{p}^{\mathcal{I}} \in \mathbb{R}^3$. Its multi-order derivatives are velocity $\mathbf{v}^{\mathcal{I}} = \dot{\mathbf{p}}^{\mathcal{I}}$, acceleration $\mathbf{a}^{\mathcal{I}} = \ddot{\mathbf{p}}^{\mathcal{I}}$, and jerk $\mathbf{j}^{\mathcal{I}} = \dddot{\mathbf{p}}^{\mathcal{I}}$.

According to the second row of Equation (15), we can solve

$$a_T = \frac{\|\dot{\mathbf{v}}^{\mathcal{I}} - \mathbf{g}^{\mathcal{I}}\|}{\|\mathbf{n}^{\mathcal{I}}\|} = \|\mathbf{a}^{\mathcal{I}} - \mathbf{g}^{\mathcal{I}}\| \tag{16}$$

and

$$\mathbf{n}^{\mathcal{I}} = \frac{\mathbf{a}^{\mathcal{I}} - \mathbf{g}^{\mathcal{I}}}{\|\mathbf{a}^{\mathcal{I}} - \mathbf{g}^{\mathcal{I}}\|}. \tag{17}$$

Note that the attitude vector $\mathbf{n}^{\mathcal{I}}$ can be arbitrary if the denominator is zero. To avoid a large variation in the attitude, we can set the attitude reference $\mathbf{n}_d^{\mathcal{I}}$ the same as the value of the last calculated result.

Apply derivation on both sides of the second row, obtaining

$$\ddot{\mathbf{v}}^{\mathcal{I}} = \dot{a}_T\mathbf{n}^{\mathcal{I}} + a_T\dot{\mathbf{n}}^{\mathcal{I}}, \tag{18}$$

which is equivalent to

$$\mathbf{j}^{\mathcal{I}} = \dot{a}_T\mathbf{n}^{\mathcal{I}} + a_T\lfloor \boldsymbol{\omega}_{\mathcal{F}}^{\mathcal{I}} \rfloor \mathbf{n}^{\mathcal{I}}. \tag{19}$$

Multiplying $(\mathbf{n}^{\mathcal{I}})^{\mathsf{T}}$ on the left of both sides yields

$$(\mathbf{n}^{\mathcal{I}})^{\mathsf{T}}\mathbf{j}^{\mathcal{I}} = \dot{a}_T \underbrace{(\mathbf{n}^{\mathcal{I}})^{\mathsf{T}}\mathbf{n}^{\mathcal{I}}}_{=1} + a_T \underbrace{(\mathbf{n}^{\mathcal{I}})^{\mathsf{T}}\lfloor \boldsymbol{\omega}_{\mathcal{F}}^{\mathcal{I}} \rfloor \mathbf{n}^{\mathcal{I}}}_{=0}, \tag{20}$$

hence,

$$\dot{a}_T = (\mathbf{n}^{\mathcal{I}})^{\mathsf{T}}\mathbf{j}^{\mathcal{I}}. \tag{21}$$

When $\dot{a}_T$ is determined, substituting it back to Equation (19) leads to

$$\lfloor \boldsymbol{\omega}_{\mathcal{F}}^{\mathcal{I}} \rfloor \mathbf{n}^{\mathcal{I}} = \frac{\mathbf{j}^{\mathcal{I}} - (\mathbf{n}^{\mathcal{I}})^{\mathsf{T}} \mathbf{j}^{\mathcal{I}} \mathbf{n}^{\mathcal{I}}}{a_T}. \tag{22}$$

Because $\boldsymbol{\omega}_{\mathcal{F}}^{\mathcal{I}}$ is always orthogonal to $\mathbf{n}^{\mathcal{I}}$, $\boldsymbol{\omega}_{\mathcal{F}}^{\mathcal{I}}$ can be uniquely determined as

$$\boldsymbol{\omega}_{\mathcal{F}}^{\mathcal{I}} = \lfloor \mathbf{n}^{\mathcal{I}} \rfloor \frac{\mathbf{j}^{\mathcal{I}} - (\mathbf{n}^{\mathcal{I}})^{\mathsf{T}} \mathbf{j}^{\mathcal{I}} \mathbf{n}^{\mathcal{I}}}{a_T} = \frac{\lfloor \mathbf{n}^{\mathcal{I}} \rfloor \mathbf{j}^{\mathcal{I}}}{a_T}. \tag{23}$$

When a_T is zero, we can keep $\boldsymbol{\omega}_{\mathcal{F}}^{\mathcal{I}}$ the same as the last calculated value to avoid singularity.

Now, all states and inputs can be represented as functions of given $\mathbf{p}^{\mathcal{I}}$, $\mathbf{v}^{\mathcal{I}}$, $\mathbf{a}^{\mathcal{I}}$, and $\mathbf{j}^{\mathcal{I}}$. Since $\mathbf{p}^{\mathcal{I}}$, $\mathbf{v}^{\mathcal{I}}$, $\mathbf{a}^{\mathcal{I}}$, and $\mathbf{j}^{\mathcal{I}}$ are the zeroth-order to third-order derivatives of flat outputs $\mathbf{p}^{\mathcal{I}}$, we use the notation $\mathbf{p}^{\mathcal{I},(i:j)}$ to represent derivatives with orders from i to j. Then, combining all component expressions of differential flatness above, we can define the compact expression as

$$\mathbf{x} = \mathcal{D}_x(\mathbf{p}^{\mathcal{I},(0:2)}), \quad \mathbf{u} = \mathcal{D}_u(\mathbf{p}^{\mathcal{I},(2:3)}) \tag{24}$$

where $\mathcal{D}_x(\cdot)$ and $\mathcal{D}_u(\cdot)$ are two functions denoting the state flatness function and input flatness function, respectively.

4.3 Formulation of on-manifold model predictive control

With differential flatness, we can plan the desired trajectory for the UAV in terms of flat outputs and convert them to the reference states and inputs. Building on this prior information about the trajectory, model predictive control (MPC) is a suitable choice for trajectory-tracking control, as it can optimize control inputs based on the future trajectory and predicted future system states over a time horizon, while considering system constraints.

We use the unit vector $\mathbf{n}_k^{\mathcal{I}}$ to represent the attitude of the UAV since there is no need to control the UAV's z axis due to free self-rotation. According to the definition of S^2 manifold (i.e., a sphere), the attitude state $\mathbf{n}^{\mathcal{I}}$ in Equation (15) is a point on it. Utilizing standard MPC formulation directly based on the system's original states and inputs, additional constraint has to be introduced to ensure that $\mathbf{n}^{\mathcal{I}}$ is always on the manifold (i.e., $\|\mathbf{n}^{\mathcal{I}}\| = 1$). Also, representing $\mathbf{n}^{\mathcal{I}}$ by a 3D unit vector leads to an over-parameterization. These two factors increase the complexity of the optimization problem. Hence, we use the error state of attitude defined in the tangent space of the S^2 manifold to eliminate the manifold constraint and to achieve minimum parameterization.

The illustration of the S^2 manifold and its $\mathbb{R}^2$ tangent space is shown in Fig. 10b. The tangent space of S^2 at the point of $\mathbf{n}_k^{\mathcal{I}}$ (the subscript k indicates the discretized state) is a 2D plane orthogonal to $\mathbf{n}_k^{\mathcal{I}}$. $\mathbf{b}_1$ and $\mathbf{b}_2$ are two orthogonal basis on this plane and can be defined as differentiable functions of $\mathbf{n}_k^{\mathcal{I}}$. Given another point $\mathbf{n}_{k+1}^{\mathcal{I}}$ on the S^2 manifold, the error $\delta\mathbf{n}_k^{\mathcal{I}} \in \mathbb{R}^2$ between $\mathbf{n}_k^{\mathcal{I}}$ and $\mathbf{n}_{k+1}^{\mathcal{I}}$ can be represented in the tangent space orthogonal to $\mathbf{n}_k^{\mathcal{I}}$ by defining a $\boxminus$ operation for the S^2 manifold leading to $\delta\mathbf{n}_k^{\mathcal{I}} = \mathbf{n}_{k+1}^{\mathcal{I}} \boxminus \mathbf{n}_k^{\mathcal{I}}$. Meanwhile, given $\mathbf{n}_k^{\mathcal{I}} \in S^2$ and $\delta\mathbf{n}_k^{\mathcal{I}} \in \mathbb{R}^2$, we can define the $\boxplus$ operation to calculate $\mathbf{n}_{k+1}^{\mathcal{I}} = \mathbf{n}_k^{\mathcal{I}} \boxplus \delta\mathbf{n}_k^{\mathcal{I}}$. (See Supplementary Note 3.3 for more details about the definition of $\boxplus$ and $\boxminus$).

Here, we established the bidirectional mapping between the S^2 manifold and its tangent space. With the defined error state of attitude, the constraint of the manifold (i.e., $\|\mathbf{n}_k^{\mathcal{I}}\| = 1$) can be eliminated such that we can optimize the error state of attitude in the Euclidean space (i.e., the tangent space $\in \mathbb{R}^2$). Then, we modeled the dynamic system through error states and linearized it at the current state (i.e., zero values of error states), and finally lead to a linear, minimum-parameterized, error-state dynamic system without manifold constraint (see Supplementary Note 3 for more details).

To control the system mentioned above, a MPC problem can be formulated as

$$\begin{aligned}
&\min_{\delta\mathbf{u}_k} \sum_{k=0}^{N-1} \left(\|\delta\mathbf{x}_k\|^2_{\mathbf{Q}_k} + \|\delta\mathbf{u}_k\|^2_{\mathbf{R}_k}\right) + \|\delta\mathbf{x}_N\|^2_{\mathbf{Q}_N} \\
&\text{s.t.} \quad \delta\mathbf{x}_{k+1} = \mathbf{F}_{\mathbf{x}_k}\delta\mathbf{x}_k + \mathbf{F}_{\mathbf{u}_k}\delta\mathbf{u}_k, \quad \delta\mathbf{x}_0 = \delta\mathbf{x}_{\text{init}} \\
&\delta\mathbf{u}_k \in \mathbb{U}, \quad k = 0, \ldots, N-1.
\end{aligned} \tag{25}$$

where $\mathbf{Q}_k$ and $\mathbf{R}_k$ are the weighting matrices for the error states and error inputs, respectively, $\mathbb{U} = \{\delta\mathbf{u}_k \in \mathbb{R}^3 \mid \mathbf{u}_{\min} \leq \mathbf{u}_{d,k} + \delta\mathbf{u}_k \leq \mathbf{u}_{\max}\}$ being the input constraints, $\delta\mathbf{x}_{\text{init}}$ being the initial error state, $\delta\mathbf{x}_N$ being the terminal error state. $\delta\mathbf{x}_k$ is the system error state and $\delta\mathbf{u}_k$ is the system input that is optimized within the prediction horizon N. As shown, the MPC problem has a standard form. It is easy to solve, because it is formulated upon the linear, minimum-parameterized, error-state dynamic system without manifold constraint obtained above. In experiments, we solve this optimal control problem (OCP) onboard the UAV by quadratic programming (QP) solvers [63].

When the optimization problem is solved successfully, the optimal solution $[\delta\mathbf{u}_k^*, ..., \delta\mathbf{u}_{k+N}^*]$ with the length of the prediction horizon N can be obtained. The first element of the optimal solution will be used to calculate the actual value of system input (i.e., control effort) as $\mathbf{u}_k = \mathbf{u}_{k,d} \boxplus \delta\mathbf{u}_k^*$ where $\mathbf{u}_{k,d}$ is the reference trajectory of the system input.

4.4 Offline trajectory generation for trajectory tracking

The trajectory optimization is to generate a smooth and dynamically feasible trajectory for the UAV. The 3D trajectory $\mathbf{p}^{\mathcal{I}}(t) : \mathbb{R} \in [0, T_f] \mapsto \mathbb{R}^3$ is a multistage polynomial trajectory, and each stage of the trajectory $\mathbf{p}_j^{\mathcal{I}}(t) : \mathbb{R} \in [0, t_j] \mapsto \mathbb{R}^3$ is a function of time t with a duration t_j. The trajectory is required to pass a sequence of waypoints $\mathbf{Q}_w = (\mathbf{q}_1, ..., \mathbf{q}_N)$ for the purposes of: (i) offline trajectory tracking, the waypoints can shape the flight trajectory so that the UAV can perform a desired flight; (ii) for autonomous navigation in environments with obstacles, the waypoints can guide the path search direction of the A* algorithm and further generate a safe flight corridor based on some key points on the path to constrain the trajectory within the corridor [64–66]. Since the trajectory is a function of t, a waypoint $\mathbf{q}_i$ is associated with a time $t_{\mathbf{q}_i}$. The one-to-one time sequence associated with the waypoint sequence satisfies $0 \leq t_{\mathbf{q}_1} < t_{\mathbf{q}_2} < ... < t_{\mathbf{q}_N} \leq T_f$.

The goal of trajectory optimization is to minimize the control effort $\mathbf{u}$ to reduce energy consumption and the duration time T_f to ensure a suitable flight speed while satisfying some constraints. Hence, we formulated the optimization problem as

$$\min_{\mathbf{p}^{\mathcal{I}}(t), T_f} \int_0^{T_f} \|\mathbf{u}(t)\|_{\mathbf{W}_u} dt + \rho T_f \tag{26a}$$

$$\text{s.t.} \quad \mathbf{x}(t) = \mathcal{D}_x(\mathbf{p}^{\mathcal{I},(0:2)}(t)),\ \mathbf{u}(t) = \mathcal{D}_u(\mathbf{p}^{\mathcal{I},(2:3)}(t)) \tag{26b}$$

$$\mathbf{p}^{\mathcal{I},(0:3)}(0) = \mathbf{s}_0,\ \mathbf{p}^{\mathcal{I},(0:3)}(T_f) = \mathbf{s}_f \tag{26c}$$

$$\mathbf{p}_j^{\mathcal{I},(0:3)}(t_j) = \mathbf{p}_{j+1}^{\mathcal{I},(0:3)}(0),\ \forall j = 1, ..., M-1 \tag{26d}$$

$$t_j < t_{j+1},\ \forall j = 1, ..., M-1 \tag{26e}$$

$$\mathbf{p}^{\mathcal{I}}(t_{\mathbf{q}_i}) = \mathbf{q}_i,\ i = 1, ..., N \tag{26f}$$

$$\mathbf{x}^{\mathcal{I}}(t) \in \mathbb{X},\ \mathbf{u}^{\mathcal{I}}(t) \in \mathbb{U} \tag{26g}$$

where $\mathbf{W}_u \in \mathbb{R}^{3\times3}$ is a positive diagonal matrix penalizing the control effort $\mathbf{u}(t)$, $\rho > 0$ is the penalty of duration time T_f, N is the total number of waypoints, and M is the total stage number of polynomial trajectories.

The optimization problem shown in Equation (26) is a nonlinear constrained optimization problem with 6 constraints. The first one (26b) is the differential flatness transform mentioned in Equation (24). The second one (26c) is the initial state $\mathbf{s}_0 \in \mathbb{R}^{12}$ and the terminal state $\mathbf{s}_f \in \mathbb{R}^{12}$ for the flat outputs and their derivatives.

The notation $\mathbf{p}^{\mathcal{I},(0:3)}$ represents the derivatives with orders from 0 to 3. The third one (26d) and fourth one (26e) ensure the continuity of each of the two stages of trajectories. The fifth one (26f) is the constraint of waypoints to ensure that the trajectory passes through the waypoints at specified times. The sixth one (26g) is the boundary constraints for both state and control effort, where $\mathbb{X} = \{\mathbf{x} \in \mathbb{R}^8 \mid \mathbf{x}_{\min} \leq \mathbf{x} \leq \mathbf{x}_{\max}\}$ and $\mathbb{U} = \{\mathbf{u} \in \mathbb{R}^3 \mid \mathbf{u}_{\min} \leq \mathbf{u} \leq \mathbf{u}_{\max}\}$ are the feasible sets of state and control effort, respectively. We solved this optimization problem by leveraging a state-of-the-art flight trajectory optimization framework, MINCO [58].

4.5 Actuator modeling and compensation

The actuator, a motor with SLM, generates both thrust and moments. However, the characteristics of moment generation (e.g., moment magnitude and delay angle of moment direction) depend on the input of sinusoidal magnitude and the current motor speed. In previous work [28], the actuator model did not consider such characteristics dynamically, leading to obvious errors in the magnitude and direction of the moment under different operating conditions. As a result, the UAV tends to deviate from the desired trajectory, especially in high-speed trajectory tracking with a wide changing range of motor speed, resulting in insufficient agility. To solve this problem, we continue to adopt a data-driven approach to establish the actuator model, as it can directly capture the system's main dynamics in actual working conditions without building a complex dynamic model containing nonlinear, high-coupling terms of the swashplateless mechanism. The remaining unmodeled dynamics are expected to be further handled by the DOB design.

To collect data, we first built a test stand as shown in Supplementary Fig. 2, which can measure the thrust and moment generated by SLM as well as the rotor angle, rotor speed, input voltage, and current of the motor, with time synchronization for all data (see Supplementary Note 2). Then, with sufficient data collected from the test stand, we established the static model through data fitting and the dynamic model via system identification. For the static model, we modeled two functions in polynomial forms. The first is that the coefficient of moment magnitude is modeled as a 2D nonlinear function of both the motor speed and the inputted amplitude of sinusoidal voltage. The second is that the delay angle of the moment direction is modeled as a 1D function of motor speed. Then, the unknown parameters of the polynomial functions are determined by solving the optimization problem with high accuracy. This static model is finally used in the low-level control system to compensate for the nonlinearity of the actuator.

For dynamic modeling, we assumed the dynamic actuator model is linear and identified it based on the data obtained from the test stand (see Supplementary Note 2). The identification result is a transfer function with a non-minimum phase characteristic, which is used for the DOB design to enhance disturbance rejection performance (see Supplementary Note 7).

4.6 Disturbance observation

4.6.1 Overview of disturbance observer

To enable collision resistance, in addition to the structural design of the protection frame, the control system's robustness against external disturbances during collisions should also be enhanced. DOB is an inner-loop output-feedback block capable of actively estimating and compensating for both external disturbances and internal model uncertainties in real time, which means that the closed-loop system with the DOB can approximate the system performance of the baseline controller and the nominal plant model. It is adopted for the following reasons: (i) its modular architecture enables seamless integration into existing low-level control system; (ii) its frequency-domain-based design aligns well with the identified frequency-domain model of the actuator; (iii) DOB can handle a part of model uncertainties existing in the identified actuator model; and (iv) it has good compatibility with MIMO systems, effectively mitigating cross-coupling effects. Hence, a DOB is designed in the frequency domain and embedded in the control system of angular velocity, as shown in Fig. 10c. Since

the gyroscopic effect introduces cross-axis coupling in the rotational dynamics of the x and y axes, we formulated the disturbance observer as a MIMO system.

For an external disturbance moment, its primary component is generally distributed in a range of low frequencies. However, when this signal is represented in the body frame $\mathcal{B}$, it couples with a frequency of self-rotation. The coupled frequency due to self-rotation increases the frequency distribution of the signal to a higher range, resulting in a performance reduction of the DOB. Hence, we designed the DOB using the dynamic model established in the fictional frame $\mathcal{F}$, which eliminates the coupling effect, allowing the DOB to effectively attenuate the original frequency component of the external disturbance moment.

4.6.2 Structure of disturbance observer

The structure of DOB is shown in Fig. 10c, highlighted with an orange background. All inputs, outputs, and internal dynamics of the system have 2×2 multi-input-multi-output (MIMO) form. The three inputs are the reference input of angular velocity r, the disturbance input d, and the noise input n. The only output is the system output y (i.e., the actual angular velocity). For internal dynamics, $\mathbf{G}(s)$ is a plant model with parameter uncertainties (see Supplementary Table 1) while $\mathbf{G}_n(s)$ is a nominal model without uncertainty. $\mathbf{C}(s)$ is a designed proportional-integral (PI) controller with tuned parameters ($\mathbf{K}_p = \text{diag}([0.19, 0.19]), \mathbf{K}_i = \text{diag}([0.25, 0.25])$) according to simulations based on the nominal model and actual flight performance. $\mathbf{L}(s)$ is a second-order Butterworth low-pass filter with a cut-off frequency of 40 Hz. This setting is to attenuate the primary vibration caused by the propeller with a frequency range from about 58 Hz to 134 Hz (i.e., 3500 rpm to 8000 rpm). There are two $\mathbf{L}(s)$ existing in the system. One is used to filter out the noise of y_n while the other is to compensate for the phase of u_e such that it synchronizes with y_{nl}, which is crucial to DOB design.

$\mathbf{Q}(s)$ is also a low-pass filter to ensure that $\mathbf{Q}(s)(\mathbf{G}_n(s))^{-1}$ is causal and further suppresses the noise n. The poles of $\mathbf{Q}(s)$ are key for the DOB design. Therefore, we optimized the parameters of $\mathbf{Q}(s)$ with the H-infinity method, resulting in robust and quantifiable performance. More detailed working principles of DOB can be found in the Supplementary Notes 7.

4.7 μ synthesis for designing disturbance observer

4.7.1 Structure of generalized system

The poles of the low-pass filter $\mathbf{Q}(s)$ (i.e., the coefficients c_i in Supplementary Equation (S55)) are very crucial for the performance of DOB. An option to determine coefficients is to tune them manually, but this process is inefficient and cannot guarantee the robustness of the uncertainty of the plant model. μ synthesis is a synthesis method for uncertain models that combines two modules, H-infinity synthesis and μ analysis, to optimize the closed-loop robust performance under specified model uncertainties [67]. We adopt the μ analysis to systematically optimize the coefficients of $\mathbf{Q}(s)$, which is efficient, robust, and quantifiable.

We first restructured the angular velocity control system shown in Fig. 10c to a generalized system shown in Fig. 10d, such that it becomes a suitable structure for μ synthesis. The external loop of the angular velocity control system constitutes a generalized plant model $\mathbf{P}(s)$ (purple area) and the DOB structure is extracted as a virtual controller $\mathbf{K}(s)$ (orange area). The generalized system has four sets of signals: the generalized disturbance input $\mathbf{W}(s)$, error output $\mathbf{Z}(s)$, generalized system input $\mathbf{U}(s)$, and generalized system output $\mathbf{Y}(s)$. More specifically, $\mathbf{W}(s)$ contains two signals, which are noise input n_w and disturbance input d_w. $\mathbf{Z}(s)$ comprises disturbance observation error d_{error} which is a new evaluation signal used in μ synthesis. $\mathbf{Y}(s)$ includes filtered output with noise y_{nl} and desired actuation u_d, which are also two input signals of DOB. $\mathbf{U}(s)$ is equivalent to the observed disturbance d_{est} outputted from DOB.

4.7.2 Weight function and H-infinity optimization

We defined two transfer function matrices. The first is $\mathbf{T}_d(s)$ that describes the relation from d to d_{est} and the second is $\mathbf{T}_n(s)$ that describes the relation from n to d_{est}. Ideally, we expect them all to have small gains to ensure the performance of disturbance rejection and noise attenuation for the whole frequency range. However, we cannot achieve it at the same time, as better disturbance rejection performance leads to worse noise attenuation performance (see the Supplementary Note 7). In the actual system, the disturbance is mainly a low-frequency signal while the noise is mainly a high-frequency signal, which requires a suitable $Q(s)$ to ensure that $\mathbf{T}_d(s)$ has small gains in the low-frequency range and $\mathbf{T}_n(s)$ has small gains in the high-frequency range.

To achieve that, two frequency-dependent weighting functions $\mathbf{W}_d(s) = \text{diag}([W_d(s), W_d(s)])$ and $\mathbf{W}_n(s) = \text{diag}([W_n(s), W_n(s)])$ are introduced, and their shape of the Bode diagram can be freely specified. As shown in Fig. 10d, the $\mathbf{K}(s)$ has a fixed structure with adjustable poles of $Q(s)$ (noted $Q_D(s)$). Then we can construct the H-infinity optimization problem as

$$\min_{Q_D(s)} \left\| \begin{bmatrix} \mathbf{W}_d(s)\mathbf{T}_d(s) \\ \mathbf{W}_n(s)\mathbf{T}_n(s) \end{bmatrix} \right\|_\infty . \tag{27}$$

To ensure that $\mathbf{K}(s)$ is causal, we fixed the order of $Q_D(s)$ to 3, which is equal to the order difference between the numerator and the denominator of $\mathbf{G}_n^{-1}(s)$.

According to Equation (27), a higher weighting function gain leads to a lower weighting function gain at a certain frequency. We designed the shapes of the Bode diagram of the weighting functions $W_d(s)$ and $W_n(s)$ with the following considerations: (i) $W_d(s) = \frac{6.31s+3.914\times10^{-4}}{s+39.14}$, its reciprocal $\frac{1}{W_d(s)}$ has small gains in the low-frequency range with a cross frequency of 1 Hz to ensure disturbance rejection performance, and (ii) $W_n(s) = \frac{1\times10^{-5}s^2+0.7166s+2.568\times10^4}{s^2+302.2s+4.566\times10^4}$, its reciprocal $\frac{1}{W_n(s)}$ has small gains and a slope of -40 dB in the high-frequency range, with -15.3 dB attenuation at 60 Hz. This is to ensure noise attenuation performance because the most obvious noise source is propeller rotation with a frequency range from 58 to 133 Hz. The Bode diagrams of the reciprocals of the weighting functions are shown in Supplementary Fig. 10.

With the designed weighting functions, the optimization problem in Equation (27) with an uncertain plant model $\mathbf{G}(s)$ can be solved by μ synthesis with the function *musyn* of the MATLAB Robust Control Toolbox. μ synthesis first finds a suitable $Q_D(s)$ that minimizes the gain of the closed-loop system with a nominal plant model using H-infinity synthesis. Then, it utilizes an iterative process called D-K iteration to solve a sequence of H-infinity problems with uncertainty. The D step performs a robustness analysis to estimate the robust H-infinity performance of the closed-loop system, with the quantity expressed as a scaled H-infinity norm which considers the uncertainty. The K step uses the H-infinity synthesis to find a controller that minimizes the H-infinity norm in the D step. These two steps are repeated until robust performance improvements stop. More detailed optimization results of μ synthesis can be found in Supplementary Notes 8.

Data availability

Source data are provided with this paper.

Code availability

Source code and design files of this work are available on the Code Ocean platform and the GitHub repository: https://github.com/hku-mars/PULSAR2.

References

[1] Petráček, P., Krátkỳ, V., Petrlík, M., Báča, T., Kratochvíl, R., Saska, M.: Large-scale exploration of cave environments by unmanned aerial vehicles. IEEE Robotics and Automation Letters **6**(4), 7596–7603 (2021)

[2] Zhao, Y., Yan, L., Xie, H., Dai, J., Wei, P.: Autonomous exploration method for fast unknown environment mapping by using uav equipped with limited fov sensor. IEEE Transactions on Industrial Electronics **71**(5), 4933–4943 (2023)

[3] Tian, Y., Liu, K., Ok, K., Tran, L., Allen, D., Roy, N., How, J.P.: Search and rescue under the forest canopy using multiple uavs. The International Journal of Robotics Research **39**(10-11), 1201–1221 (2020)

[4] Tan, Y., Li, S., Liu, H., Chen, P., Zhou, Z.: Automatic inspection data collection of building surface based on bim and uav. Automation in Construction **131**, 103881 (2021)

[5] Car, M., Markovic, L., Ivanovic, A., Orsag, M., Bogdan, S.: Autonomous wind-turbine blade inspection using lidar-equipped unmanned aerial vehicle. IEEE access **8**, 131380–131387 (2020)

[6] Castelar Wembers, C., Pflughaupt, J., Moshagen, L., Kurenkov, M., Lewejohann, T., Schildbach, G.: Lidar-based automated uav inspection of wind turbine rotor blades. Journal of Field Robotics **41**(4), 1116–1132 (2024)

[7] Bolourian, N., Hammad, A.: Lidar-equipped uav path planning considering potential locations of defects for bridge inspection. Automation in Construction **117**, 103250 (2020)

[8] Zhou, L., Gu, X., Cheng, S., Yang, G., Shu, M., Sun, Q.: Analysis of plant height changes of lodged maize using uav-lidar data. Agriculture **10**(5), 146 (2020)

[9] Moreno, H., Valero, C., Bengochea-Guevara, J.M., Ribeiro, Á., Garrido-Izard, M., Andújar, D.: On-ground vineyard reconstruction using a lidar-based automated system. Sensors **20**(4), 1102 (2020)

[10] Alsalam, B.H.Y., Morton, K., Campbell, D., Gonzalez, F.: Autonomous uav with vision based on-board decision making for remote sensing and precision agriculture. In: 2017 IEEE Aerospace Conference, pp. 1–12 (2017). IEEE

[11] Roelofsen, S., Martinoli, A., Gillet, D.: 3d collision avoidance algorithm for unmanned aerial vehicles with limited field of view constraints. In: 2016 IEEE 55th Conference on Decision and Control (CDC), pp. 2555–2560 (2016). IEEE

[12] Miiller, M., Steidle, F., Schuster, M.J., Lutz, P., Maier, M., Stoneman, S., Tomic, T., Stürzl, W.: Robust visual-inertial state estimation with multiple odometries and efficient mapping on an mav with ultra-wide fov stereo vision. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3701–3708 (2018). IEEE

[13] Gao, W., Wang, K., Ding, W., Gao, F., Qin, T., Shen, S.: Autonomous aerial robot using dual-fisheye cameras. Journal of Field Robotics **37**(4), 497–514 (2020)

[14] Mohsan, S.A.H., Othman, N.Q.H., Li, Y., Alsharif, M.H., Khan, M.A.: Unmanned aerial vehicles (uavs): Practical aspects, applications, open challenges, security issues, and future trends. Intelligent service robotics **16**(1), 109–137 (2023)

[15] Gurtner, A., Greer, D.G., Glassock, R., Mejias, L., Walker, R.A., Boles, W.W.: Investigation of fish-eye lenses for small-uav aerial photography. IEEE Transactions on Geoscience and Remote Sensing **47**(3), 709–721 (2009)

[16] Lin, Y., Gao, F., Qin, T., Gao, W., Liu, T., Wu, W., Yang, Z., Shen, S.: Autonomous aerial navigation using monocular visual-inertial fusion. Journal of Field Robotics **35**(1), 23–51 (2018)

[17] Respall, V.M., Devitt, D., Fedorenko, R.: Unmanned aerial vehicle path planning for exploration mapping. In: 2020 International Conference Nonlinearity, Information and Robotics (NIR), pp. 1–6 (2020). IEEE

[18] Yin, L., Zhu, F., Ren, Y., Kong, F., Zhang, F.: Decentralized swarm trajectory generation for lidar-based aerial tracking in cluttered environments. In: 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 9285–9292 (2023). IEEE

[19] Harmat, A., Trentini, M., Sharf, I.: Multi-camera tracking and mapping for unmanned aerial vehicles in unstructured environments. Journal of Intelligent & Robotic Systems **78**(2), 291–317 (2015)

[20] Wierzbicki, D.: Multi-camera imaging system for uav photogrammetry. Sensors **18**(8), 2433 (2018)

[21] Gohl, P., Honegger, D., Omari, S., Achtelik, M., Pollefeys, M., Siegwart, R.: Omnidirectional visual obstacle detection using embedded fpga. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3938–3943 (2015). IEEE

[22] Zhou, G., Fang, L., Tang, K., Zhang, H., Wang, K., Yang, K.: Guidance: A visual sensing platform for robotic applications. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 9–14 (2015)

[23] Zhao, S., Zhang, H., Wang, P., Nogueira, L., Scherer, S.: Super odometry: Imu-centric lidar-visual-inertial estimator for challenging environments. In: 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 8729–8736 (2021). IEEE

[24] MacAllister, B., Butzke, J., Kushleyev, A., Pandey, H., Likhachev, M.: Path planning for non-circular micro aerial vehicles in constrained environments. In: 2013 Ieee International Conference on Robotics and Automation, pp. 3933–3940 (2013). IEEE

[25] Son, J.-H., Kim, D., Cha, J.: Autonomous structural inspection with an octocopter uav: Integration of 3d lidar and gimbal-mounted camera. In: 2024 IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 2034–2040 (2024). IEEE

[26] Jacobsson, M., Willén, J., Swarén, M.: A drone-mounted depth camera-based motion capture system for sports performance analysis. In: International Conference on Human-Computer Interaction, pp. 489–503 (2023). Springer

[27] García-López, S., Vélez-Nicolás, M., Zarandona-Palacio, P., Curcio, A.C., Ruiz-Ortiz, V., Barbero, L.: Uav-borne lidar revolutionizing groundwater level mapping. Science of the Total Environment **859**, 160272 (2023)

[28] Chen, N., Kong, F., Xu, W., Cai, Y., Li, H., He, D., Qin, Y., Zhang, F.: A self-rotating, single-actuated uav with extended sensor field of view for autonomous navigation. Science Robotics **8**(76), 4538 (2023)

[29] Jameson, S., Fregene, K., Chang, M., Allen, N., Youngren, H., Scroggins, J.: Lockheed martin's samarai nano air vehicle: challenges, research, and realization. In: 50th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition, p. 584 (2012)

[30] Yoon, S., Lee, C., Son, Y., Son, J.J., Han, S.: Spinning a small-sized quadrotor efficiently to achieve a 3d omnidirectional field of view. IEEE Transactions on Aerospace and Electronic Systems (2024)

[31] Bai, S., Chirarattananon, P.: Design and take-off flight of a samara-inspired revolving-wing robot. In: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 6070–6076 (2019). IEEE

[32] Bai, S., He, Q., Chirarattananon, P.: A bioinspired revolving-wing drone with passive attitude stability and efficient hovering flight. Science Robotics **7**(66), 5913 (2022)

[33] Safaee, A., Moussavi, S.Z., Menhaj, M.B.: Design and construction of monocopter and its nonlinear control using photo diode array. U. Porto Journal of Engineering **4**(2), 34–41 (2018)

[34] Ulrich, E.R., Pines, D.J., Humbert, J.S.: From falling to flying: the path to powered flight of a robotic samara nano air vehicle. Bioinspiration & biomimetics **5**(4), 045009 (2010)

[35] Fregene, K., Jameson, S., Sharp, D., Youngren, H., Stuart, D.: Development and flight validation of an autonomous mono-wing uas. In: American Helicopter Society Forum, Phoenix, AZ (2010)

[36] Isaacs, J.T., Magee, C., Subbaraman, A., Quitin, F., Fregene, K., Teel, A.R., Madhow, U., Hespanha, J.P.: Gps-optimal micro air vehicle navigation in degraded environments. In: 2014 American Control Conference, pp. 1864–1871 (2014). IEEE

[37] Sharp, D., Stoneking, C., Fregene, K.: Micro air vehicle based navigation aiding in degraded environments. In: 2016 IEEE/ION Position, Location and Navigation Symposium (PLANS), pp. 305–312 (2016). IEEE

[38] Bhardwaj, H., Cai, X., Win, L.S.T., Foong, S.: Nature-inspired in-flight foldable rotorcraft. Bioinspiration & Biomimetics **18**(4), 046012 (2023)

[39] Win, S.K.H., Win, L.S.T., Sufiyan, D., Soh, G.S., Foong, S.: An agile samara-inspired single-actuator aerial robot capable of autorotation and diving. IEEE Transactions on Robotics (2021)

[40] Piccoli, M., Yim, M.: Passive stability of a single actuator micro aerial vehicle. In: 2014 IEEE International Conference on Robotics and Automation (ICRA), pp. 5510–5515 (2014). IEEE

[41] Shen, W., Peng, J., Ma, R., Wu, J., Li, J., Liu, Z., Leng, J., Yan, X., Qi, M.: Sunlight-powered sustained flight of an ultralight micro aerial vehicle. Nature **631**(8021), 537–543 (2024)

[42] Piccoli, M., Yim, M.: Piccolissimo: The smallest micro aerial vehicle. In: 2017 IEEE International Conference on Robotics and Automation (ICRA), pp. 3328–3333 (2017). IEEE

[43] Zhang, W., Mueller, M.W., D'Andrea, R.: A controllable flying vehicle with a single moving part. In: 2016 IEEE International Conference on Robotics and Automation (ICRA), pp. 3275–3281 (2016). IEEE

[44] Wang, J., Curtis, A.G., Yim, M., Rubenstein, M.: A single motor nano aerial vehicle with novel peer-to-peer communication and sensing mechanism. arXiv preprint arXiv:2405.14144 (2024)

[45] Win, L.S.T., Win, S.K.H., Sufiyan, D., Soh, G.S., Foong, S.: Achieving efficient controlled flight with a single actuator. In: 2020 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM), pp. 1625–1631 (2020). IEEE

[46] Win, S.K.H., Win, L.S.T., Sufiyan, D., Foong, S.: Design and control of the first foldable single-actuator rotary wing micro aerial vehicle. Bioinspiration & biomimetics **16**(6), 066019 (2021)

[47] Lu, G., Xu, W., Zhang, F.: On-manifold model predictive control for trajectory tracking on robotic systems. IEEE Transactions on Industrial Electronics **70**(9), 9192–9202 (2022)

[48] Zheng, M., Lyu, X., Liang, X., Zhang, F.: A generalized design method for learning-based disturbance observer. IEEE/ASME Transactions on Mechatronics **26**(1), 45–54 (2020)

[49] Zhou, S., Zheng, M., Zhang, F., Tomizuka, M.: Synthesized disturbance observer for vehicle lateral disturbance rejection. In: 2018 Annual American Control Conference (ACC), pp. 398–403 (2018). IEEE

[50] Khoshnazar, M., Barjini, A.H., Moradi, H.: Using a robust mu-synthesis based controller to eliminate the adverse effects of uncertainties and external disturbances in nonlinear 3d overhead cranes with hoisting mechanism. Meccanica **59**(9), 1517–1538 (2024)

[51] Paulos, J., Yim, M.: An underactuated propeller for attitude control in micro air vehicles. In: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1374–1379 (2013). IEEE

[52] Nxt-FC Flight Controller (2024). https://github.com/HKUST-Aerial-Robotics/Nxt-FC Accessed 2025-02-07

[53] PX4 Autopilot Software (2025). https://github.com/PX4/PX4-Autopilot Accessed 2025-05-08

[54] Zhu, F., Ren, Y., Kong, F., Wu, H., Liang, S., Chen, N., Xu, W., Zhang, F.: Swarm-lio: Decentralized swarm lidar-inertial odometry. In: 2023 IEEE International Conference on Robotics and Automation (ICRA), pp. 3254–3260 (2023). IEEE

[55] MAVROS, extendable communication node for ROS (2024). https://github.com/mavlink/mavros Accessed 2025-04-27

[56] Zhu, F., Ren, Y., Zhang, F.: Robust real-time lidar-inertial initialization. In: 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3948–3955 (2022). IEEE

[57] Ren, Y., Zhu, F., Lu, G., Cai, Y., Yin, L., Kong, F., Lin, J., Chen, N., Zhang, F.: Safety-assured high-speed navigation for mavs. Science Robotics **10**(98), 6187 (2025)

[58] Wang, Z., Zhou, X., Xu, C., Gao, F.: Geometrically constrained trajectory optimization for multicopters. IEEE Transactions on Robotics **38**(5), 3259–3278 (2022)

[59] Hesai Technology: JT128 - Hesai Technology. https://www.hesaitech.com/product/jt128/ Accessed 2025-11-20

[60] Exyn Technologies: Robotic Autonomy - Exyn Technologies. https://www.exyn.com/purchase?hsCtaAttrib=190420223306 Accessed 2026-02-11

[61] Van Nieuwstadt, M.J., Murray, R.M.: Real-time trajectory generation for differentially flat systems. International Journal of Robust and Nonlinear Control: IFAC-Affiliated Journal **8**(11), 995–1020 (1998)

[62] Mellinger, D., Kumar, V.: Minimum snap trajectory generation and control for quadrotors. In: 2011 IEEE International Conference on Robotics and Automation, pp. 2520–2525 (2011). IEEE

[63] Frison, G., Diehl, M.: Hpipm: a high-performance quadratic programming framework for model predictive control. IFAC-PapersOnLine **53**(2), 6563–6569 (2020)

[64] Liu, S., Watterson, M., Mohta, K., Sun, K., Bhattacharya, S., Taylor, C.J., Kumar, V.: Planning dynamically feasible trajectories for quadrotors using safe flight corridors in 3-d complex environments. IEEE Robotics and Automation Letters **2**(3), 1688–1695 (2017)

[65] Gao, F., Wu, W., Gao, W., Shen, S.: Flying on point clouds: Online trajectory generation and autonomous navigation for quadrotors in cluttered environments. Journal of Field Robotics **36**(4), 710–733 (2019)

[66] Ren, Y., Zhu, F., Liu, W., Wang, Z., Lin, Y., Gao, F., Zhang, F.: Bubble planner: Planning high-speed smooth quadrotor trajectories using receding corridors. In: 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 6332–6339 (2022). IEEE

[67] Das, E.: Combined design of robust controller and disturbance observer in a fixed-order H∞ control framework. Mechatronics **88**, 102912 (2022)

[68] He, D., Xu, W., Zhang, F.: Symbolic representation and toolkit development of iterated error-state extended kalman filters on manifolds. IEEE Transactions on Industrial Electronics **70**(12), 12533–12544 (2023)

[69] He, D., Xu, W., Zhang, F.: Kalman filters on differentiable manifolds. arXiv preprint arXiv:2102.03804 (2021)

[70] Marshalling / communication library for drones (2024). https://github.com/mavlink/mavlink Accessed 2025-04-27

[71] Wang, L., Su, J.: Disturbance rejection control for non-minimum phase systems with optimal disturbance observer. ISA transactions **57**, 1–9 (2015)

Funding statements

F.Zhang discloses support for the research of this work from Hong Kong Research Grants Council (RGC) General Research Fund (grant number 17205924).

Author contributions

N.C. proposed the initial idea of the research. N.C. and H.L. designed and manufactured the UAV prototype. Y.R. developed the simulation software and the initial version of the controller. N.C. formulated models, designed system framework, developed remaining modules, conducted simulations, and integrated all modules on the UAV prototype with the help of Y.R., G.L., F.Zhu, and J.L. The experiments were designed by N.C. and performed by N.C. and H.L. N.C. wrote the manuscript with the advice of all authors. F.Zhang provided funding and supervised the research.

Competing interests

The authors declare no competing interests.

Fig. 1: Overview of the single-actuator UAV. **a** (i-ii) PULSAR II can fly autonomously in dense woods with clutter and fly in a parking lot at high speed of over 6 m s^{-1}. (iii-iv) PULSAR II has high agility that tracks aggressive flip trajectory, and collision resistance that pushes obstacles away during forward flight. **b** (i) The inherent self-rotation can naturally extend the original LiDAR field of view (FoV) (blue area) to the omnidirectional FoV (yellow area). (ii) Before take-off, the original FoV can observe only a portion of the environment such that the built point cloud map is obviously incomplete. (iii) After take-off, the omnidirectional FoV due to self-rotation enables PULSAR II to build a complete point cloud map covering the whole environment without any unobserved area.

Fig. 2: Mechanical structure and components of PULSAR II. **a** Overall structure and information of the main components including weight, brand, type, specification, or materials. **b** Detailed structure of the propulsion system and swashplateless mechanism. **c** Electronic components onboard the UAV.

Fig. 3: Control system structure. **a** Overall structure of the control system. **b** Detailed structure of the low-level control.

Fig. 4: Ablation experiments for evaluation of actuator compensation through trajectory tracking. **a** Overlaid photos of the tracking of helix trajectory when actuator compensation is enabled. (i) Front view. (ii) Side view. **b** Flight data for tracking a helix trajectory. The legends of "w/o comp.", "w/ comp", and "ref." represent "without compensation", "with compensation", and "reference trajectory", respectively. (i) 3D position and 3-axis positions. (ii) Velocity norm and 3-axis position errors. **c** Overlaid photos of the tracking of tilted figure 8 trajectory with 5-s period when actuator compensation is enabled. (i) Front view. (ii) Side view. **d** Flight data for tracking a tilted figure 8 trajectory. Legend definition is the same as before. Without compensation, the UAV failed to track the reference trajectory and crashed halfway, indicated by the dropping of blue line in z axis. (i) 3D position and 3-axis positions. (ii) Velocity norm and 3-axis position errors.

Fig. 5: Tracking of tilted figure 8 trajectory with two periods, each for 4 s. **a** Shape of the figure 8 trajectory. (i) Overlaid photo during tracking of a section of the trajectory (flight from P1 to P2). (ii) Planned trajectory in the simulator (the section from P1 to P2 is corresponding to (i)). **b** Violin plot of the position error norm for all 6 experiments (3 times for OMMPC and PID with feedforward (PID+FF), respectively). The shape of the violin plot represents the density of the data distribution, with the red point indicating the mean value. The black error bar denotes the range between the 25% (lower quartile) and the 75% (upper quartile), capturing the central 50% of the data. **c** Tracking performance comparison of the high-speed figure 8 trajectory with two 4-s periods. The trajectory tracking experiments are performed for 6 times in total, with 3 times for PID or OMMPC, respectively. The central line represents the mean value and the error band (i.e., shadow area) indicates the maximum and minimum values. (i) Position in three axes. (ii) Position error in three axes. (iii) Three components of the attitude unit vector on S^2 manifold. (iv) Attitude error in the tangent space $\mathbb{R}^2$ and the attitude angle error (defined by a scalar intersection angle between two vectors on S^2). (v) The velocity norm, the control efforts of angular velocity norm, and thrust.

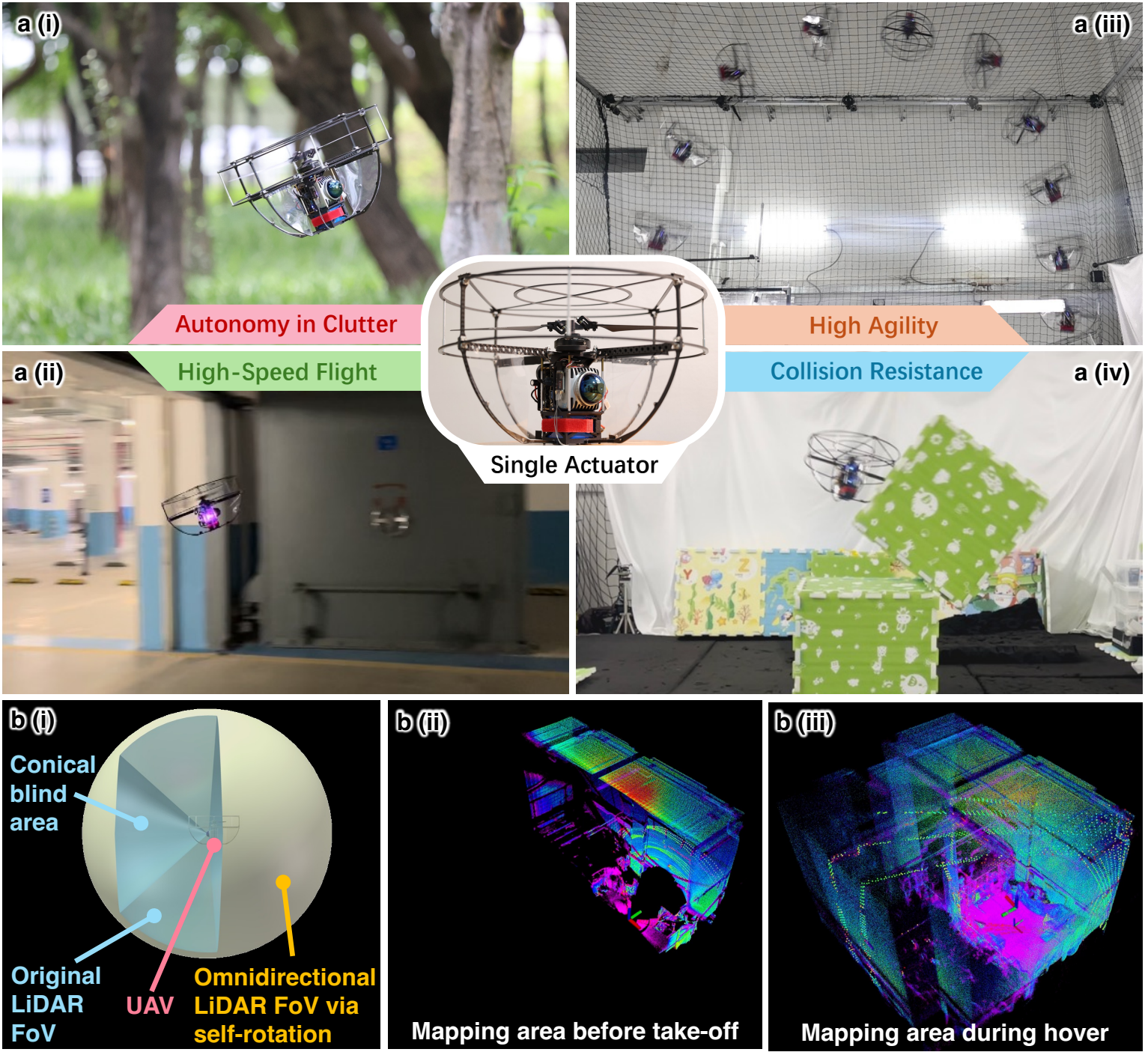
Fig. 6: Tracking of flip trajectory. **a** Shape of the flip trajectory. (i) Overlaid photo of the whole tracking process. (ii) Planned trajectory in the simulator. **b** Violin plot of position error norm for all 6 experiments (3 times for OMMPC and PID with feedforward (PID+FF), respectively). The shape of the violin plot represents the density of the data distribution, with the red point indicating the mean value. The black error bar denotes the range between the 25% (lower quartile) and the 75% (upper quartile), capturing the central 50% of the data. **c** Tracking performance comparison of the flip trajectory. Content explanations are same as the previous figure.

Fig. 7: Collision resistance of PULSAR II. **a** Snapshots of collision with a dropping ball during hover flight. **b** Curves of 3-axis position errors and attitude angle error during ball collision with different control configurations. Each type of experiment is conducted for 3 times and the lines in the figures are the mean values. **c** Violin plot of 3-axis position errors and attitude angle error. Each type of experiment is conducted for 3 times and all 3-times data are merged into one violin plot. The shape of the violin plot represents the density of the data distribution, with the red point indicating the mean value. The black error bar denotes the range between the 25% (lower quartile) and the 75% (upper quartile), capturing the central 50% of the data. **d** Snapshots of collision with an unfixed box during forward flight. **e** Position, velocity norm, and attitude responses during box collision with different control configuration. Each type of experiment is repeated 3 times and the data lines represent the mean values of the 3-times experiments.

Fig. 8: Autonomous navigation in a dense woods with loop and star trajectories. **a** Top view of the point cloud map built in real time when performing autonomous navigation with the loop trajectory. The blue stars represent the guide points and the yellow arrows indicate the zoom-in directions whose views are shown in the right-side figures. **b** (i-iii) Zoom-in trajectories with corresponding snapshots in real world with red arrow pointing to the UAV. **c** Data of the autonomous navigation with loop trajectory. (i) Position and errors. (ii) The three components of the attitude vector on S^2 manifold. (iii) velocity norm, angular velocity norm, and thrust. **d** Top view of the point cloud map built in real time when performing autonomous navigation with the star trajectory. **e** (i-iii) Zoom-in trajectories with corresponding snapshots in real world with red arrow pointing to the UAV. **f** Data of the autonomous navigation with star trajectory.

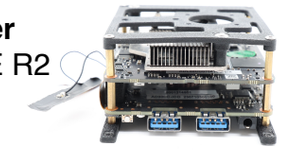
Fig. 9: Autonomous navigation in an underground parking lot. **a** Top view of the point cloud map built in real time. The blue stars represent the guide points and the yellow arrows indicate the zoom-in directions whose views are shown in right-side figures. **b** (i-iii) Zoom-in trajectories with snapshots in real world. **c** Data of the autonomous navigation. (i) Position and errors. (ii) The three components of the attitude vector on S^2 manifold. (iii) velocity norm, angular velocity norm, and thrust.

Fig. 10: Illustration of the coordinate definition, the attitude state on S^2 manifold, and the system structure of disturbance observation. **a** Three coordinate frames are defined for PULSAR II which are inertial frame $\mathcal{I}$, body-fixed frame $\mathcal{B}$, and fictional frame $\mathcal{F}$. Both frames $\mathcal{B}$ and $\mathcal{F}$ are fixed on the center of mass (CoM) of UAV and their z axes are same. Frame $\mathcal{F}$ has zero angular velocity in z axis while frame $\mathcal{B}$ has a z-axis angular velocity due to self-rotation. The rotation matrix $\mathbf{R}^{\mathcal{FB}}$ describes the transformation of coordinates from $\mathcal{B}$ frame to $\mathcal{F}$ frame and the rotation matrix $\mathbf{R}^{\mathcal{IF}}$ describes the transformation of coordinates from $\mathcal{F}$ frame to $\mathcal{I}$ frame. Three CoM are depicted, which are CoM of rotor part including motor's rotor and propeller, CoM of the body part (i.e., the whole UAV excluding the rotor part), and CoM of the whole UAV. **b** Illustration of the attitude state on S^2 manifold and its $\mathbb{R}^2$ tangent space which are the bases for minimum parameterization of the OMMPC. **c** System structure of angular velocity control with disturbance observer (DOB) and explanations of signals. **d** Structure of the generalized system for the optimization of DOB parameters via μ synthesis.

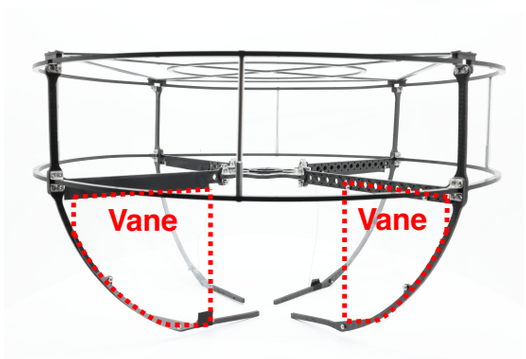

a (i)
a (iii)
Autonomy in Clutter
High Agility
Single Actuator
a (ii)
High-Speed Flight
Collision Resistance
a (iv)
b (i)
Conical blind area
Original LiDAR FoV
UAV
Omnidirectional LiDAR FoV via self-rotation
b (ii)
Mapping area before take-off
b (iii)
Mapping area during hover

a

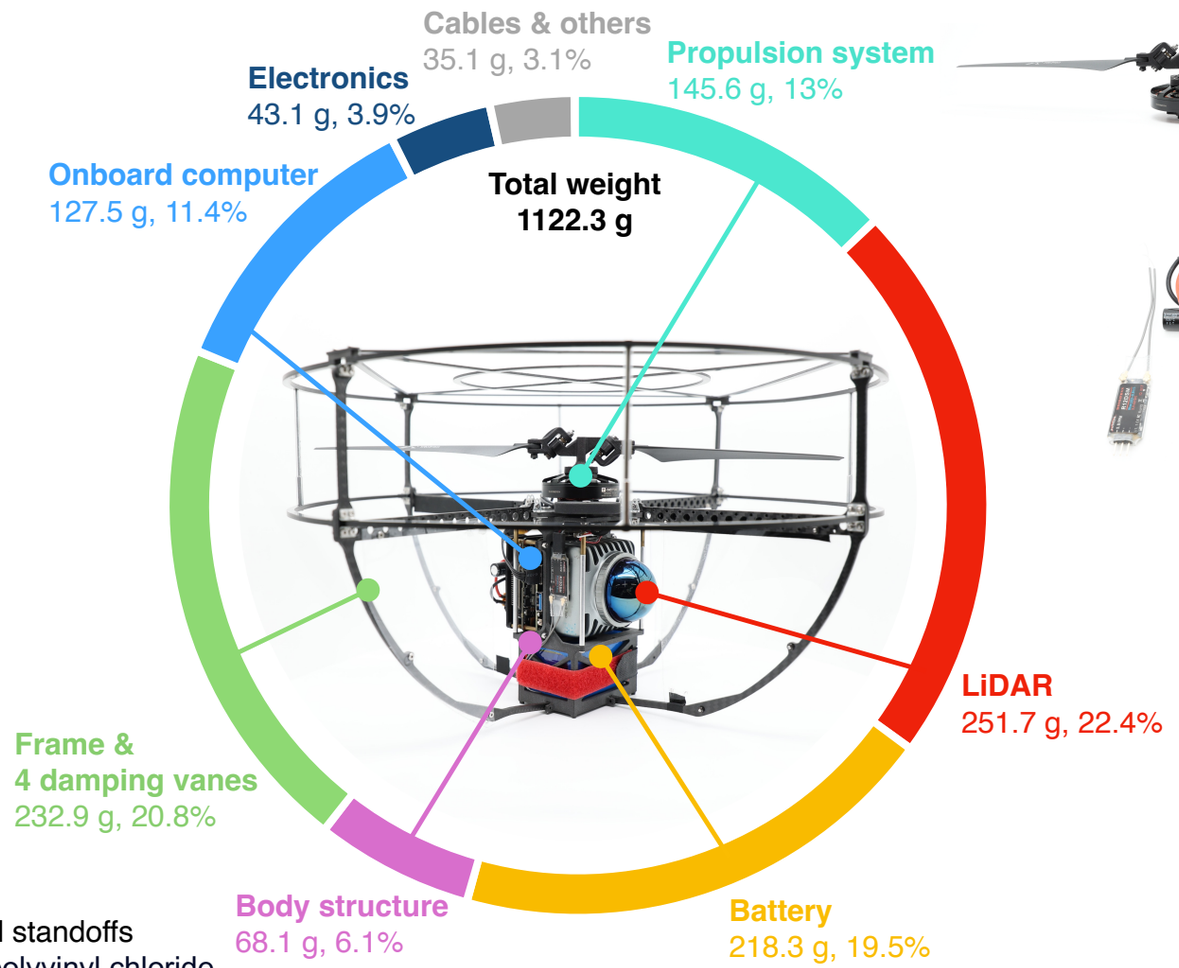
Onboard computer
Type: iKOOLCORE R2
CPU: Intel N200



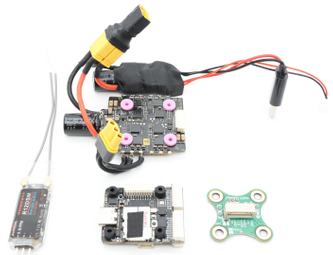
Body structure
Material: 3D-printed nylon
with metal standoffs



Frame & 4 damping vanes
Frame material: Carbon fiber with metal standoffs
Damping vanes material: Transparent polyvinyl chloride



Propulsion system
Motor: T-MOTOR MN5006 450KV
Propeller: T-MOTOR MF1302 blades
with swashplateless mechanism

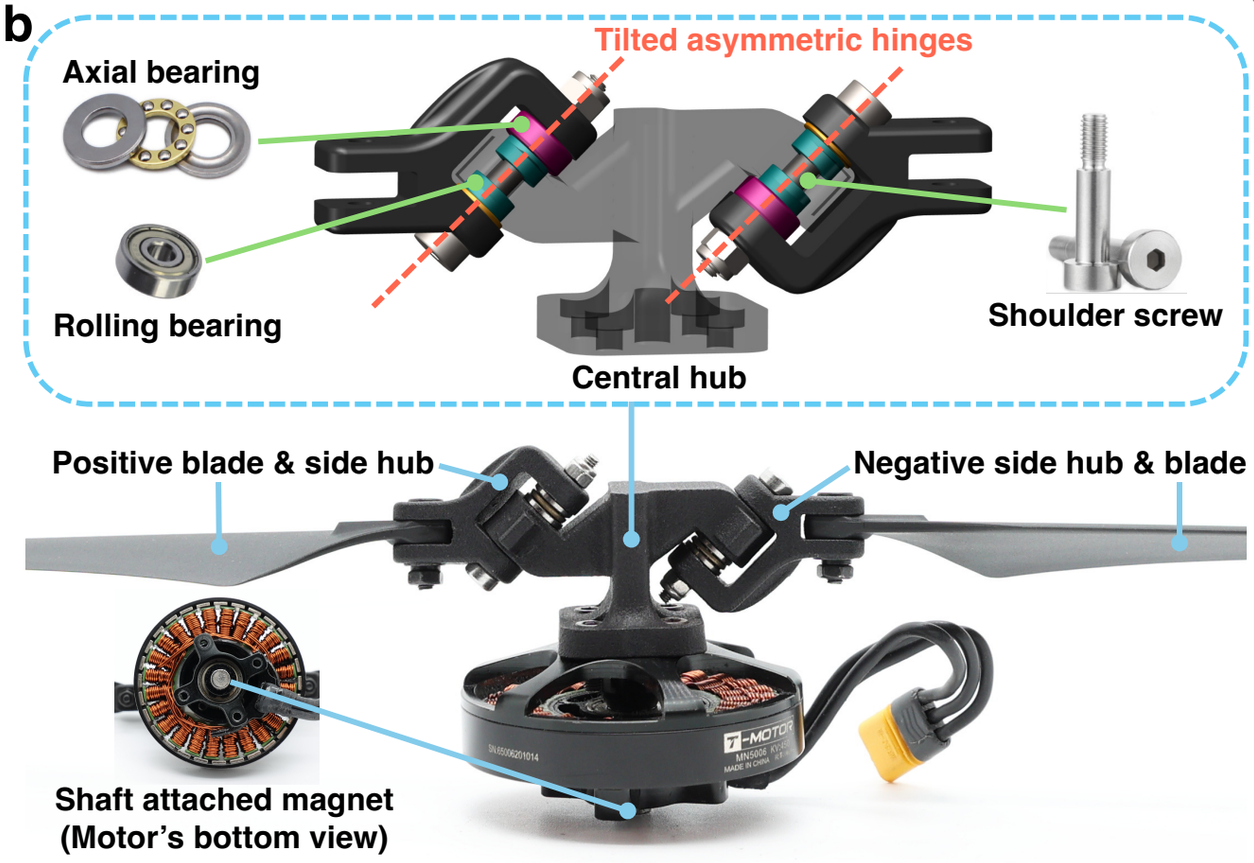


Electronics
DC-DC convertor: 25.2 V to 12 V
ESC: T-MOTOR MINI F45A
RC receiver: Radio R12DSM
Flight controller: Nxt-FC V1.2
Magnetic encoder: ams AS5147U

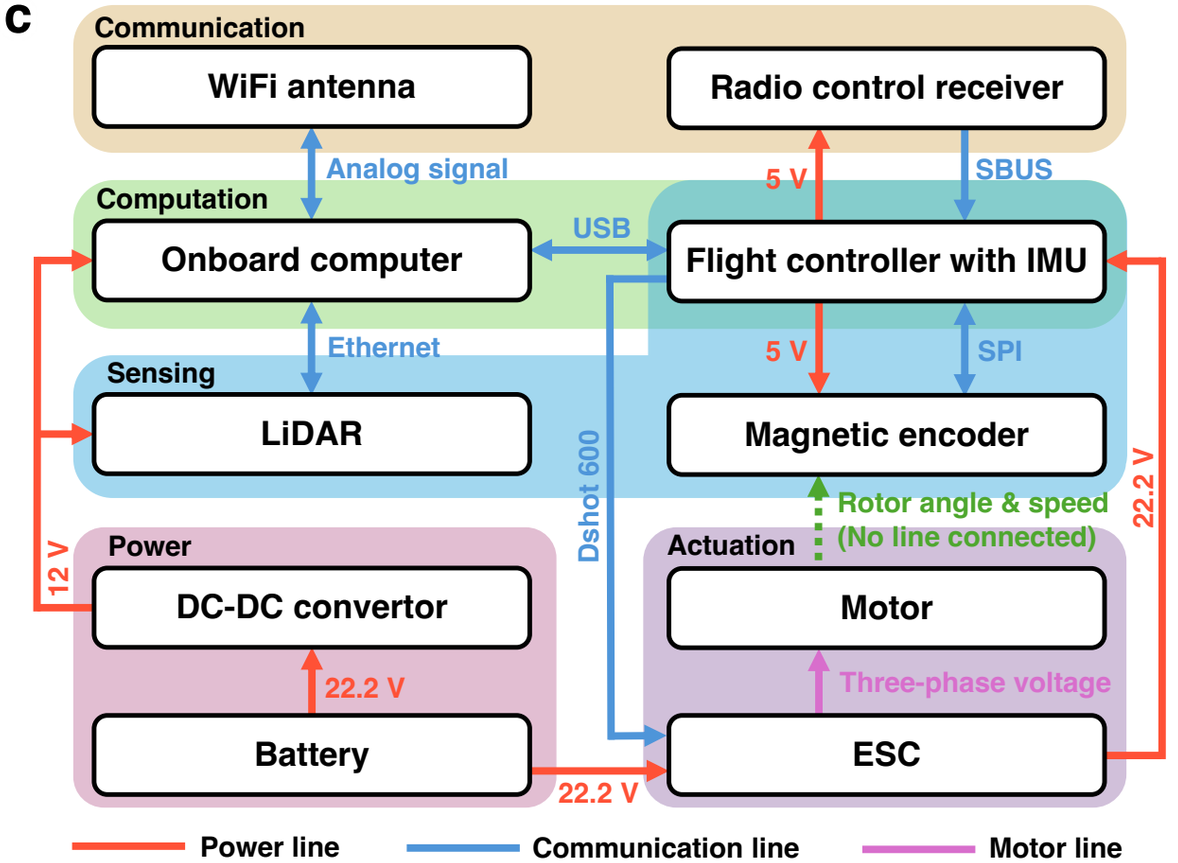
LiDAR
Type: Livox Mid360
Range: 40 m
FoV: Horizontal 360° & Vertical 59°

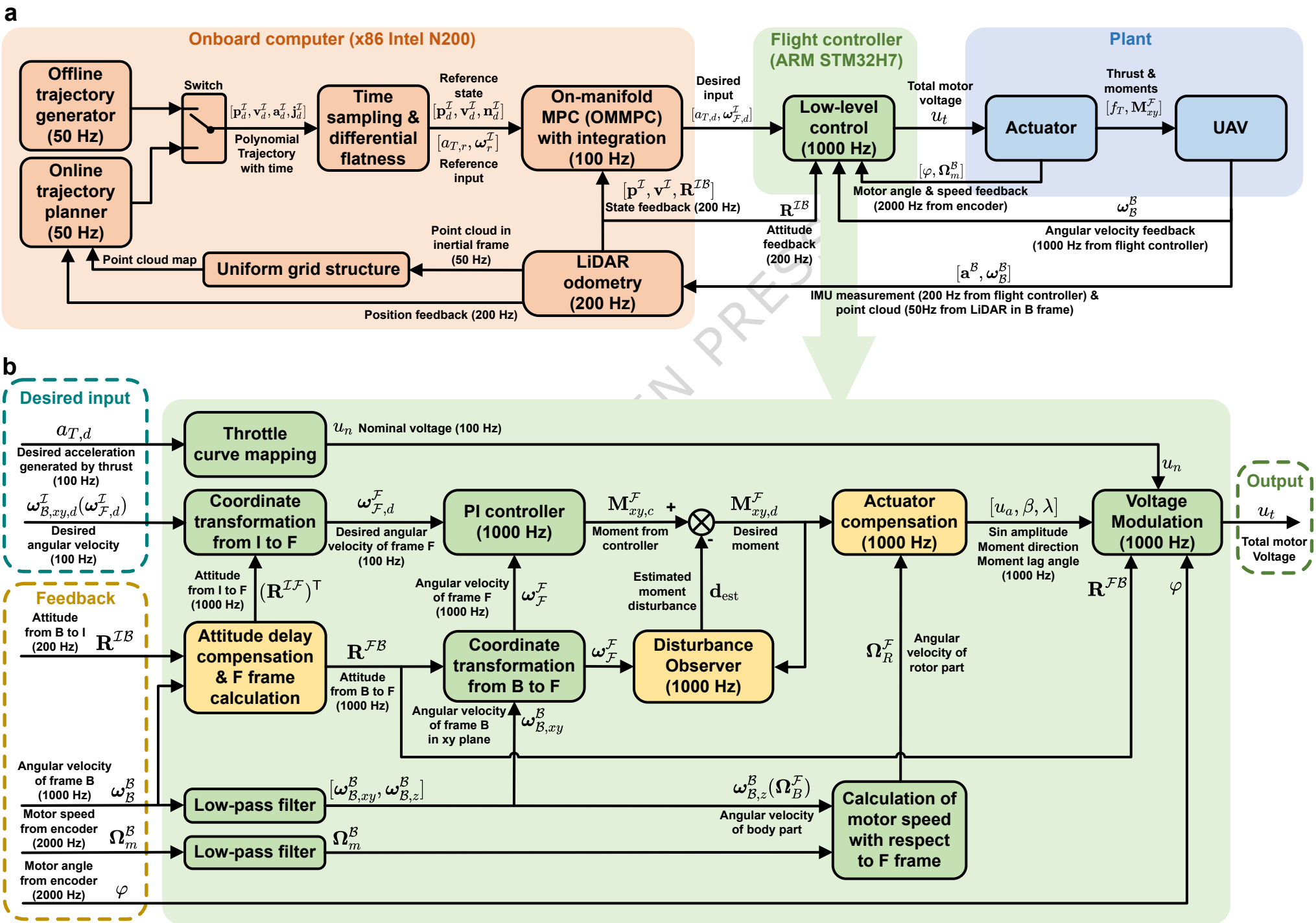
Battery
Specification:
6 cells, 53.7 Wh, 2420 mAh

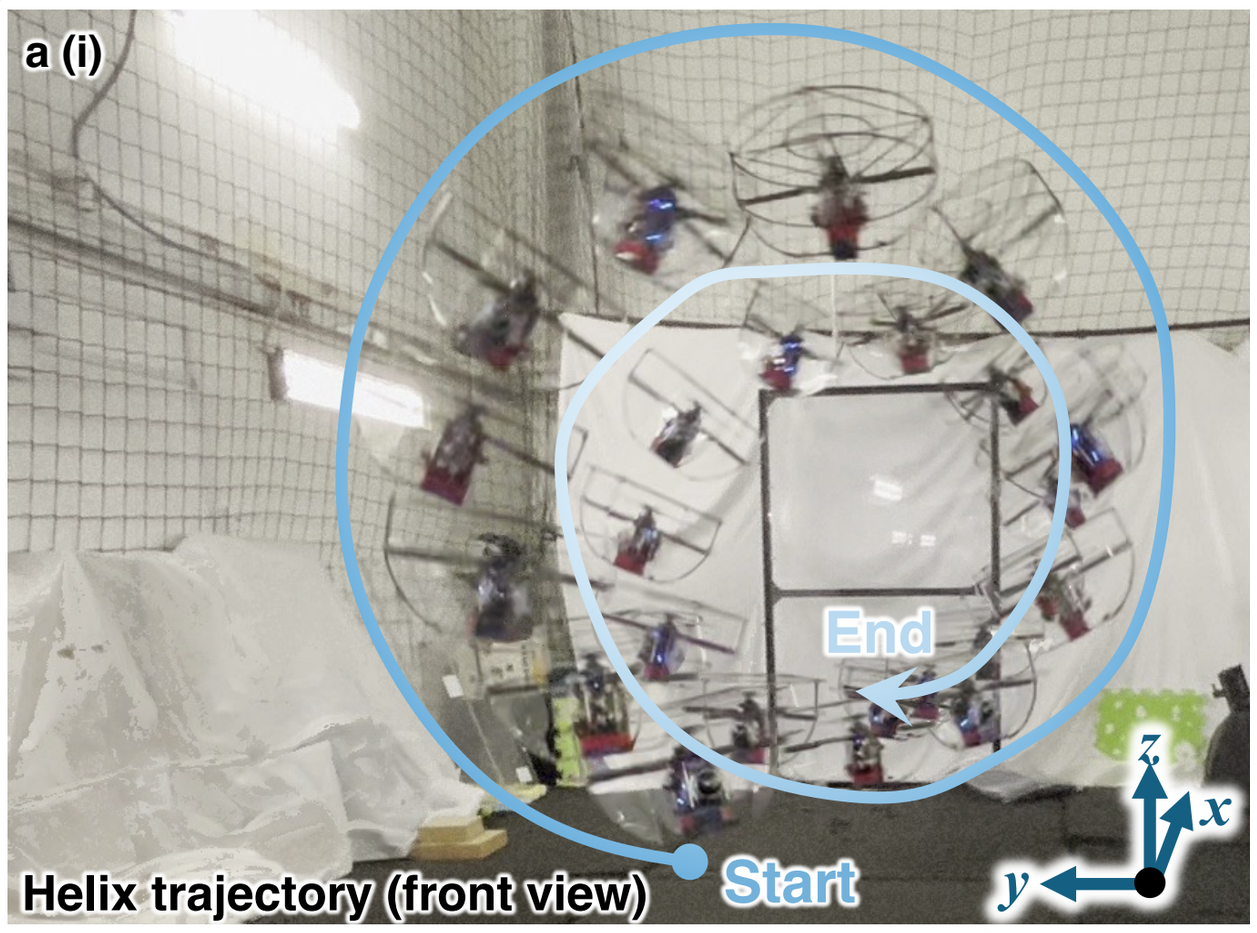
b

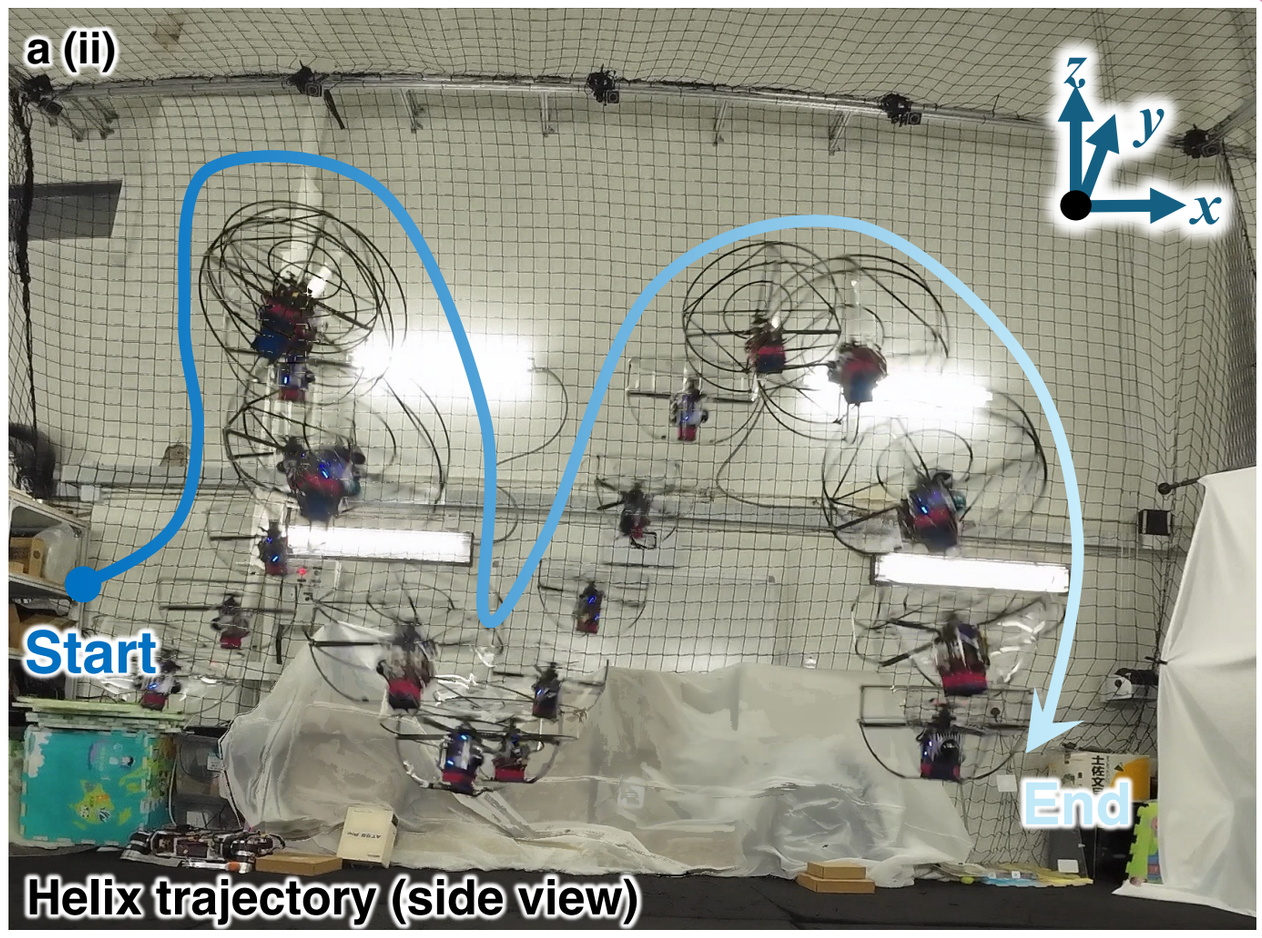


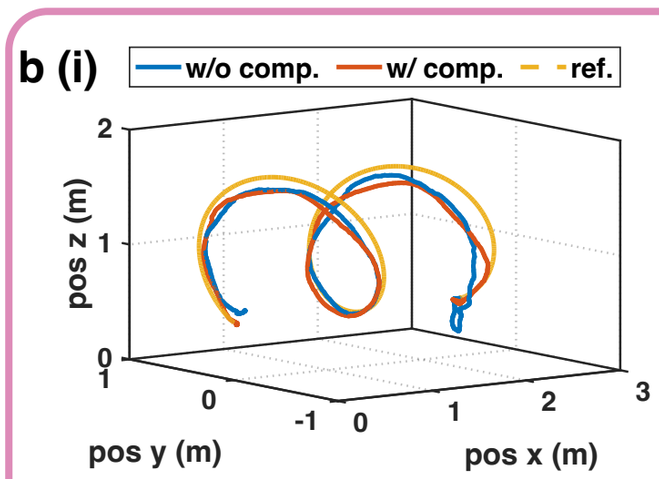
c

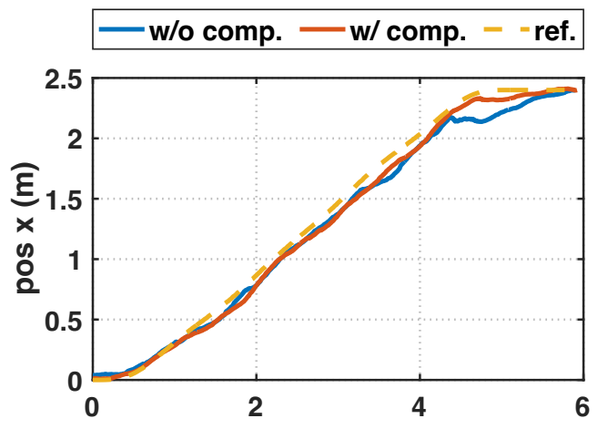


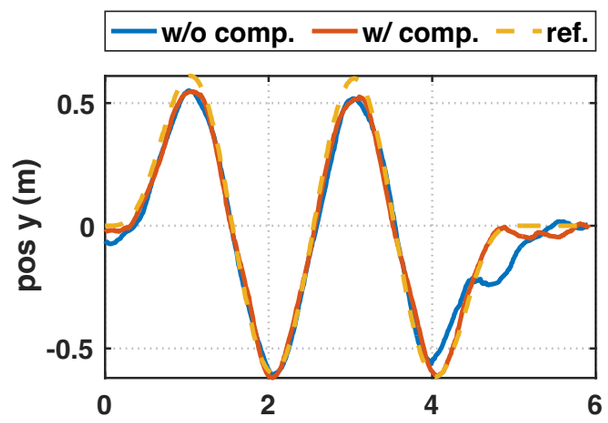

a
Onboard computer (x86 Intel N200)
Flight controller (ARM STM32H7)
Plant
Offline trajectory generator (50 Hz)
Online trajectory planner (50 Hz)
Switch
Polynomial Trajectory with time
Time sampling & differential flatness
Reference state
Reference input
On-manifold MPC (OMMPC) with integration (100 Hz)
Desired input
Low-level control (1000 Hz)
Total motor voltage
Actuator
Thrust & moments
UAV
State feedback (200 Hz)
Attitude feedback (200 Hz)
Motor angle & speed feedback (2000 Hz from encoder)
Angular velocity feedback (1000 Hz from flight controller)
Point cloud map
Uniform grid structure
Point cloud in inertial frame (50 Hz)
LiDAR odometry (200 Hz)
Position feedback (200 Hz)
IMU measurement (200 Hz from flight controller) & point cloud (50Hz from LiDAR in B frame)
b
Desired input
Desired acceleration generated by thrust (100 Hz)
Desired angular velocity (100 Hz)
Feedback
Attitude from B to I (200 Hz)
Angular velocity of frame B (1000 Hz)
Motor speed from encoder (2000 Hz)
Motor angle from encoder (2000 Hz)
Throttle curve mapping
Nominal voltage (100 Hz)
Coordinate transformation from I to F
Desired angular velocity of frame F (100 Hz)
PI controller (1000 Hz)
Moment from controller
Desired moment
Actuator compensation (1000 Hz)
Sin amplitude
Moment direction
Moment lag angle
(1000 Hz)
Voltage Modulation (1000 Hz)
Output
Total motor Voltage
Attitude from I to F (1000 Hz)
Angular velocity of frame F (1000 Hz)
Estimated moment disturbance
Attitude delay compensation & F frame calculation
Attitude from B to F (1000 Hz)
Coordinate transformation from B to F
Angular velocity of frame B in xy plane
Disturbance Observer (1000 Hz)
Angular velocity of rotor part
Low-pass filter
Low-pass filter
Angular velocity of body part
Calculation of motor speed with respect to F frame

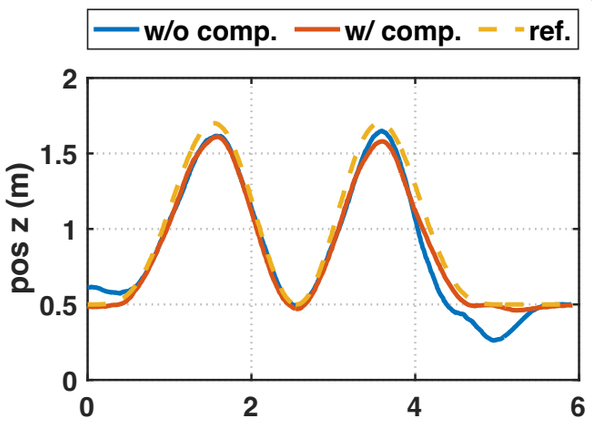

a (i)
End
Start
Helix trajectory (front view)
z
x
y

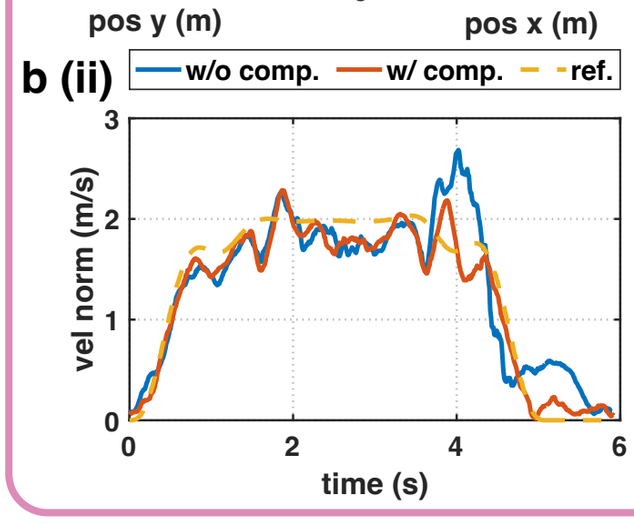

a (ii)
z
y
x
Start
End
Helix trajectory (side view)

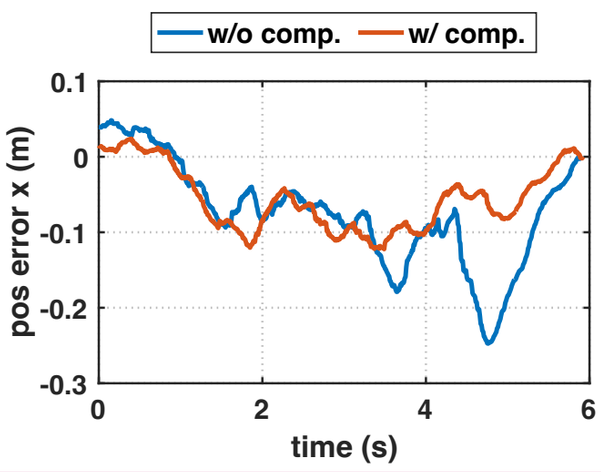

b (i)
w/o comp.
w/ comp.
ref.
pos z (m)
pos y (m)
pos x (m)

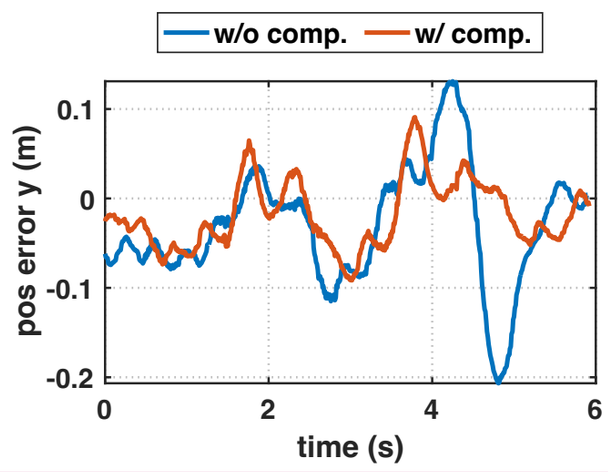

w/o comp.
w/ comp.
ref.
pos x (m)

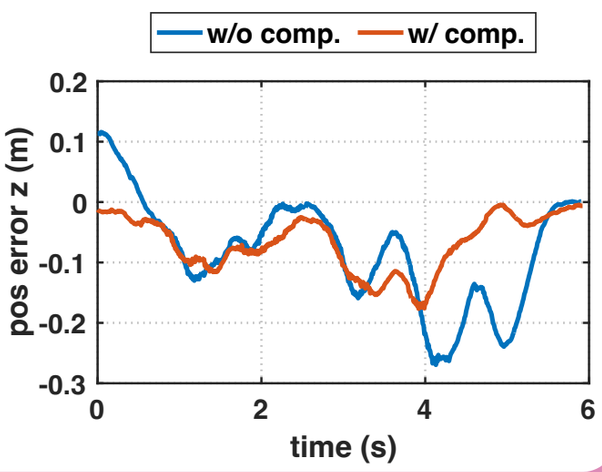

w/o comp.
w/ comp.
ref.
pos y (m)

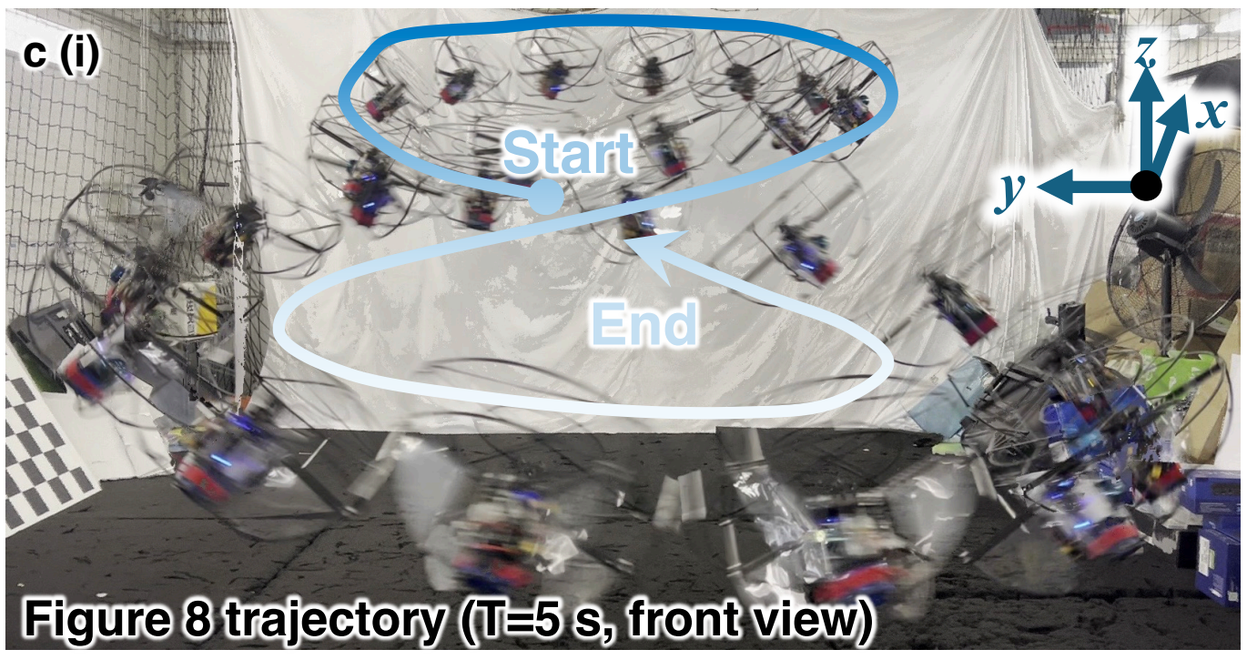

w/o comp.
w/ comp.
ref.
pos z (m)

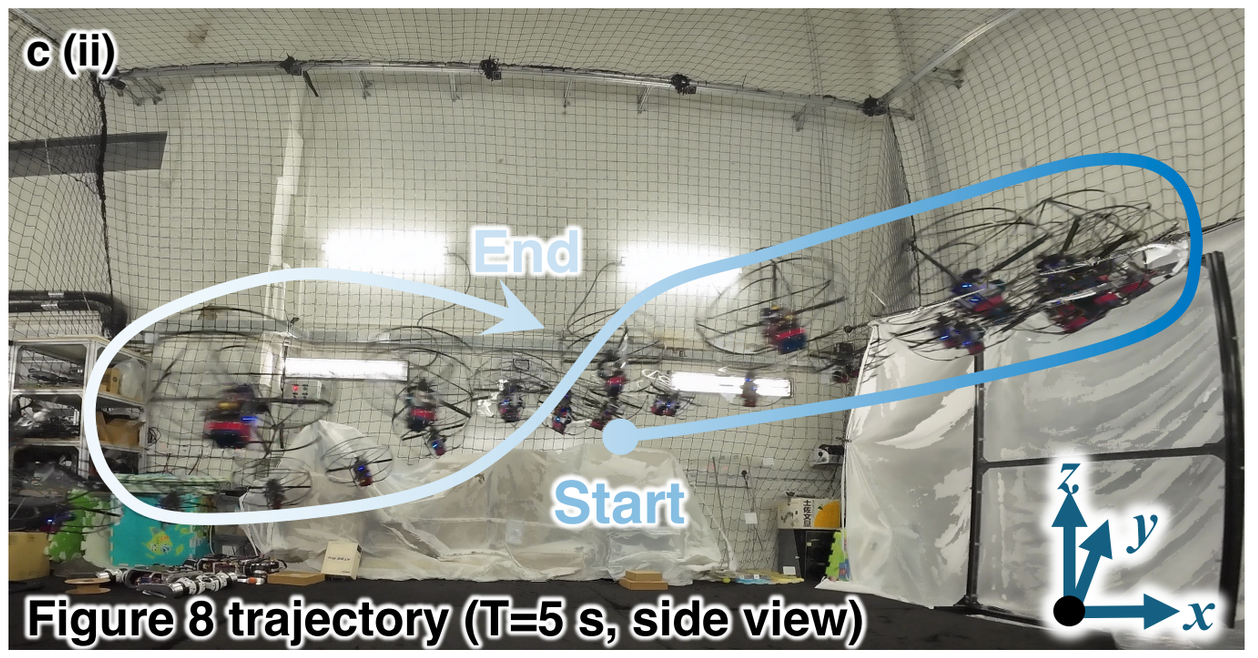

b (ii)
w/o comp.
w/ comp.
ref.
vel norm (m/s)
time (s)

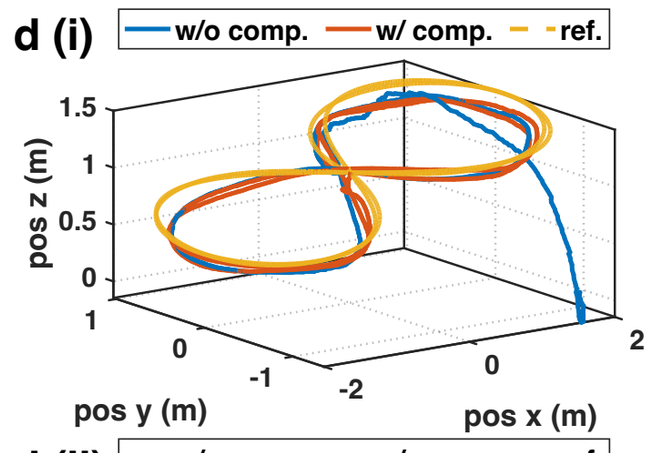

w/o comp.
w/ comp.
pos error x (m)
time (s)

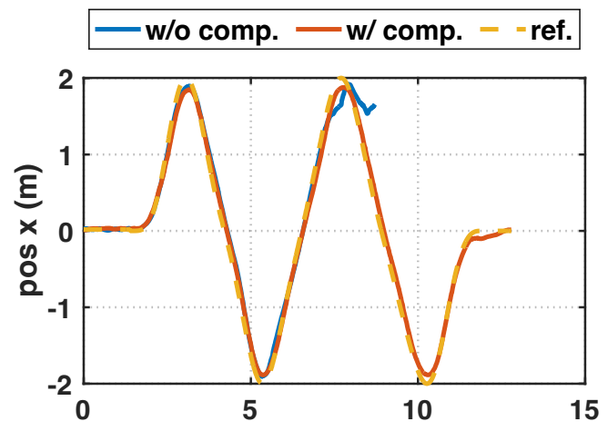

w/o comp.
w/ comp.
pos error y (m)
time (s)

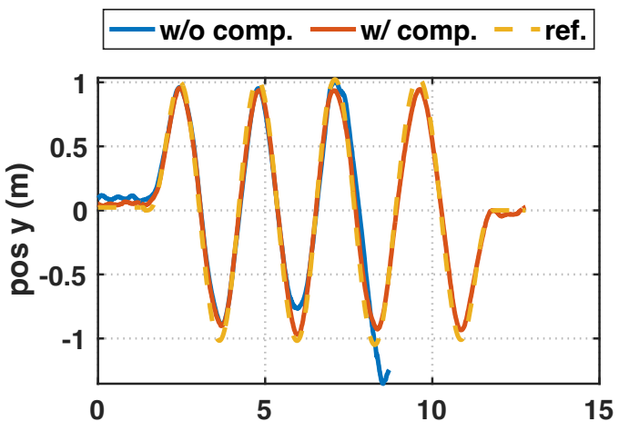

w/o comp.
w/ comp.
pos error z (m)
time (s)

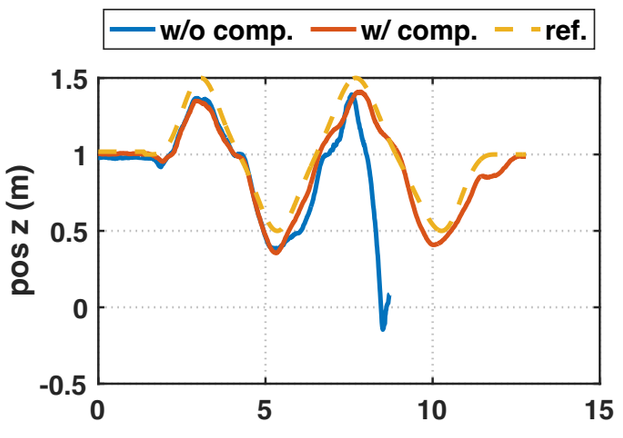

c (i)
Start
End
z
x
y
Figure 8 trajectory (T=5 s, front view)

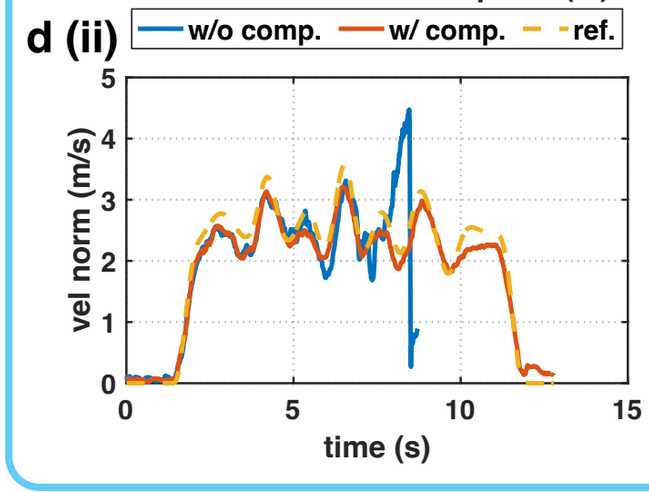

c (ii)
End
Start
z
y
x
Figure 8 trajectory (T=5 s, side view)

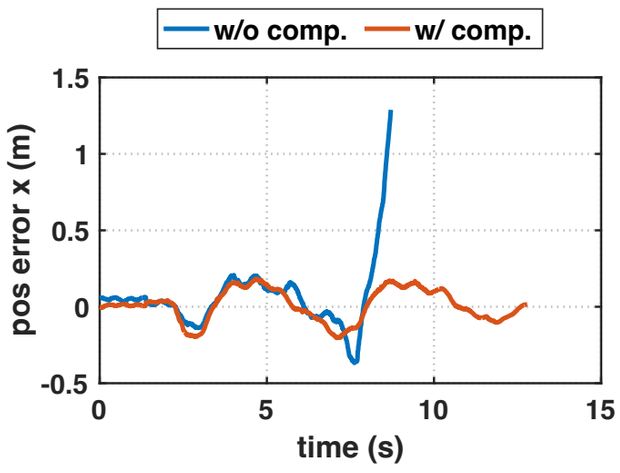

d (i)
w/o comp.
w/ comp.
ref.
pos z (m)
pos y (m)
pos x (m)

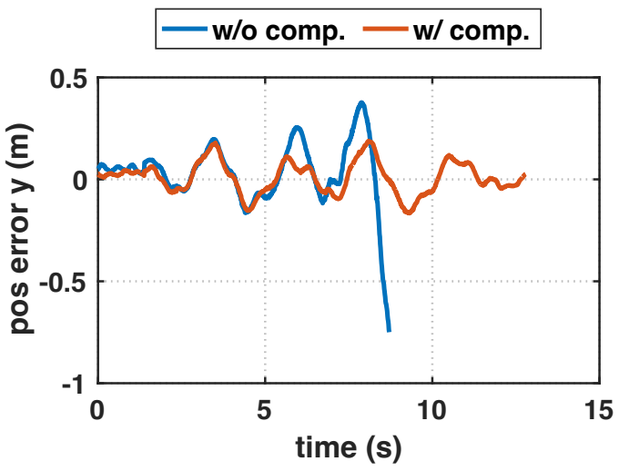

w/o comp.
w/ comp.
ref.
pos x (m)

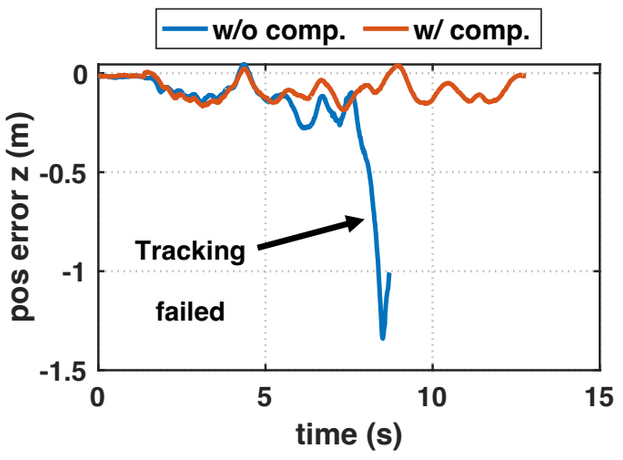

w/o comp.
w/ comp.
ref.
pos y (m)

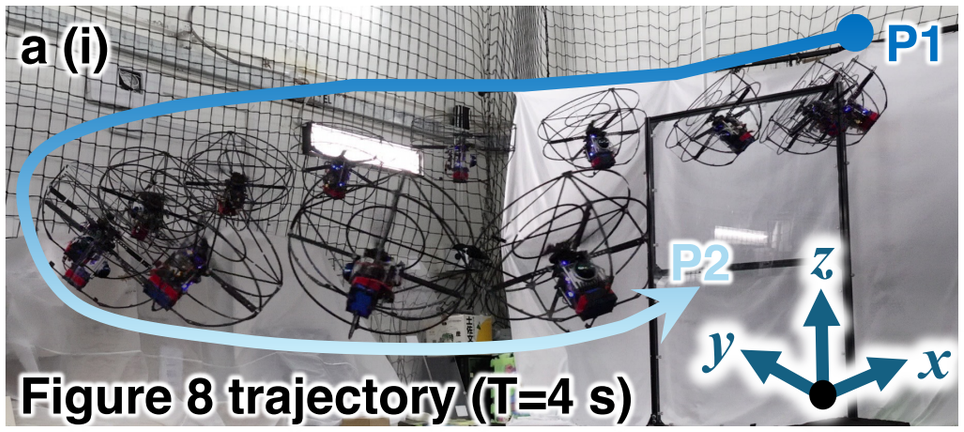

w/o comp.
w/ comp.
ref.
pos z (m)

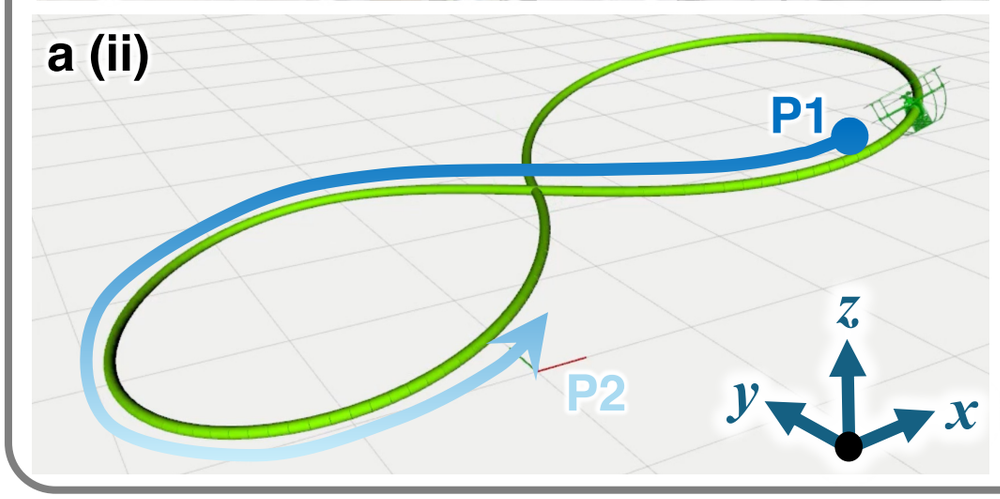

d (ii)
w/o comp.
w/ comp.
ref.
vel norm (m/s)
time (s)

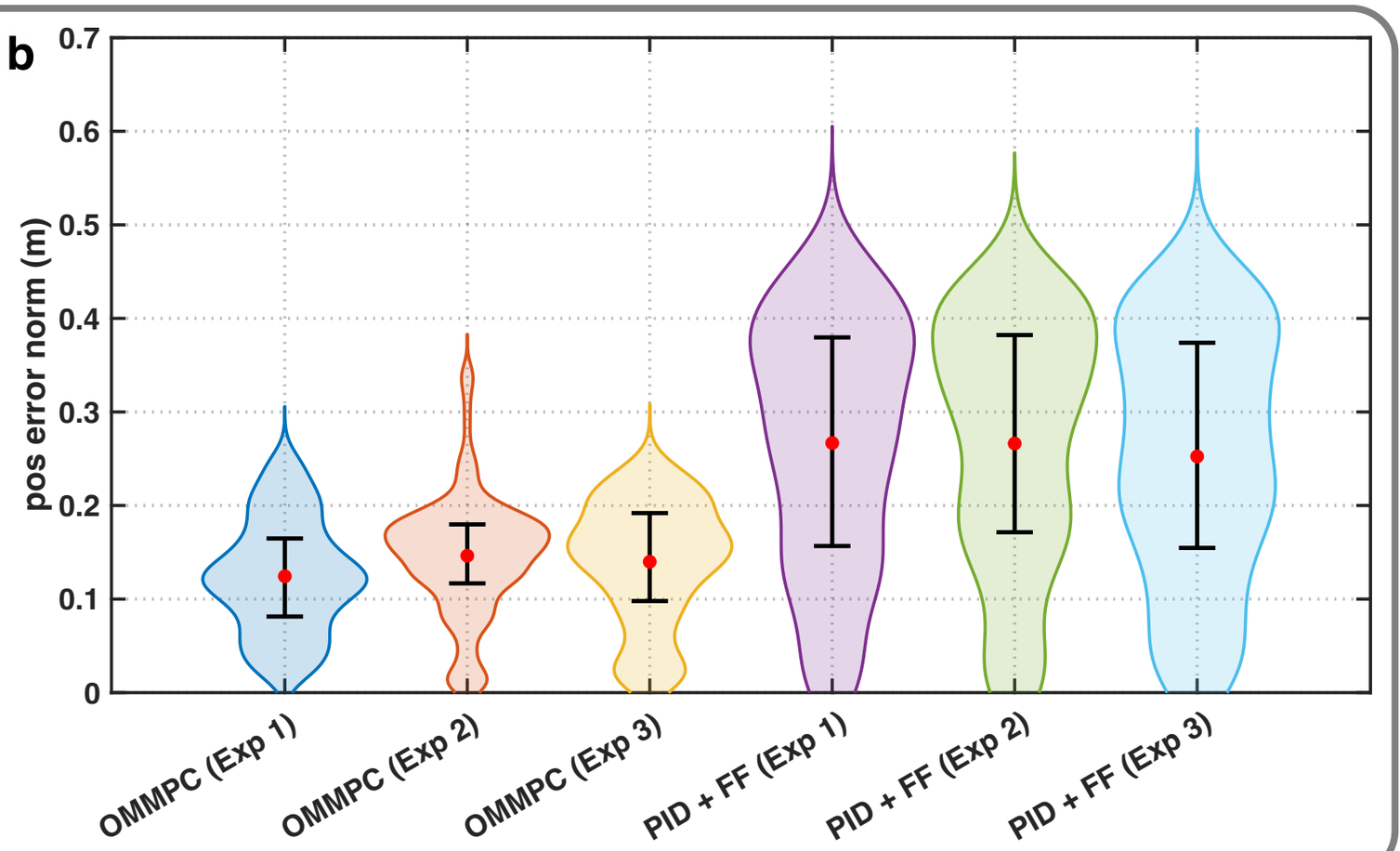

w/o comp.
w/ comp.
pos error x (m)
time (s)

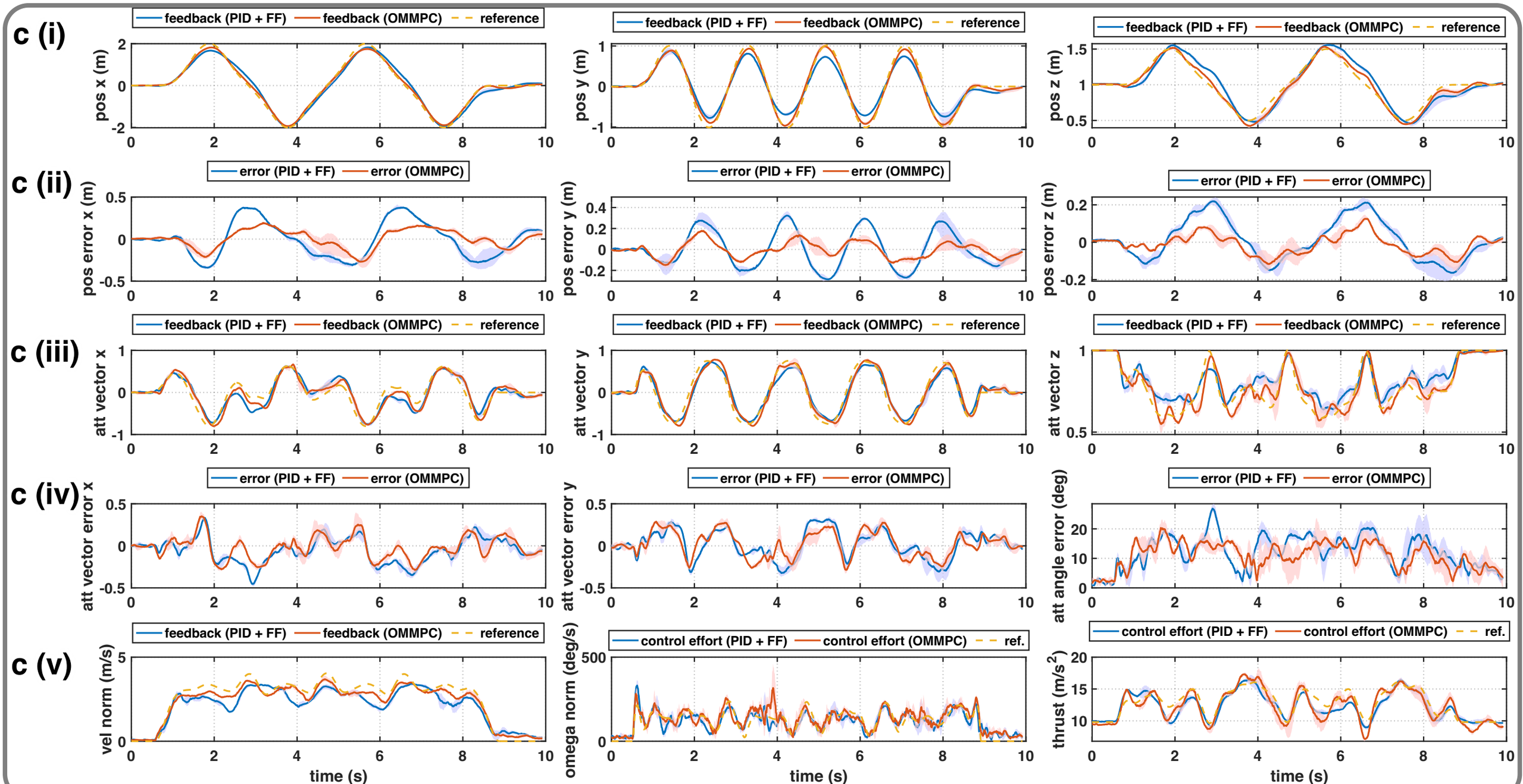

w/o comp.
w/ comp.
pos error y (m)
time (s)

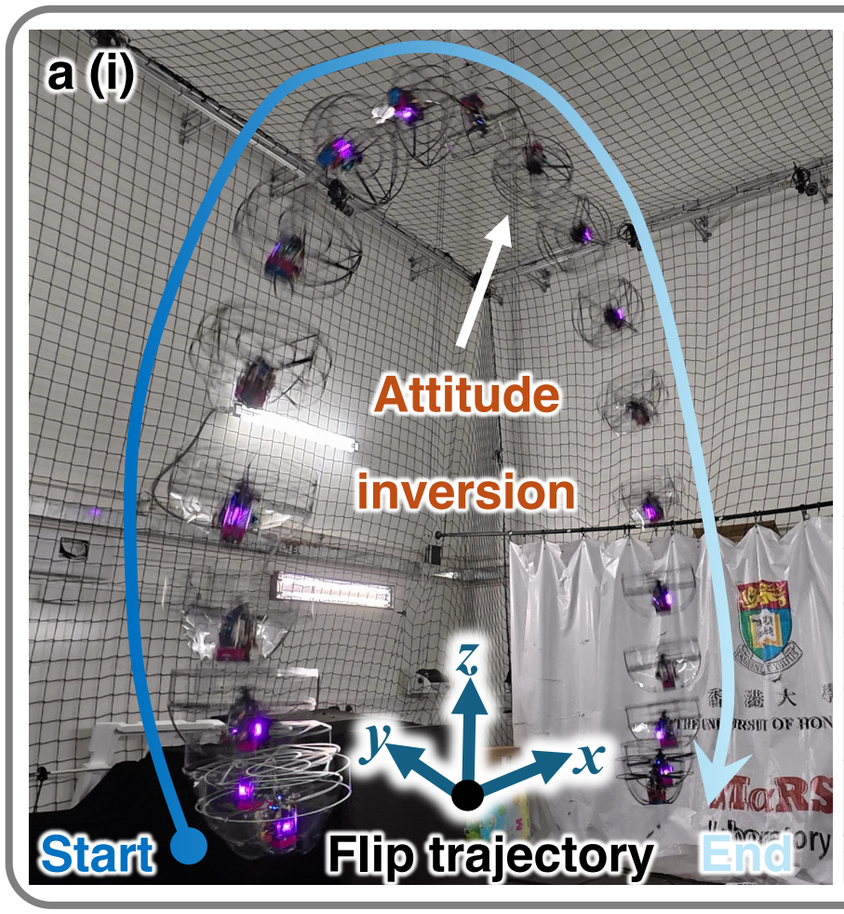

w/o comp.
w/ comp.
pos error z (m)
Tracking
failed
time (s)

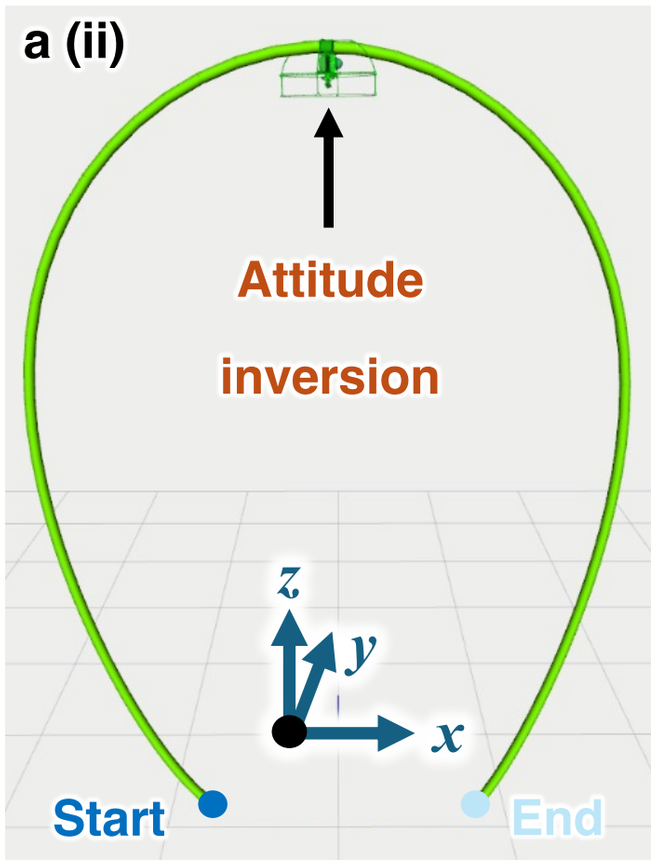

a (i)
P1
P2
z
y
x
Figure 8 trajectory (T=4 s)

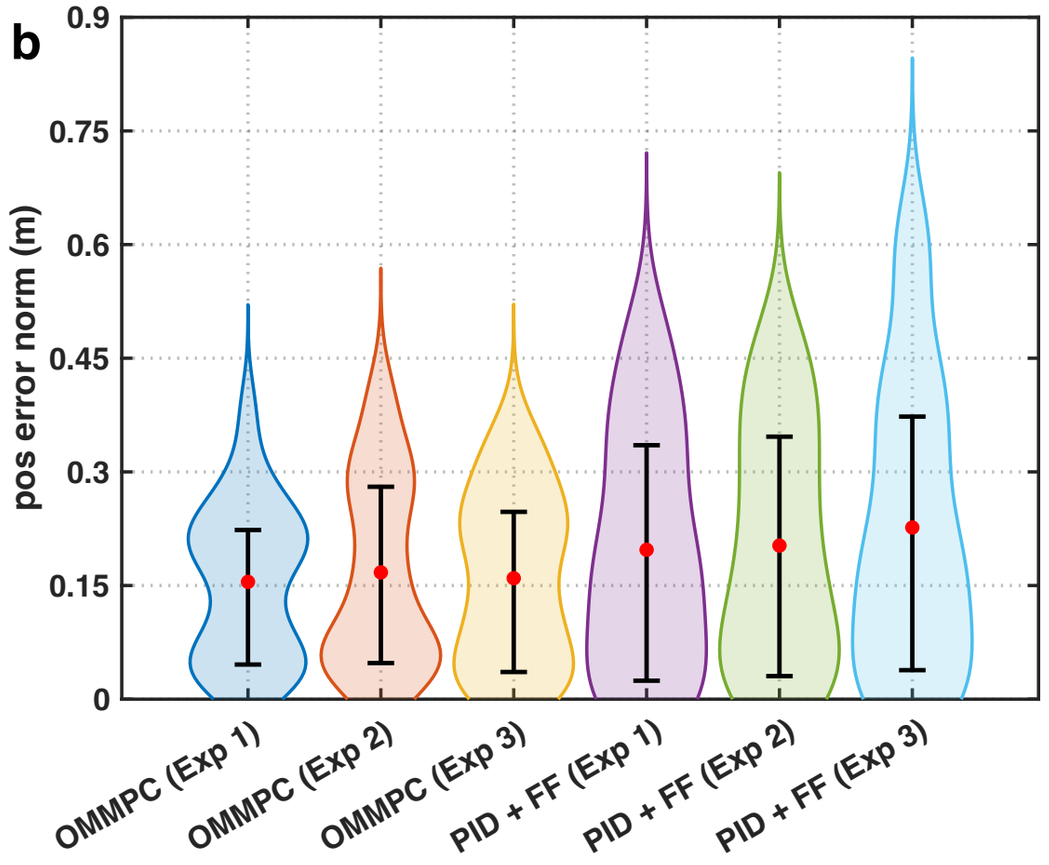

a (ii)
P1
P2
z
y
x

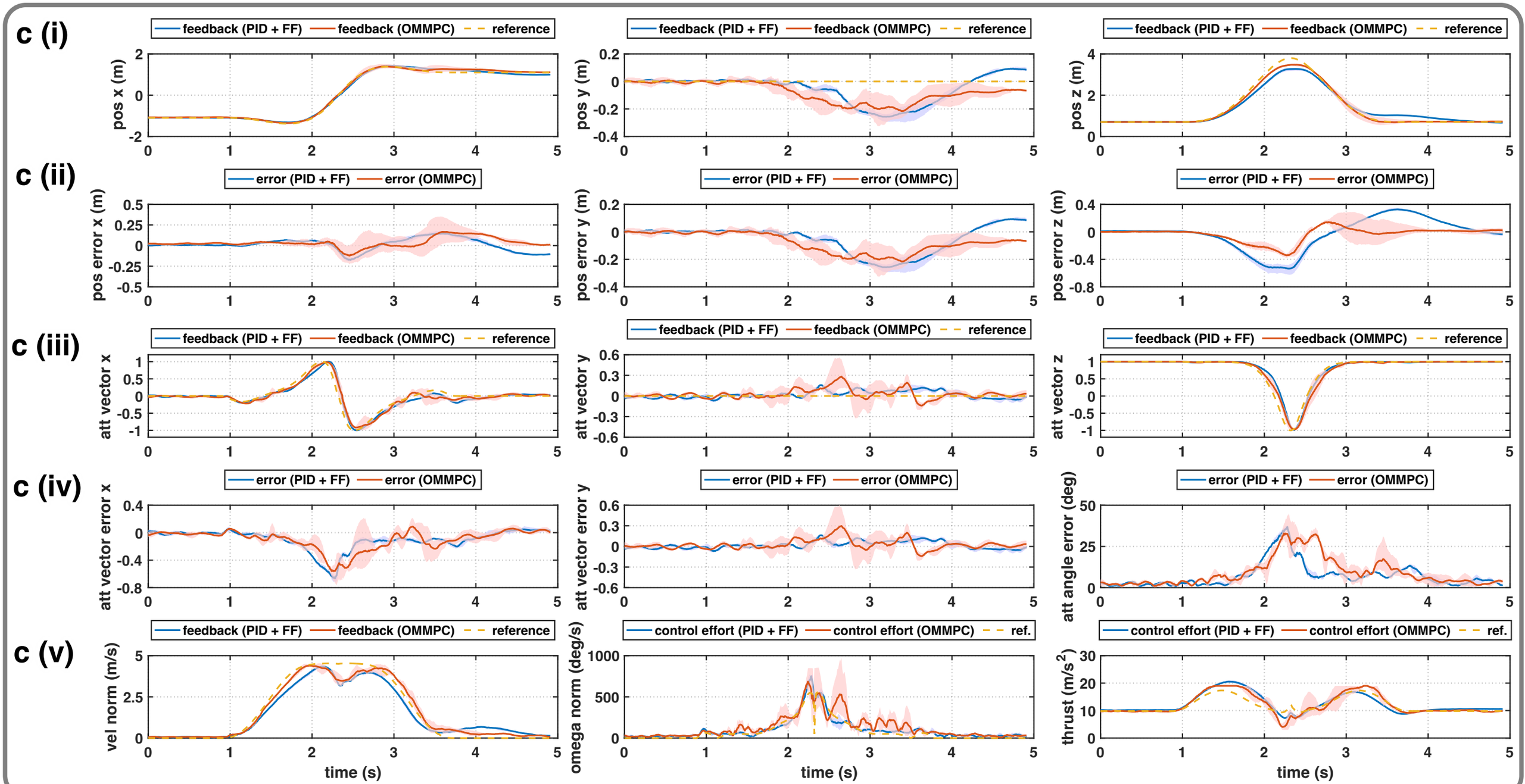

b
pos error norm (m)
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
OMMPC (Exp 1)
OMMPC (Exp 2)
OMMPC (Exp 3)
PID + FF (Exp 1)
PID + FF (Exp 2)
PID + FF (Exp 3)

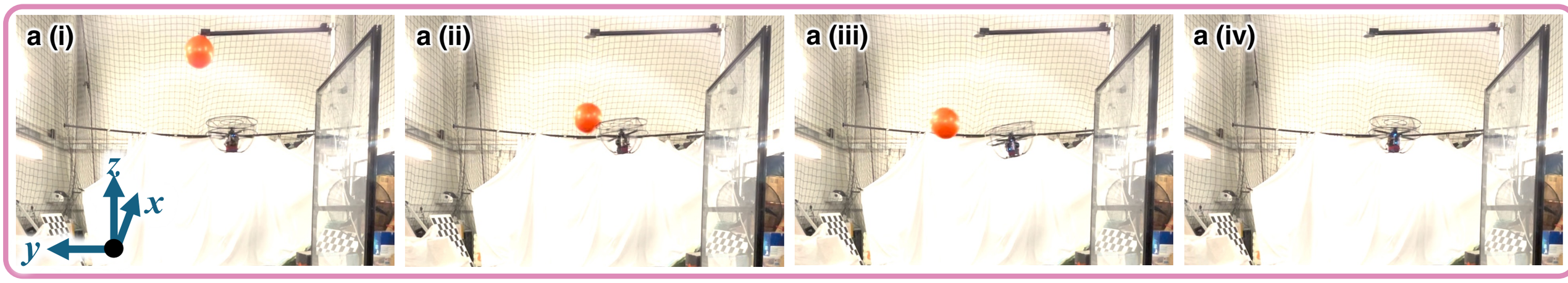

c (i)
feedback (PID + FF)
feedback (OMMPC)
reference
pos x (m)
pos y (m)
pos z (m)
c (ii)
error (PID + FF)
error (OMMPC)
pos error x (m)
pos error y (m)
pos error z (m)
c (iii)
att vector x
att vector y
att vector z
c (iv)
att vector error x
att vector error y
att angle error (deg)
c (v)
vel norm (m/s)
control effort (PID + FF)
control effort (OMMPC)
ref.
omega norm (deg/s)
thrust (m/s²)
time (s)

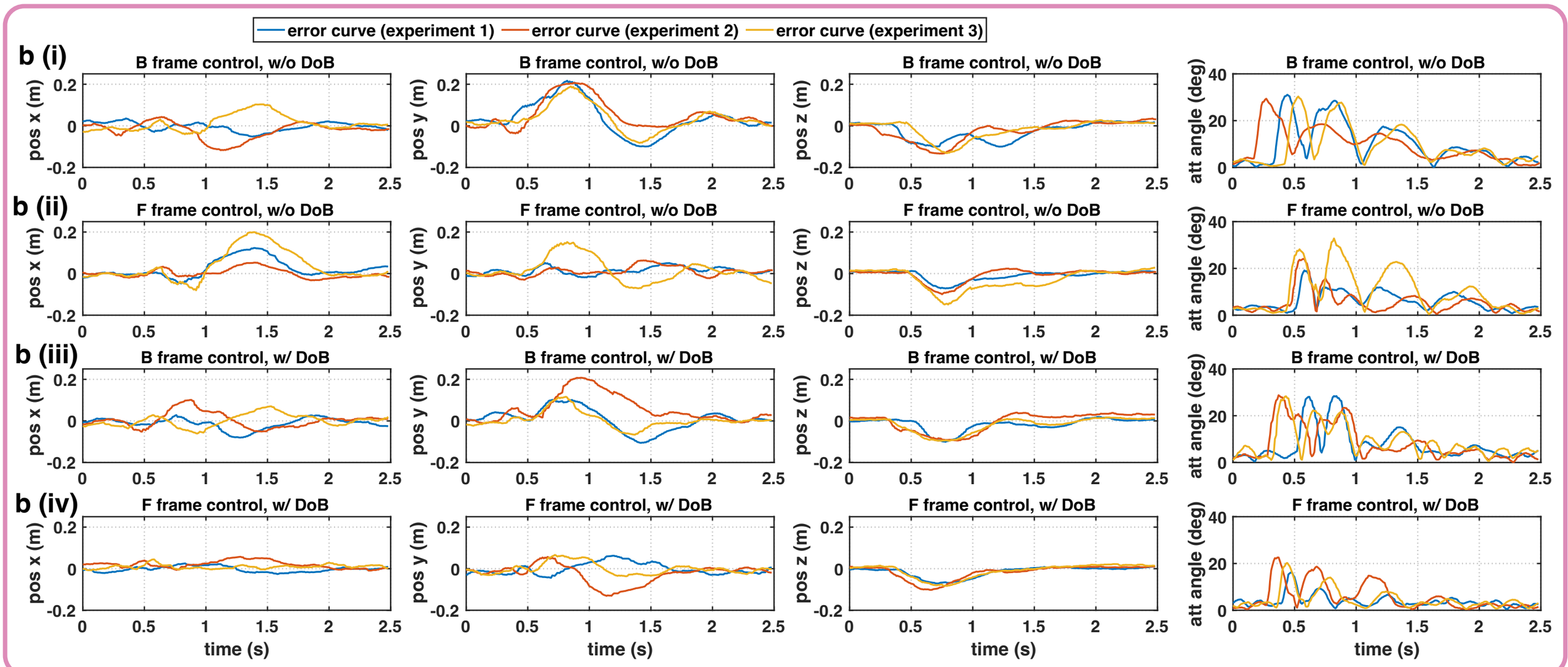

a (i)
Attitude
inversion
Start
Flip trajectory
End

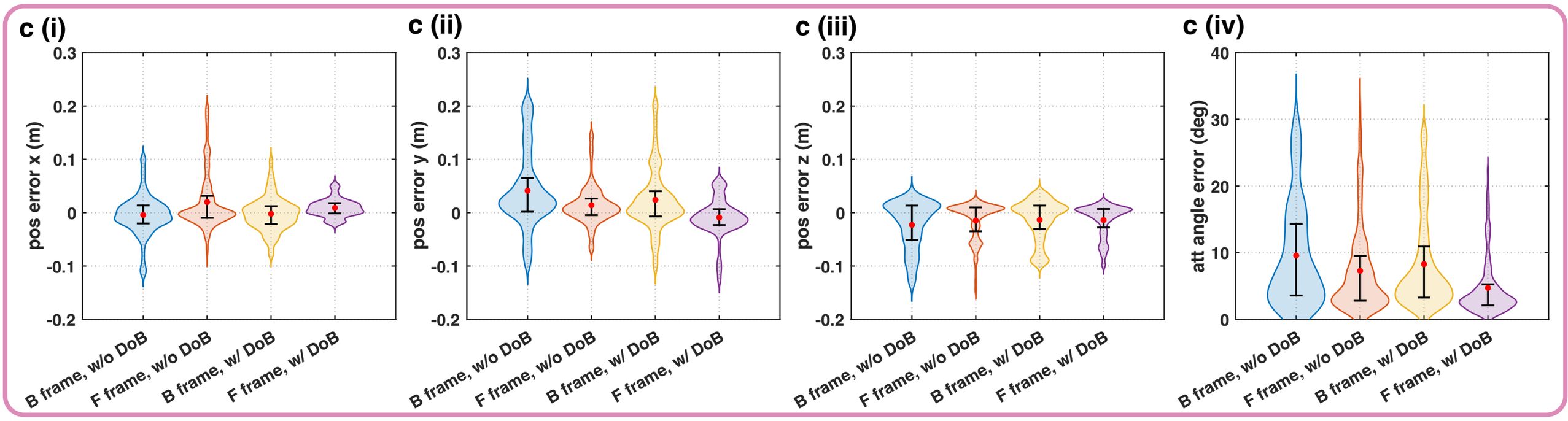

a (ii)
Attitude
inversion
Start
End

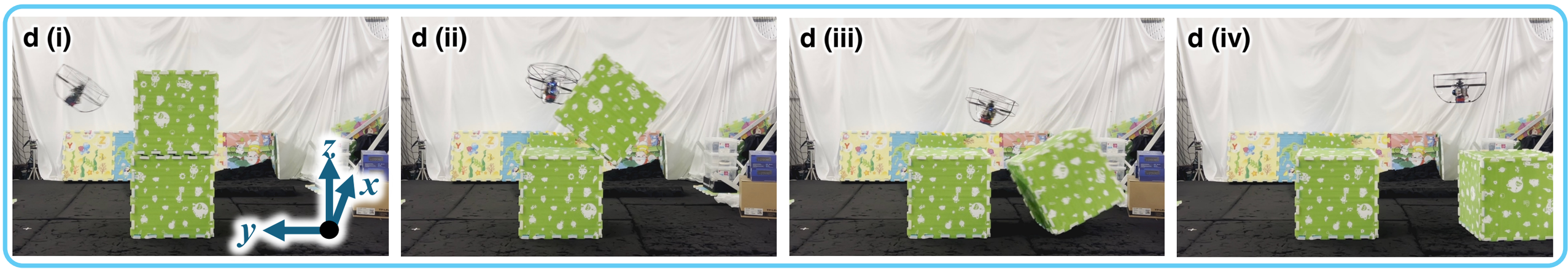

b
pos error norm (m)
OMMPC (Exp 1)
OMMPC (Exp 2)
OMMPC (Exp 3)
PID + FF (Exp 1)
PID + FF (Exp 2)
PID + FF (Exp 3)

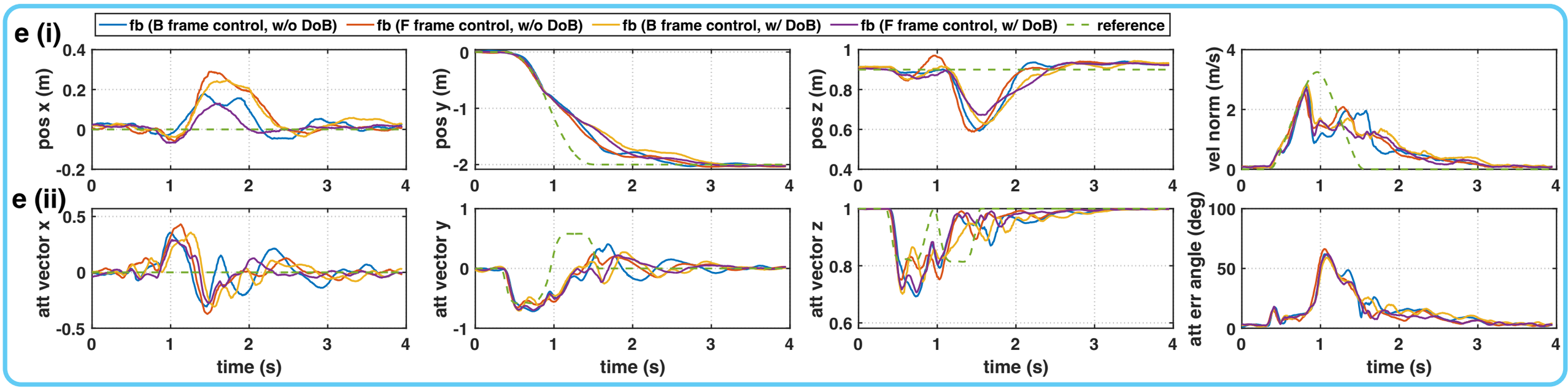

c (i)
c (ii)
c (iii)
c (iv)
c (v)
feedback (PID + FF)
feedback (OMMPC)
reference
error (PID + FF)
error (OMMPC)
control effort (PID + FF)
control effort (OMMPC)
ref.
pos x (m)
pos y (m)
pos z (m)
pos error x (m)
pos error y (m)
pos error z (m)
att vector x
att vector y
att vector z
att vector error x
att vector error y
att angle error (deg)
vel norm (m/s)
omega norm (deg/s)
thrust (m/s²)
time (s)

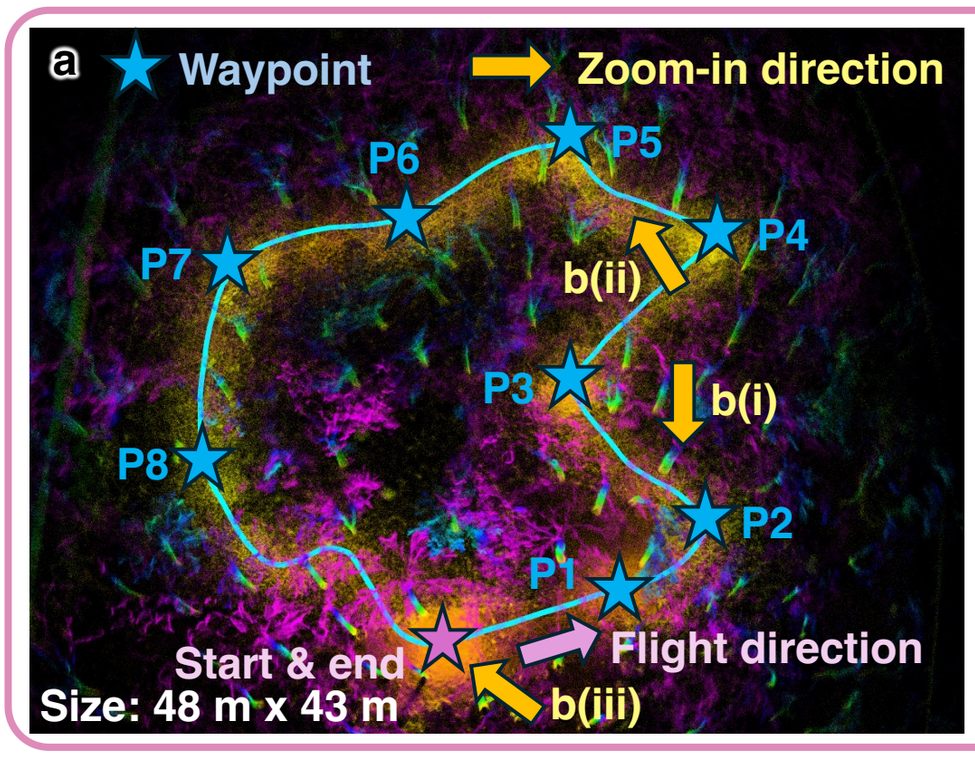

a (i)
a (ii)
a (iii)
a (iv)
z
x
y

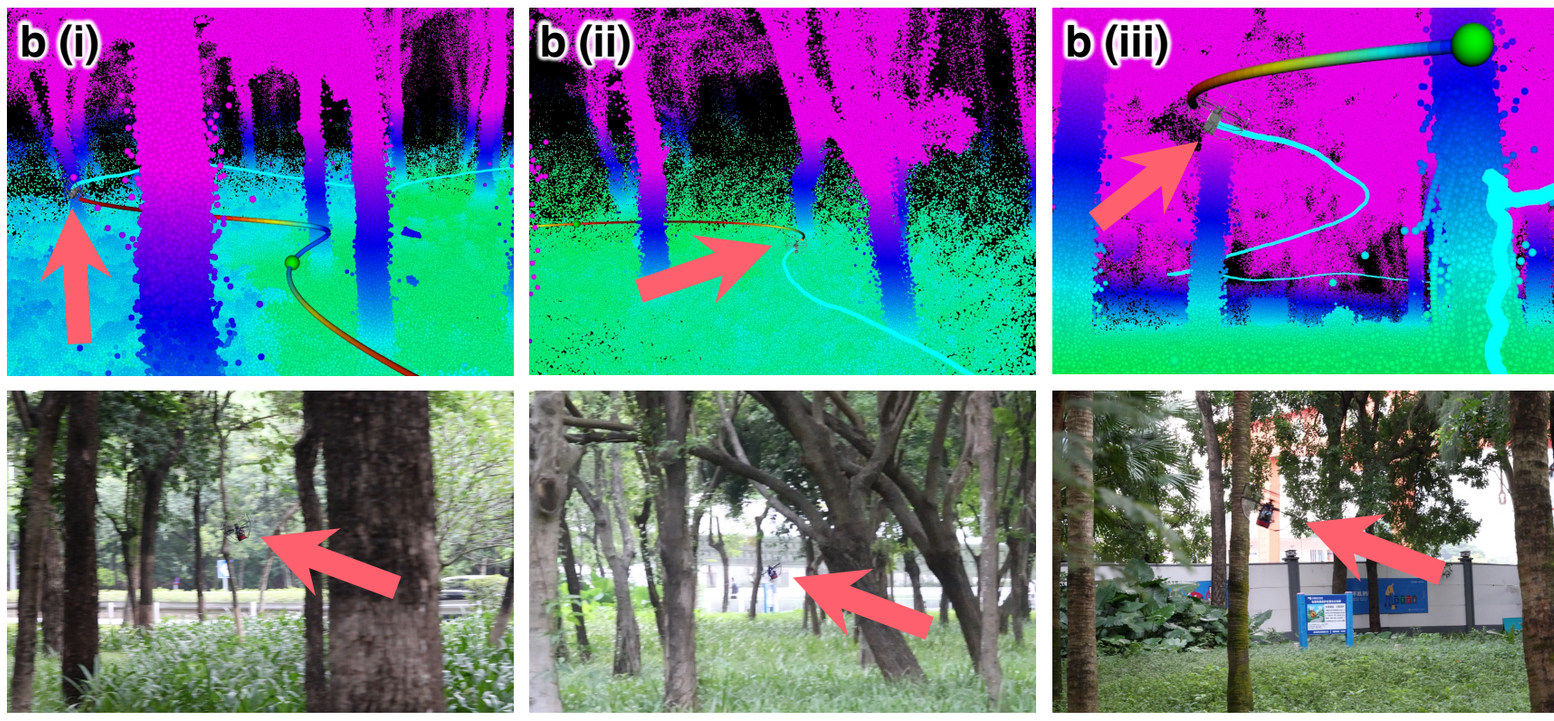

error curve (experiment 1)
error curve (experiment 2)
error curve (experiment 3)
b (i)
b (ii)
b (iii)
b (iv)
B frame control, w/o DoB
F frame control, w/o DoB
B frame control, w/ DoB
F frame control, w/ DoB
pos x (m)
pos y (m)
pos z (m)
att angle (deg)
time (s)

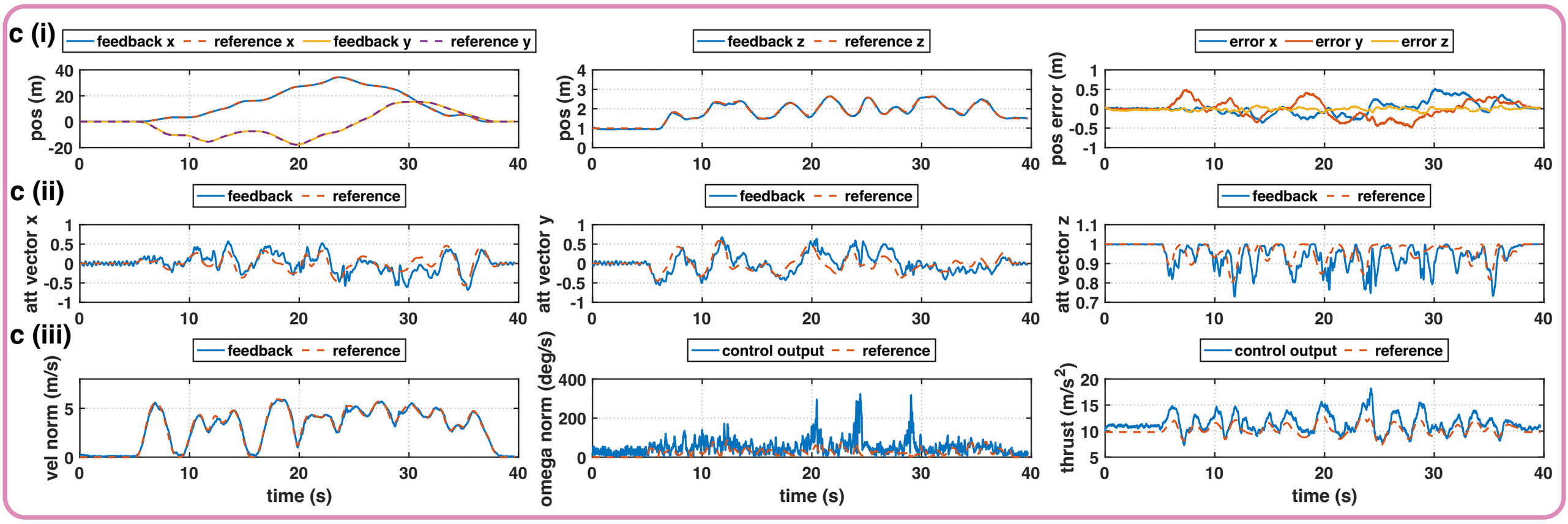

c (i)
c (ii)
c (iii)
c (iv)
pos error x (m)
pos error y (m)
pos error z (m)
att angle error (deg)
B frame, w/o DoB
F frame, w/o DoB
B frame, w/ DoB
F frame, w/ DoB

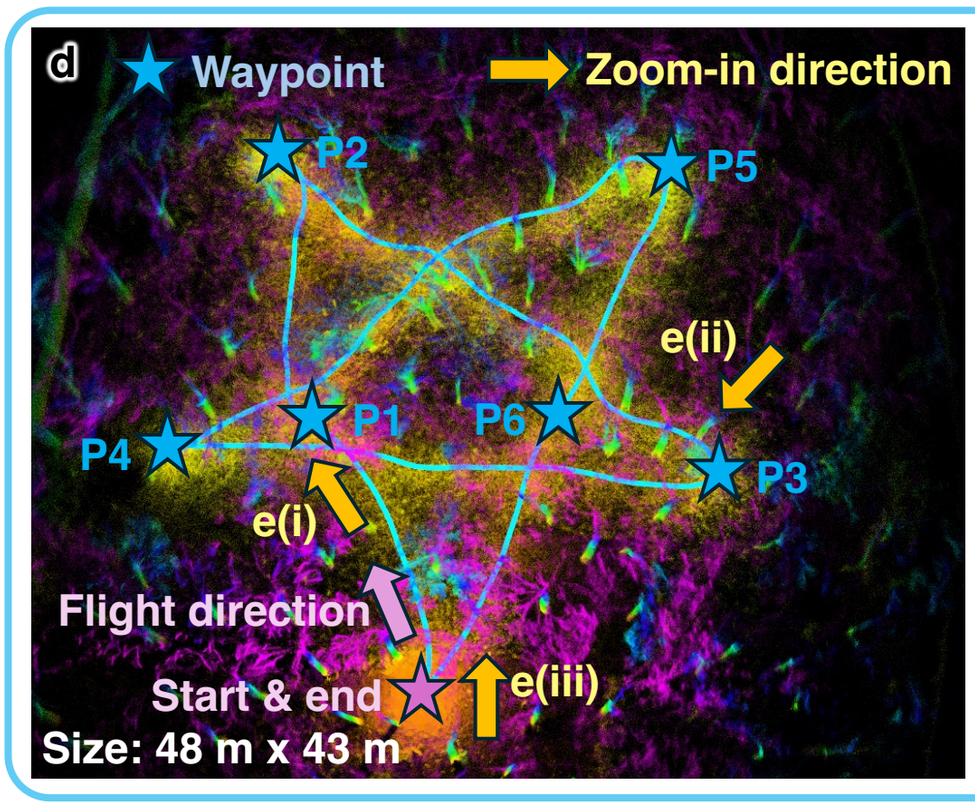

d (i)
d (ii)
d (iii)
d (iv)
z
x
y

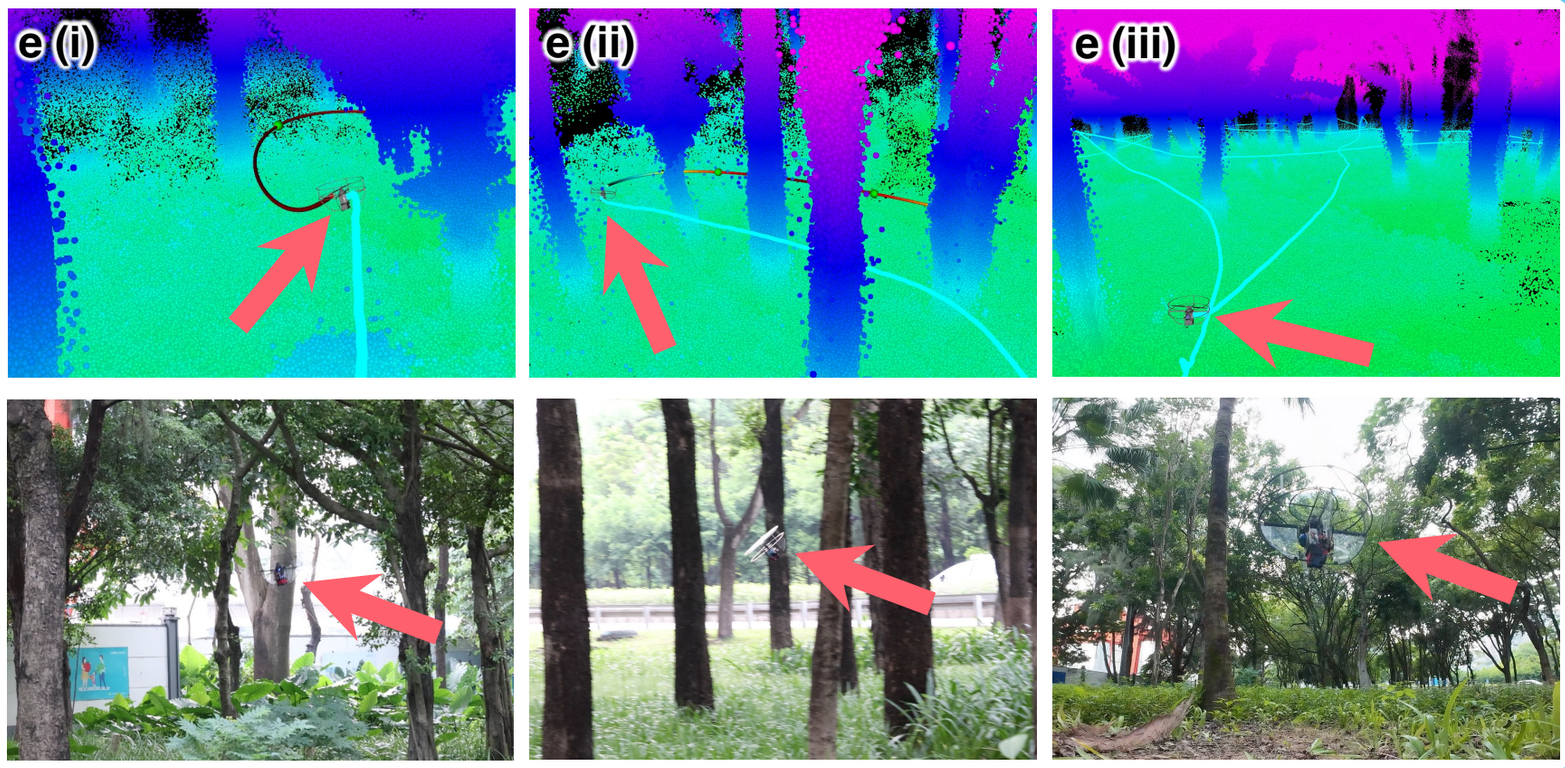

fb (B frame control, w/o DoB)
fb (F frame control, w/o DoB)
fb (B frame control, w/ DoB)
fb (F frame control, w/ DoB)
reference
e (i)
e (ii)
pos x (m)
pos y (m)
pos z (m)
vel norm (m/s)
att vector x
att vector y
att vector z
att err angle (deg)
time (s)

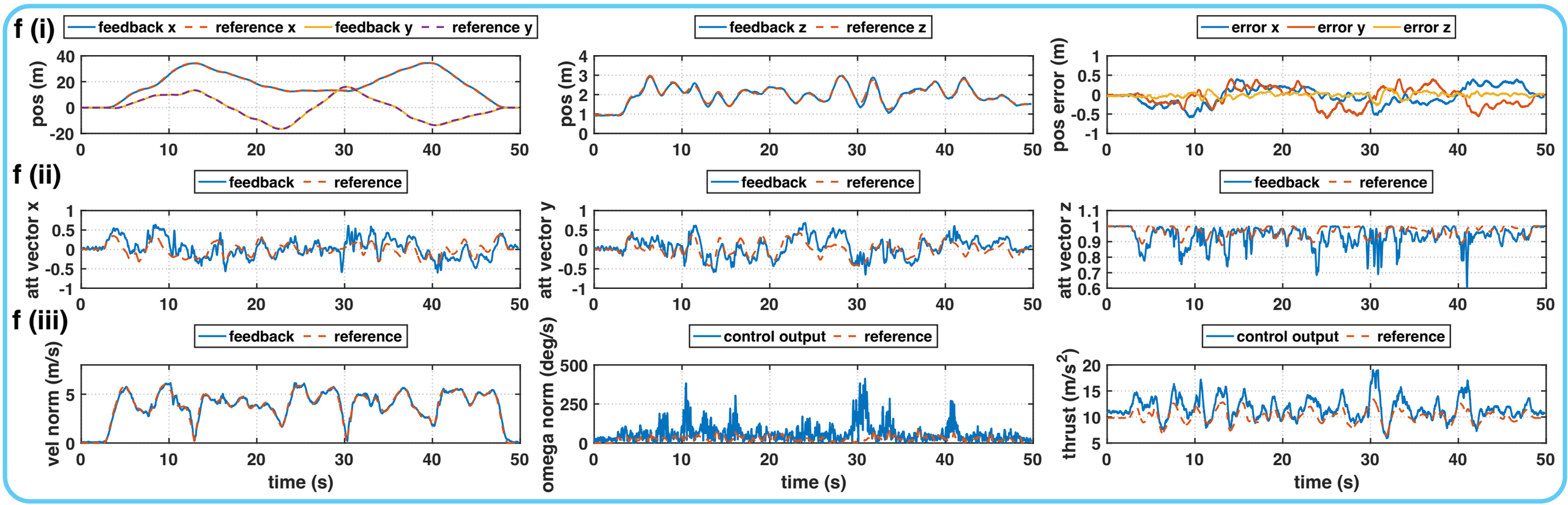

a
Waypoint
Zoom-in direction
P1
P2
P3
P4
P5
P6
P7
P8
b(i)
b(ii)
b(iii)
Start & end
Flight direction
Size: 48 m x 43 m

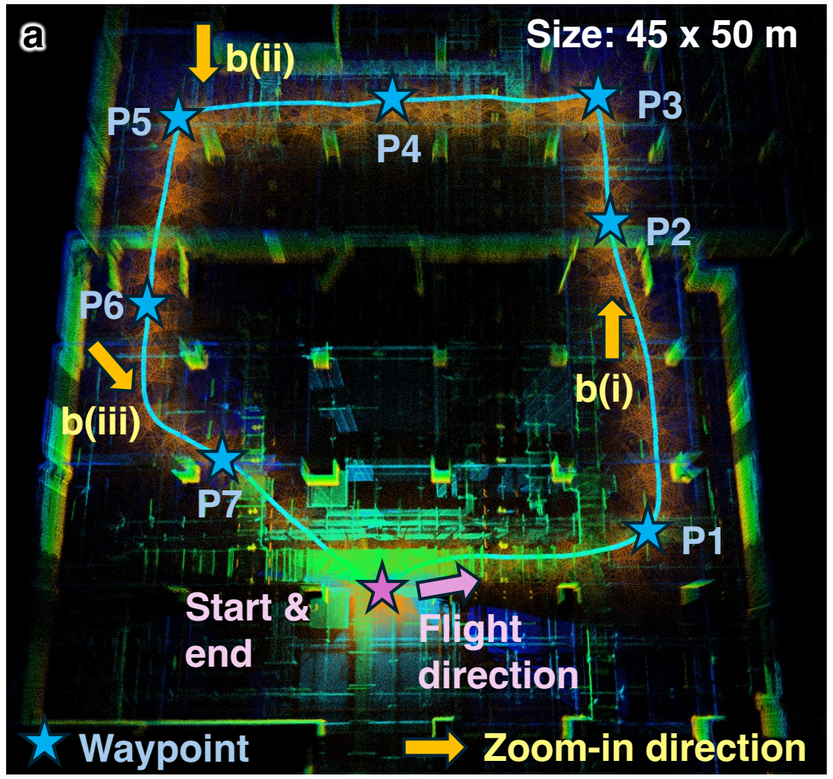

b (i)
b (ii)
b (iii)

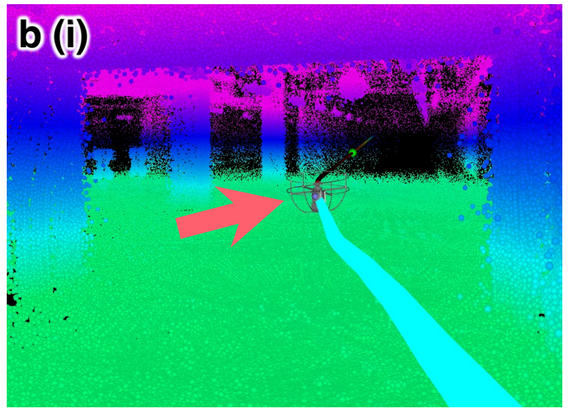

c (i)
feedback x
reference x
feedback y
reference y
pos (m)
feedback z
reference z
error x
error y
error z
pos error (m)
c (ii)
feedback
reference
att vector x
att vector y
att vector z
c (iii)
vel norm (m/s)
control output
omega norm (deg/s)
thrust (m/s2)
time (s)

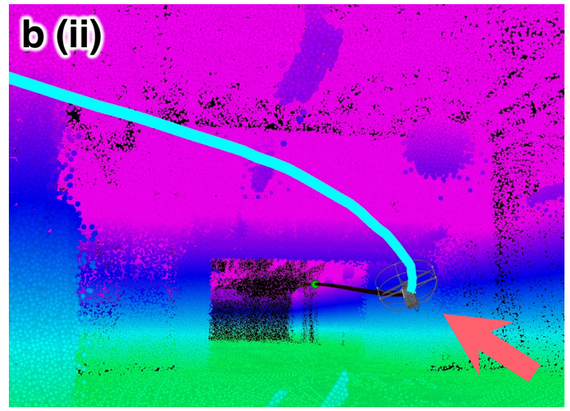

d
Waypoint
Zoom-in direction
P1
P2
P3
P4
P5
P6
e(i)
e(ii)
e(iii)
Flight direction
Start & end
Size: 48 m x 43 m

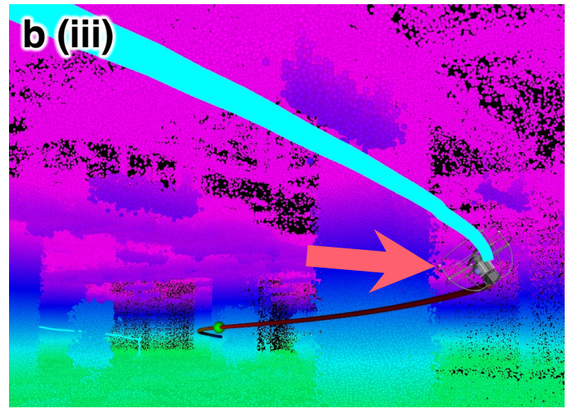

e (i)
e (ii)
e (iii)

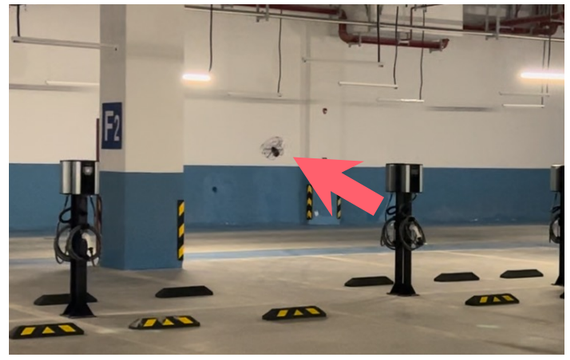

f (i)
feedback x
reference x
feedback y
reference y
pos (m)
feedback z
reference z
error x
error y
error z
pos error (m)
f (ii)
feedback
reference
att vector x
att vector y
att vector z
f (iii)
vel norm (m/s)
control output
omega norm (deg/s)
thrust (m/s2)
time (s)

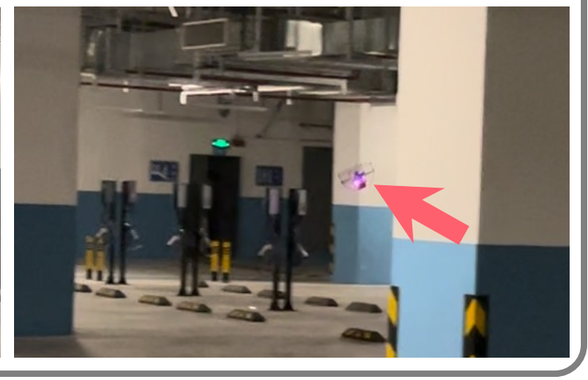

a
Size: 45 x 50 m
b(ii)
P5
P4
P3
P2
P6
b(i)
b(iii)
P7
P1
Start &
end
Flight
direction
Waponit
Zoom-in direction

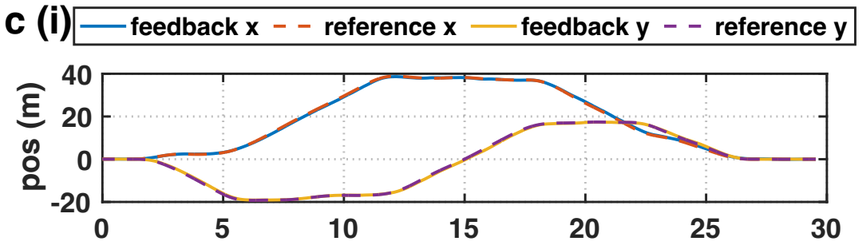

b (i)

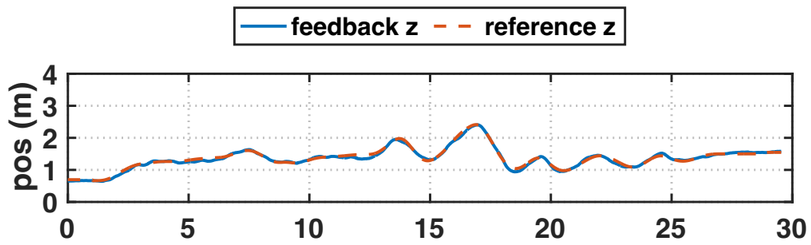

b (ii)

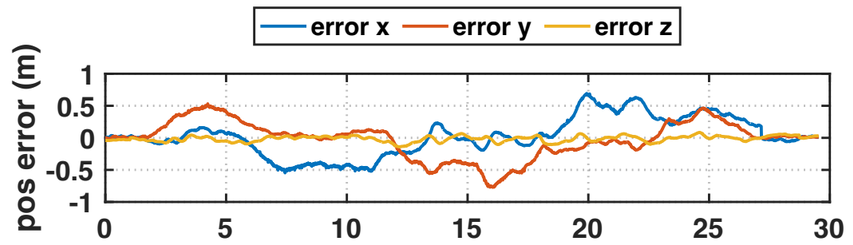

b (iii)

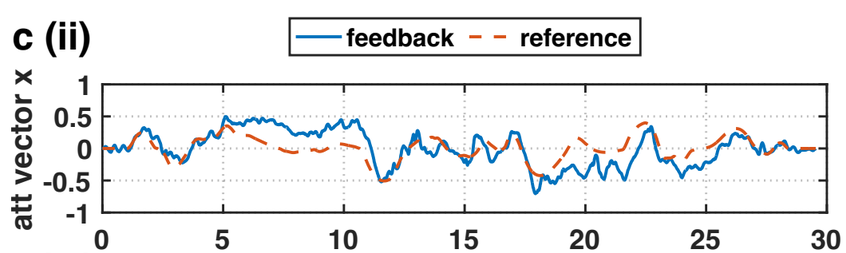


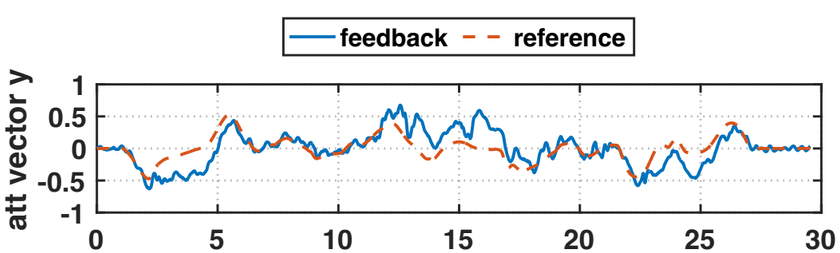


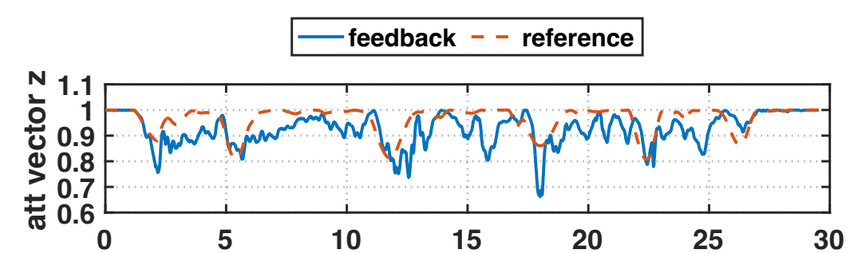

c (i)
feedback x
reference x
feedback y
reference y
pos (m)

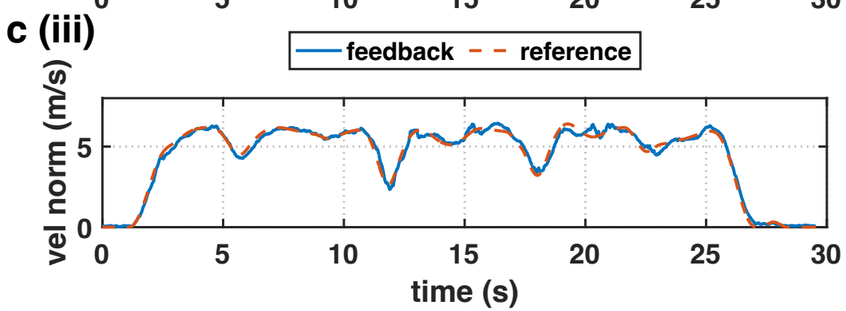

feedback z
reference z
pos (m)

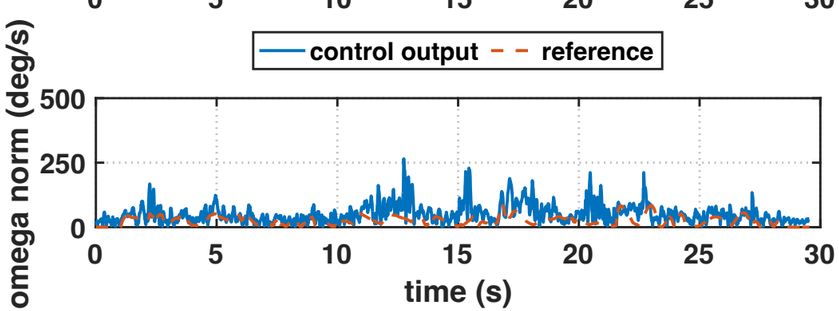

error x
error y
error z
pos error (m)

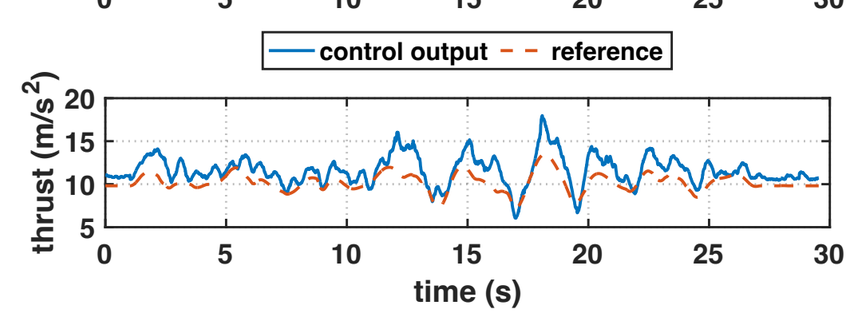

c (ii)
feedback
reference
att vector x

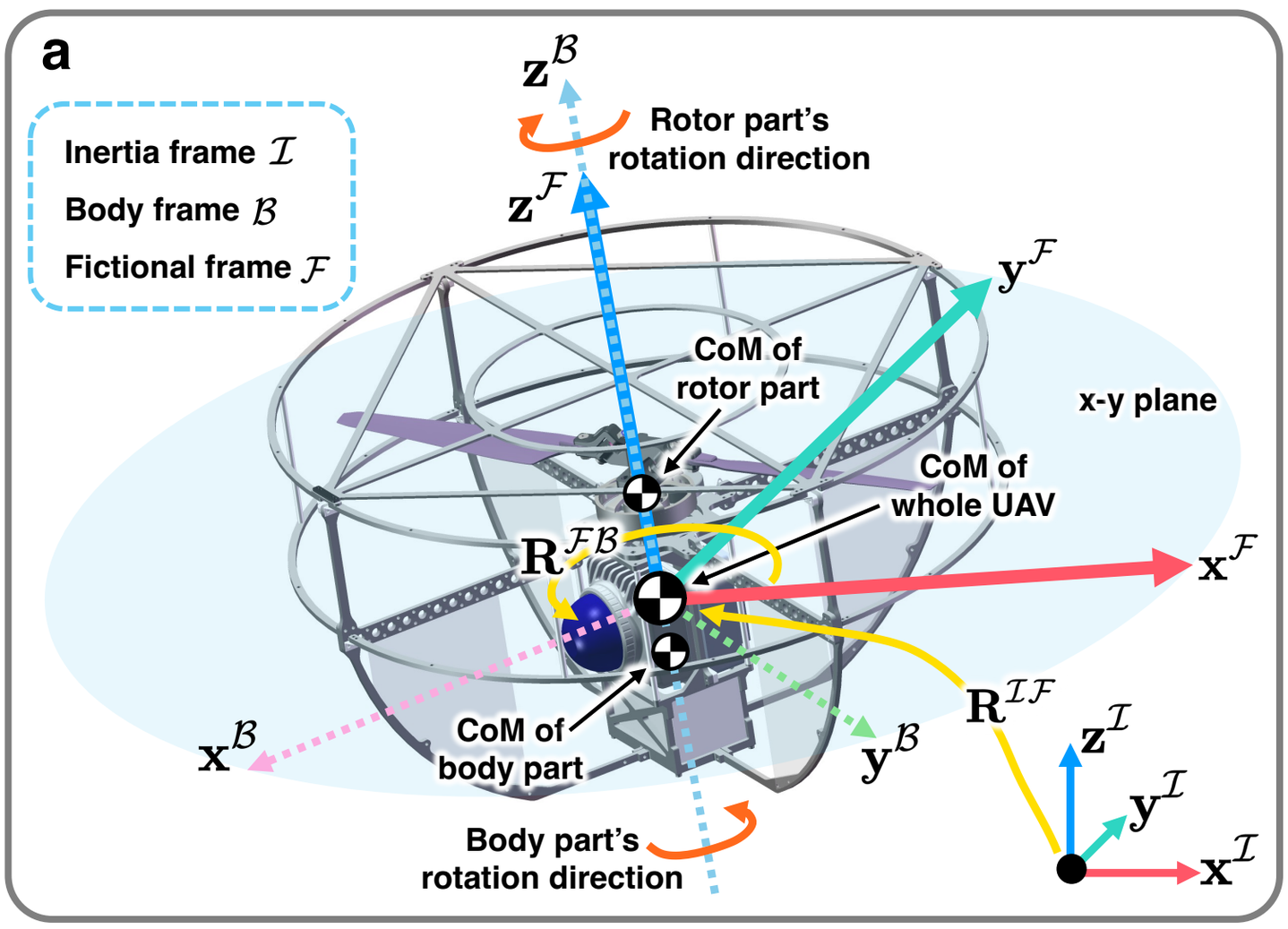

feedback
reference
att vector y

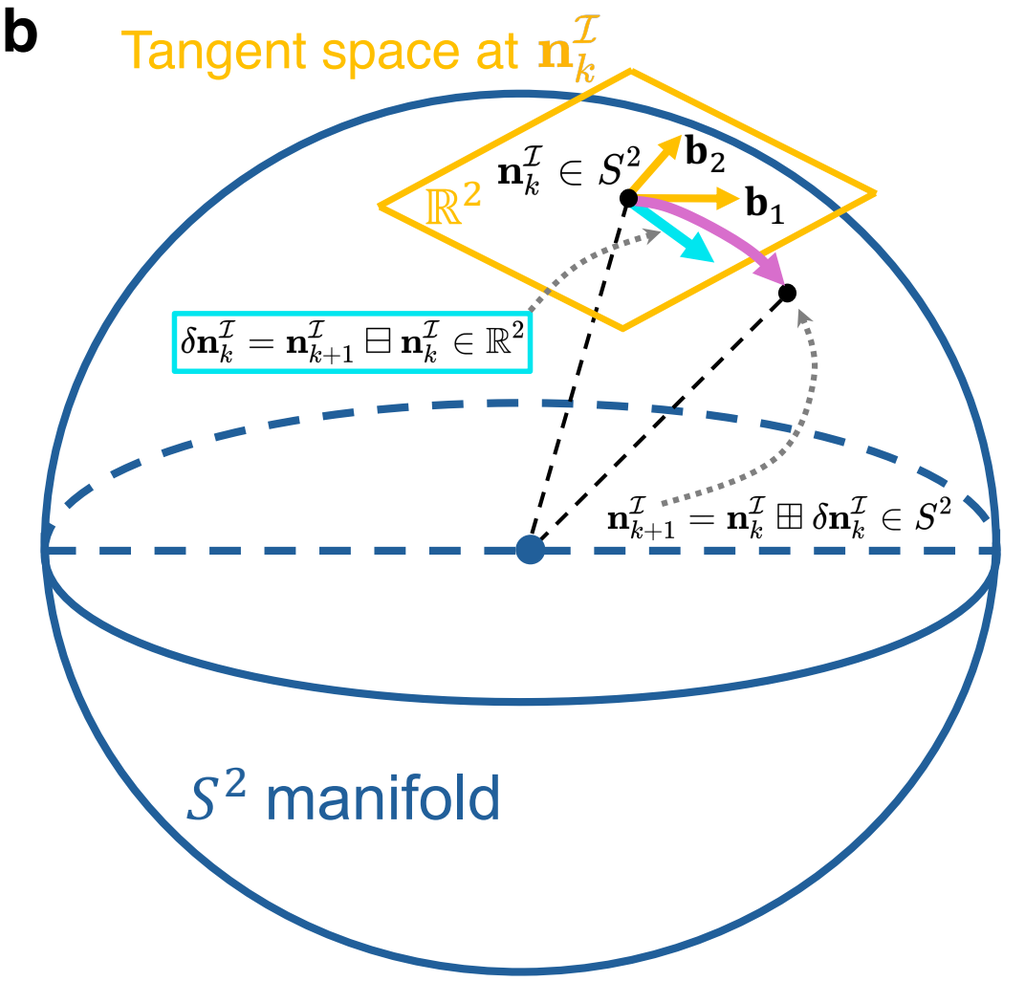

feedback
reference
att vector z

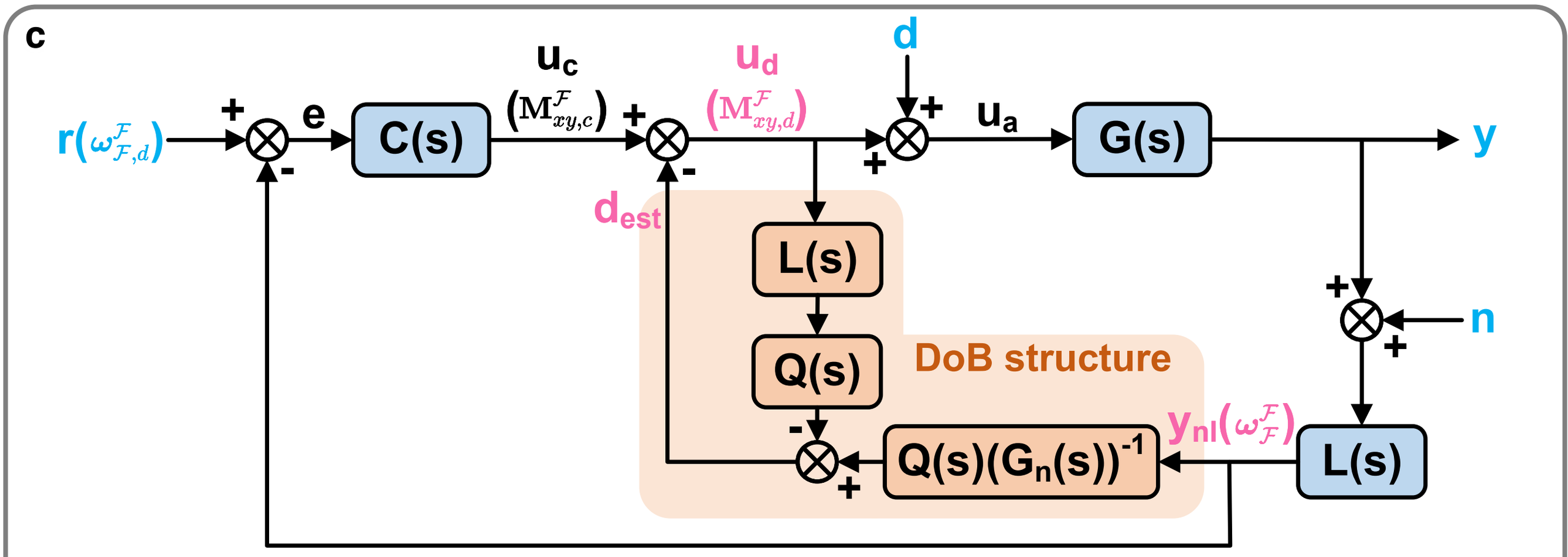

c (iii)
feedback
reference
vel norm (m/s)
time (s)

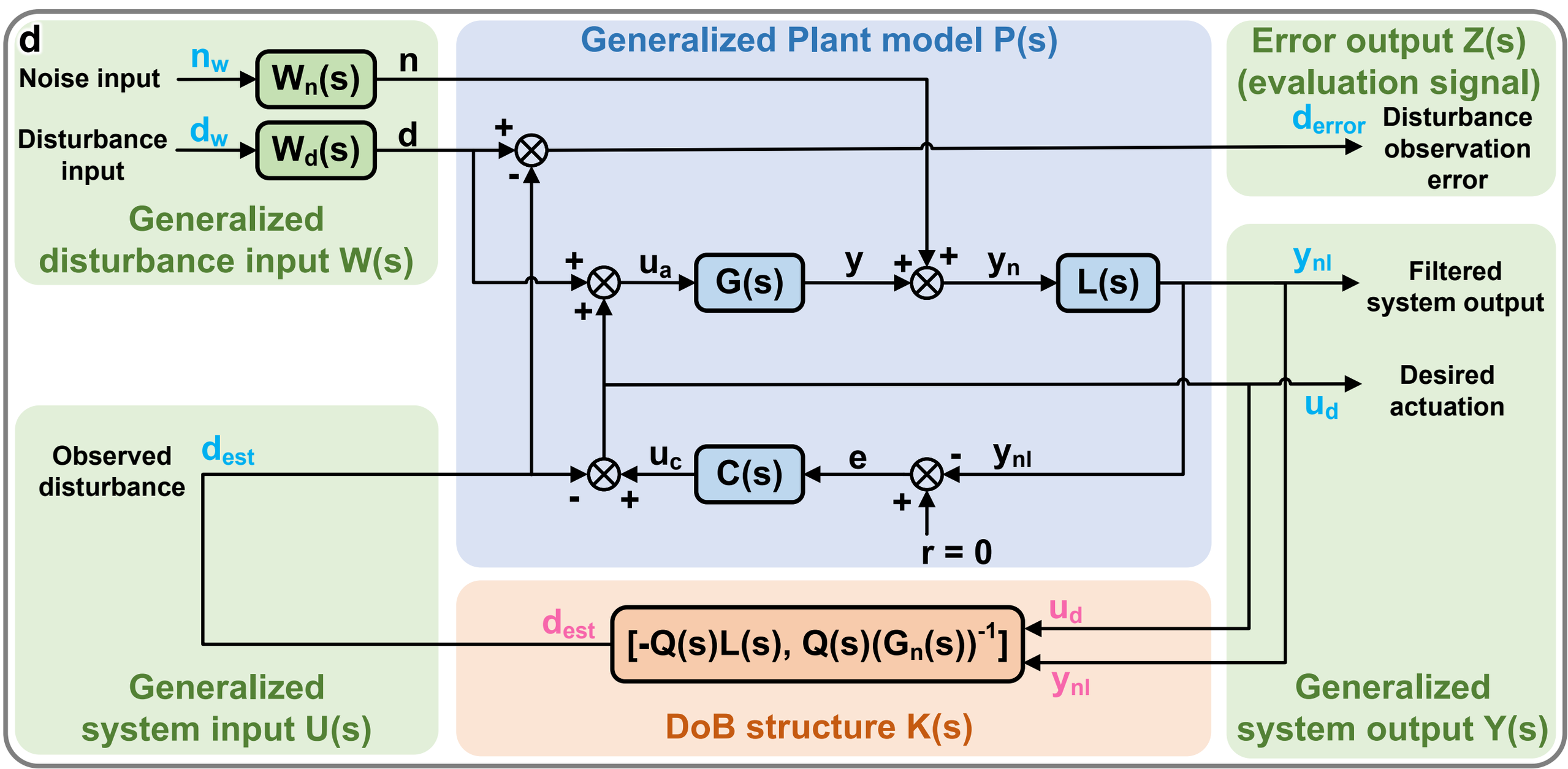

control output
reference
omega norm (deg/s)
time (s)


control output
reference
thrust (m/s2)
time (s)


a
Inertia frame $\mathcal{I}$
Body frame $\mathcal{B}$
Fictional frame $\mathcal{F}$
$z^{\mathcal{B}}$
Rotor part's rotation direction
$z^{\mathcal{F}}$
$y^{\mathcal{F}}$
CoM of rotor part
x-y plane
CoM of whole UAV
$R^{\mathcal{F}\mathcal{B}}$
$x^{\mathcal{F}}$
$x^{\mathcal{B}}$
CoM of body part
$y^{\mathcal{B}}$
$R^{\mathcal{I}\mathcal{F}}$
$z^{\mathcal{I}}$
$y^{\mathcal{I}}$
$x^{\mathcal{I}}$
Body part's rotation direction


b
Tangent space at $\mathbf{n}_k^{\mathcal{I}}$
$\mathbb{R}^2$
$\mathbf{n}_k^{\mathcal{I}} \in S^2$
$\mathbf{b}_2$
$\mathbf{b}_1$
$\delta\mathbf{n}_k^{\mathcal{I}} = \mathbf{n}_{k+1}^{\mathcal{I}} \boxminus \mathbf{n}_k^{\mathcal{I}} \in \mathbb{R}^2$
$\mathbf{n}_{k+1}^{\mathcal{I}} = \mathbf{n}_k^{\mathcal{I}} \boxplus \delta\mathbf{n}_k^{\mathcal{I}} \in S^2$
S^2 manifold


c
r($\omega_{\mathcal{F},d}^{\mathcal{F}}$)
+
-
e
C(s)
u_c
($M_{xy,c}^{\mathcal{F}}$)
+
-
u_d
($M_{xy,d}^{\mathcal{F}}$)
d
+
+
u_a
G(s)
y
+
n
d_{est}
L(s)
Q(s)
DoB structure
-
+
$Q(s)(G_n(s))^{-1}$
$y_{nl}(\omega_{\mathcal{F}}^{\mathcal{F}})$
L(s)
System signals: r (reference input), d (disturbance input), n (noise input), y (system output)
DoB signals: u_d (desired actuation), y_{nl} (filtered system output), d_{est} (observed disturbance)
Other signals: e (control error), u_c (actuation from controller), u_a (actual actuation)


d
Noise input
n_w
$W_n(s)$
n
Disturbance input
d_w
$W_d(s)$
d
Generalized disturbance input W(s)
Generalized system input U(s)
Observed disturbance
d_{est}
Generalized plant model P(s)
+
-
+
u_a
G(s)
y
+
+
y_n
L(s)
+
u_c
C(s)
e
-
y_{nl}
+
r = 0
DoB structure K(s)
d_{est}
$[-Q(s)L(s), Q(s)(G_n(s))^{-1}]$
u_d
y_{nl}
Error output Z(s) (evaluation signal)
d_{error}
Disturbance observation error
y_{nl}
Filtered system output
Desired actuation
u_d
Generalized system output Y(s)