AUTONOMOUS VEHICLES

Safety-assured high-speed navigation for MAVs

Yunfan Ren†, Fangcheng Zhu†, Guozheng Lu, Yixi Cai, Longji Yin, Fanze Kong, Jiarong Lin, Nan Chen, Fu Zhang*

Micro air vehicles (MAVs) capable of high-speed autonomous navigation in unknown environments have the potential to improve applications like search and rescue and disaster relief, where timely and safe navigation is critical. However, achieving autonomous, safe, and high-speed MAV navigation faces systematic challenges, necessitating reduced vehicle weight and size for high-speed maneuvering, strong sensing capability for detecting obstacles at a distance, and advanced planning and control algorithms maximizing flight speed while ensuring obstacle avoidance. Here, we present the safety-assured high-speed aerial robot (SUPER), a compact MAV with a 280-millimeter wheelbase and a thrust-to-weight ratio greater than 5.0, enabling agile flight in cluttered environments. SUPER uses a lightweight three-dimensional light detection and ranging (LIDAR) sensor for accurate, long-range obstacle detection. To ensure high-speed flight while maintaining safety, we introduced an efficient planning framework that directly plans trajectories using LIDAR point clouds. In each replanning cycle, two trajectories were generated: one in known free spaces to ensure safety and another in both known and unknown spaces to maximize speed. Compared with baseline methods, this framework reduced failure rates by 35.9 times while flying faster and with half the planning time. In real-world tests, SUPER achieved autonomous flights at speeds exceeding 20 meters per second, successfully avoiding thin obstacles and navigating narrow spaces. SUPER represents a milestone in autonomous MAV systems, bridging the gap from laboratory research to real-world applications.

INTRODUCTION

Birds have long captivated humans with their exceptional flight capabilities to navigate at high speeds through cluttered environments, with remarkably low failure rates. Similarly, micro air vehicles (MAVs), among the most agile machines created by humans (*1*), hold the potential to achieve bird-like high-speed agile flights. The rapid and safe arrival of MAVs at their destinations is a critical factor for successfully deploying MAVs in practical applications. Being fast implies that an MAV is able to reach designated locations in a timely manner, enabling rapid response in time-critical missions like search and rescue (*2*) or disaster relief (*3*, *4*). Being safe means that the MAV could detect and avoid obstacles on the way without a collision failure. In this work, we explore how to endow MAVs with bird-like capabilities, relying solely on onboard sensing and computational units, to achieve safe and high-speed flights in unknown environments (Movie 1).

Achieving safety-assured, high-speed flights in unknown environments is a complex task that necessitates a holistic system design. First, to execute aggressive maneuvers that avoid previously unknown obstacles during high-speed flights, the MAV must have high agility, characterized by its compact size and high thrust-to-weight ratio (TWR). Second, the MAV must have a long detection range to provide sufficient reaction time for avoiding obstacles at high speeds. Moreover, such detection ability should be achieved with lightweight, compact sensors to not degrade the MAV's agility. Last, the MAV has to carefully balance flight speed with safety in its trajectory planning. Flight speed and safety are two mutually conflicting factors, and their balance has to be achieved efficiently with the limited computation power onboard the MAV.

Several existing works have focused on addressing the challenges in autonomous drone racing applications (*5*–*9*) and have achieved notable results in terms of flight speed and agility. Song *et al.* (*5*) used reinforcement learning (RL) techniques to achieve flight speeds exceeding 30 m/s on a human-made racing track, using external computing for control and a motion capture system for state feedback. Foehn *et al.* (*6*) proposed a strategy based on progress optimization to compute a time-optimal trajectory offline and track the trajectory with an onboard computer, whereas Romero *et al.* proposed a sample-based online planning approach (*7*) and used model predictive contouring control (*8*) to track the trajectory, achieving high-speed racing flights greater than 18 m/s. Kaufmann *et al.* (*9*), following a similar RL approach, combined onboard sensing and computation to surpass the performance of champion-level human pilots in racing missions. Despite achieving notable flight speed and agility, these methods depend on external sensing (such as a motion capture system) (*5*–*8*), external computation during actual flights (*5*), or offline training (*5*, *6*, *9*). Moreover, these methods assume a fixed, known environment (such as the racing track) (*5*–*9*), where the control policy is exhaustively trained for the best performance. Their applicability to unknown environments is not clear.

Autonomous flights in unknown environments using only onboard sensing and computation have been extensively investigated in the literature. The first set of approaches prioritizes flight speed (*10*–*17*). Escobar-Alvarez *et al.* adopted an expansion rate (ER)–based method to achieve high-speed flight ranging from 6 to 19 m/s in a relatively open area (*17*), using visual sensors and a single-line time-of-flight (TOF) light detection and ranging (LIDAR) sensor. Loquercio *et al.* (*16*) used imitation learning to enable autonomous flight under challenging conditions, using an end-to-end approach and achieving a maximum speed exceeding 10 m/s.

Another approach is the Bubble planner proposed by Ren *et al.* (*13*), which used receding horizon corridors for trajectory optimization, resulting in a maximum speed of 13.7 m/s in complex environments. An inherent issue with these works is their exclusive focus on flight speed at the expense of safety guarantees. To maximize flight speed, existing works (*10*–*15*, *17*) all assume that the unknown regions caused by occlusions or

Department of Mechanical Engineering, University of Hong Kong, Pokfulam, Hong Kong, China.
†These authors contributed equally to this work.
*Corresponding author. Email: fuzhang@hku.hk

Movie 1. Overview of the proposed SUPER system. SUPER demonstrates its ability to safely navigate through unknown, cluttered environments at high speeds; avoid thin obstacles like power lines; and perform robustly in various scenarios, including object tracking and autonomous exploration.

limited sensor field of view (FOV) are free of obstacles during trajectory planning. Such an optimistic trajectory poses a risk of colliding with hidden obstacles in the unknown region, particularly in cluttered environments with frequent occlusions given that high-speed flights require sufficient space to decelerate. Similarly, the learning-based approach in (*16*) does not incorporate safety considerations into its control policy, resulting in a low overall success rate (~50 to 80%) when the flight speed exceeds 10 m/s. These limitations in success rate and the absence of safety guarantees restrict the deployment of these methods (*10*–*16*) in real-world applications.

Alternatively, there are approaches that prioritize flight safety. The first stream of methods focuses on enhancing flight safety in a safety-aware manner (*18*–*20*). Quan *et al.* (*18*) and Wang *et al.* (*19*) achieved safety awareness by actively slowing down the MAV when approaching unknown spaces, whereas Zhou *et al.* (*20*) maximized the visibility to unknown spaces on the trajectory. Although these methods could enhance flight safety, they compromise flight speed. Field experiments reported in safety-aware methods (*18*–*20*) indicate that their designs are rather conservative and cannot fully use the MAV agility, leading to a flight speed of no more than 5 m/s. Moreover, these methods rely on various heuristic strategies for safety awareness and lack a rigorous safety guarantee. Another category of methods adopts safety-assured strategies that confine the trajectory completely within known free spaces (*21*, *22*). Although they offer a guaranteed level of safety, the safety-assured methods (*21*, *22*) tend to be even more conservative in flight speed, especially in cluttered environments. Such conservatism was partially overcome in Faster by Tordesillas *et al.* (*23*), which plans an additional trajectory in both unknown and known free space alongside the trajectory in the known free space. This two-trajectory planning strategy has notably improved flight speed, achieving a fast flight of greater than 8 m/s, several times faster than those reported in prior works (*21*, *22*), without losing the guarantee of safety. However, Faster suffers from high computation delay because it uses a computationally expensive occupancy grid map (OGM) to identify known free spaces and mixed-integer quadratic programming (MIQP) for trajectory optimization. The high computation delay severely limits the flight speed and success rate.

In addition to the planning strategy, the sensing capability onboard the MAV plays another crucial role in achieving high-speed flights. Existing works on autonomous MAV navigation (*14*–*16*, *18*–*21*, *23*) often used vision sensors for localization and obstacle perception. Vision sensors suffer from several limitations, such as limited sensing range (typically 3 to 5 m), low dynamic range (*24*), and susceptibility to strong motion blur. These factors severely restrict the attainable flight speeds and level of safety. Moreover, the performance of vision-based systems is affected by lighting conditions, thereby imposing additional constraints on their practicality in real-world applications where insufficient illumination or large illumination variations are present.

In this work, we present a safety-assured high-speed aerial robot (SUPER) to fulfill the task of safe, high-speed flights in unknown environments (Fig. 1 and Movie 1). The primary sensing modality of SUPER is a three-dimensional (3D) LIDAR sensor delivering a 70-m sensing range at centimeter-level accuracy, all within a compact form factor and lightweight configuration (*25*). With all sensing, computing, and other necessary components onboard (see the system breakdown in Supplementary Methods), SUPER achieves a TWR exceeding 5.0 [versus 0.87 for an F35 fighter aircraft at gross weight (*26*)] and a compact size, with a 280-mm wheelbase [versus 380 mm for that of DJI Mavic 3 (*27*)], which enables agile flights in cluttered environments. To achieve high-speed flights while ensuring safety, we adopted the two-trajectory planning strategy proposed in (*28*) and planned two trajectories in each replanning cycle: one in known free space to ensure safety and the other in both known free space and unknown space to boost flight speed. We improved the computation efficiency of the two-trajectory strategy by one order of magnitude by redesigning the planning and mapping modules. For mapping, we proposed a method to distinguish known free spaces directly on LIDAR points, which fundamentally enables the use of a more efficient point cloud map in the two-trajectory strategy. For trajectory planning, we used a differentiable trajectory parameterization, specifically the minimum control effort (MINCO) method (*29*), to improve the efficiency of trajectory optimization and to optimize switching time between the two trajectories. The improved computation efficiency and optimally determined switching time improved the attainable flight speeds and success rate as demonstrated in the benchmark comparison. Last, the point cloud map effectively retained measurements on thin objects and represented the environments at centimeter-level accuracy, allowing the unmanned aerial vehicle (UAV) to avoid thin obstacles and navigate through tight spaces. The robustness of SUPER against small objects, cluttered environments, and lighting variations expanded its operation envelope in both natural and artificial environments, enabling round-the-clock operations spanning both daytime and nighttime. As a result, SUPER represents a milestone in transitioning high-speed autonomous navigation from laboratory settings to real-world applications.

RESULTS

Overview of SUPER

We have demonstrated the safe and high-speed flight capabilities of SUPER through extensive simulations and real-world experiments. In more than 1000 simulated flights, the planning module of SUPER reduced the failure rate by 35.9 to 95.8 times compared with the state-of-the-art baseline methods while achieving notably higher

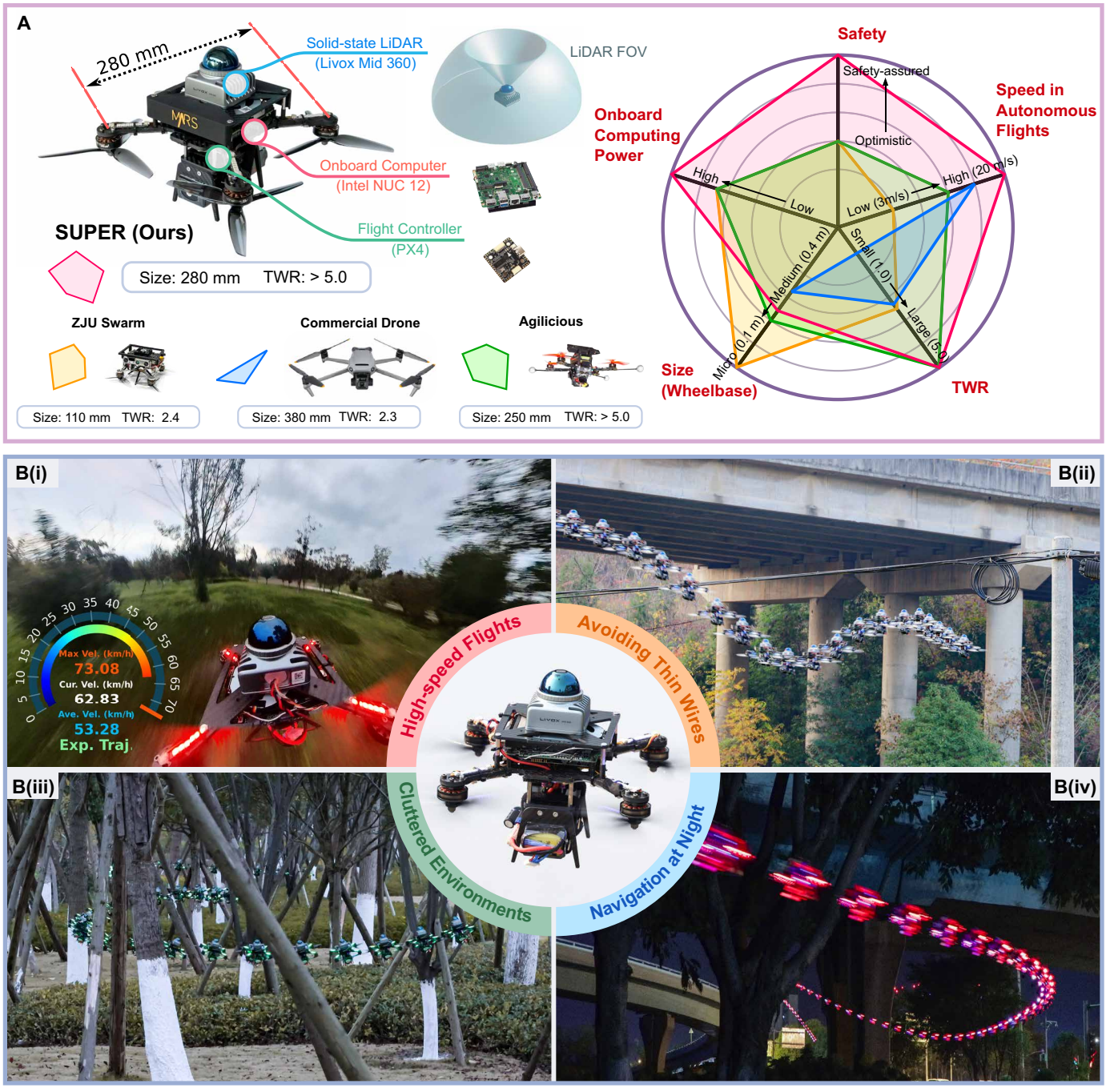

Fig. 1. Overview of the proposed autonomous aerial system. (**A**) Radar chart comparing SUPER with other state-of-the-art autonomous aerial robots, including the palm-sized autonomous MAV (ZJU Swarm) (*35*), the open-source agile quadrotor platform (Agilicious) (*33*), and a representative commercial drone (*27*). For Agilicious, we obtained the data from its best performance in onboard autonomous flights as reported in (*16*). For the commercial drone, we obtained its size, TWR, and the maximum flight speed allowed by its APAS from the official manufacturer's website (*27*). The onboard computation power and safety strategy are not publicly disclosed and hence are not shown. (**B**) Demonstration of SUPER in real-world flights. (i) High-speed flights in the wild; (ii) avoidance of thin electrical wires; (iii) navigation through cluttered environments; (iv) flights at night.

average flight speeds. The performance of SUPER in real-world scenarios has also been validated with a substantial number of field tests. In all experiments, SUPER was tasked to reach a goal position with fully onboard perception, planning, and control. The line connecting the goal and the MAV's current position was usually not collision free, and the environment was completely unknown before the flight, requiring SUPER to detect and avoid those obstacles in the environment in an online manner. In these real-world tests, SUPER achieved a speed of 20 m/s in unknown, unstructured environments and maintained a 100% success rate across eight different

trials (Fig. 1B, i). Moreover, SUPER outperformed advanced commercial MAVs regarding flight safety against small obstacles and adaptability to environments of high clutter and low lighting conditions. In particular, SUPER was able to detect and avoid thin objects, such as power lines and tree branches, in the wild (Fig. 1B, ii) and navigate through extremely cluttered environments (Fig. 1B, iii), even at night (Fig. 1B, iv). SUPER was also successfully applied in object tracking, autonomous exploration, and waypoint navigation missions (see Fig. 2), demonstrating its robustness and versatility in real-world scenarios. Video recordings of some flights are supplied in Movie 1.

Safe high-speed navigation in unknown environments

We tested the flight speed of SUPER in an unknown forest environment (see Fig. 3 and movie S1). The environment covered an area of around 280 by 90 m^2 and featured trees of varying thicknesses and diverse natural vegetation (Fig. 3A, i). We conducted eight experiments at different times of day, creating a wide range of lighting conditions from normal bright day to completely dark night (Fig. 3A, ii to v). We also set different maximum speed constraints across the eight experiments, ranging from 5 to 20 m/s. In all experiments, SUPER started from position $\mathbf{p}_s$ and was assigned to reach four waypoints $\mathbf{p}_1$~$\mathbf{p}_4$ sequentially, where $\mathbf{p}_4$ coincides with $\mathbf{p}_s$ (Fig. 3B, i). The four waypoints guided the MAV to experience areas of different obstacle densities (Fig. 3B, ii to iv).

SUPER succeeded in all eight experiments, achieving a 100% success rate. Figure 3C illustrates the speed distribution in the eight experiments along with the maximum speed constraints. SUPER reached a maximum flight speed of 20 m/s in tests 6 to 8. Moreover, tests 6 to 8 were conducted during daytime, dusk, and nighttime, respectively, with the same maximum speed limit of 20 m/s. SUPER consistently demonstrated high performance in these three experiments, demonstrating the robustness of our system to variations under lighting conditions. Besides high-speed flights, SUPER was able to adapt to different speeds and consistently met the speed constraints in all experiments. Figure 3D presents the distribution of position tracking errors in the eight experiments. SUPER exhibited an average tracking error of 0.13 m, even at a speed of 20 m/s, showcasing high tracking precision and ensuring stable performance during high-speed flights.

Navigation in cluttered environments

To validate SUPER's navigation capability in complex environments, we commanded it to track a person jogging in a dense wooded environment (Fig. 4A). The person passed two wooded areas sequentially: The first area had a relatively sparse distribution of trees (Fig. 4C), and the second one presented a higher density where the tracked person had to lower her body to pass through (Fig. 4D). We compared SUPER's tracking and navigation performance with a state-of-the-art commercial product (*27*) by conducting two separate experiments. In both experiments, the tracked person followed roughly the same route and speed. Considering that the commercial drone has a larger size than SUPER (0.31 m versus 0.21 m in radius), we increased SUPER's size to 0.32 m by adding four sticks on each arm to ensure a fair comparison (Fig. 4B). For target detection and tracking, the commercial drone used a visual recognition technique that is not available in SUPER. To work around this problem, the target wore a high-reflectivity vest, which can be easily detected in LIDAR point clouds on the basis of the point reflectivity measurements. Both systems demonstrated successful detection and tracking of the target throughout the entire task, ensuring a fair comparison for their navigation capabilities.

The tracking trajectories of both MAVs are presented in Fig. 4A. In the beginning stage where the target is in open areas, the commercial drone and SUPER can achieve a comparable tracking performance. After the target entered the first wooded area, the commercial drone failed to track the target and stopped in front of the woods (position A in Fig. 4, A and C, i). A likely reason is that the visual navigation used by the commercial drone had a limited map resolution and accuracy (typically tens of centimeters), which necessitated more conservative collision-free trajectories to enhance safety. Subsequently, the target exited the first wooded area and ran toward the second one. The commercial drone successfully resumed tracking by finding a path detouring the wooded area (the orange line between positions A and B in Fig. 4A). However, after the target entered the second wooded area with a higher density, the commercial drone failed the tracking mission completely and disengaged from the automatic tracking mode (Fig. 4D, i) for the same reason mentioned above. In contrast, SUPER planned its trajectory directly on LIDAR point clouds with centimeter-level accuracy, enabling it to navigate through small corridors in cluttered environments. As a consequence, SUPER exhibited smooth and consistent target tracking throughout the entire mission, successfully navigating both wooded areas without a stop encountered by the commercial drone (Fig. 4, C, ii and D, ii). The tracking processes of the commercial drone and SUPER in this experiment are shown in movie S2.

Avoiding thin objects

Thin objects, such as power lines and tree branches, present challenges for MAVs during field missions because of their abundance in the wild and difficulties in detection. To validate SUPER's ability to avoid small objects, we conducted a series of quantitative experiments and compared its performance with that of a commercial drone (*27*). As shown in Fig. 5, we selected four thin wires of different diameters: 30, 20, 11, and 2.5 mm (Fig. 5A). The wires were hung between two trees ~3 m apart. To test the ability to avoid the thin wires, the MAV started at a position 4 m away on one side of the wire, with the target position 4 m away on the opposite side. Because the commercial drone does not accept waypoint commands, we used joysticks to command it to fly from the start position to the target. To test whether the commercial drone is able to detect and avoid the thin wires on the flight path, we switched on its Advanced Pilot Assistance System (APAS) during the flight. The flight speed was ~2 to 3 m/s, which was also set as the speed constraint for SUPER for a fair comparison.

The experimental results are shown in Fig. 5. In the case of the 30-mm wire, both SUPER and the commercial drone successfully avoided the wire and reached the target position (Fig. 5, B, i and C, i). However, in the experiments with the thinner wires with diameters of 20, 11, and 2.5 mm, the commercial drone failed to detect and avoid them. Consequently, it crashed on the wire and caused the wire to swing (Fig. 5B, ii to iv). The commercial drone's failure to detect thin wires may be attributed to challenges in accurately computing disparities and generating depth images for navigation with the vision-based navigation system. In contrast, SUPER reached its target position successfully in all four experiments, avoiding even the thinnest 2.5-mm wire (Fig. 5C, i to iv). The exceptional obstacle avoidance capability is attributed to both the LIDAR sensor and our planning module: The LIDAR sensor provides point measurements

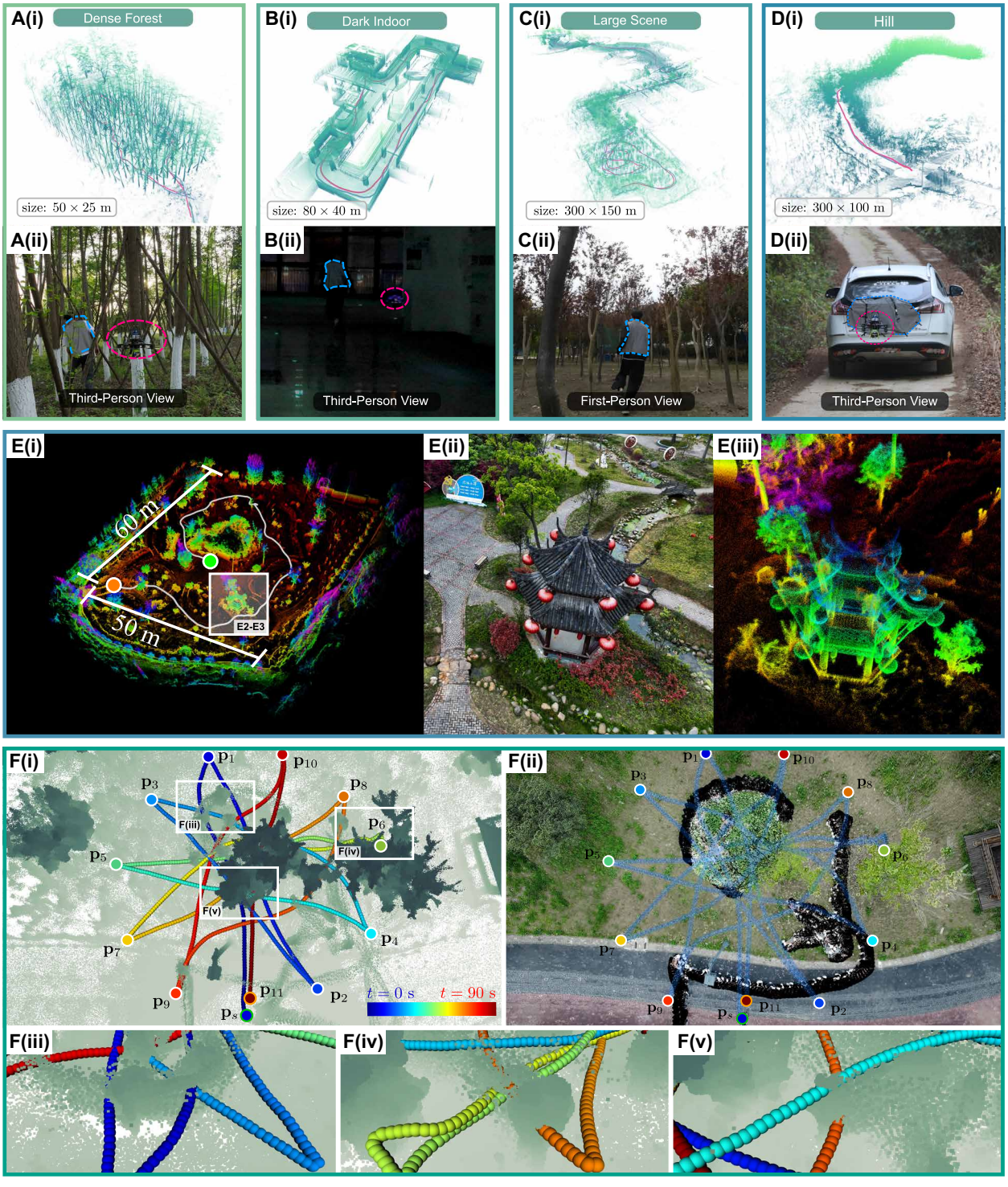

Fig. 2. Additional applications of SUPER. Examples of object tracking including (**A**) tracking of a person running in a dense forest, (**B**) a dark indoor scene, (**C**) a large scene, and (**D**) a car moving on a hilly village road. (A)(i) and (D)(i) show the point cloud maps, which were synthesized from online LIDAR measurements after the flights were completed, with trajectories of the target and the MAV being colored in blue and red, respectively. (A)(ii) and (D)(ii) show snapshots in either third-person or first-person view during the experiments. (**E**) (i) Autonomous exploration experiment in a 50 m–by–60 m area with the executed trajectory of SUPER shown in the white path. (ii and iii) Snapshots of the scene and the corresponding reconstruction results, respectively. (**F**) Example of waypoint navigation. (i) The MAV starting from $\mathbf{p}_s$ visited 11 waypoints, $\mathbf{p}_1 \sim \mathbf{p}_{11}$, sequentially. The MAV trajectory is color coded on the basis of time. (ii) Time-lapse image composition capturing the changes in the environments and the flight trajectories of the MAV. The black artifacts are caused by the two persons moving randomly in the area, and the MAV trajectory is highlighted in blue. (iii to v) MAV flight trajectories at locations traveled by moving persons. SUPER can use the spaces previously traversed by moving objects.

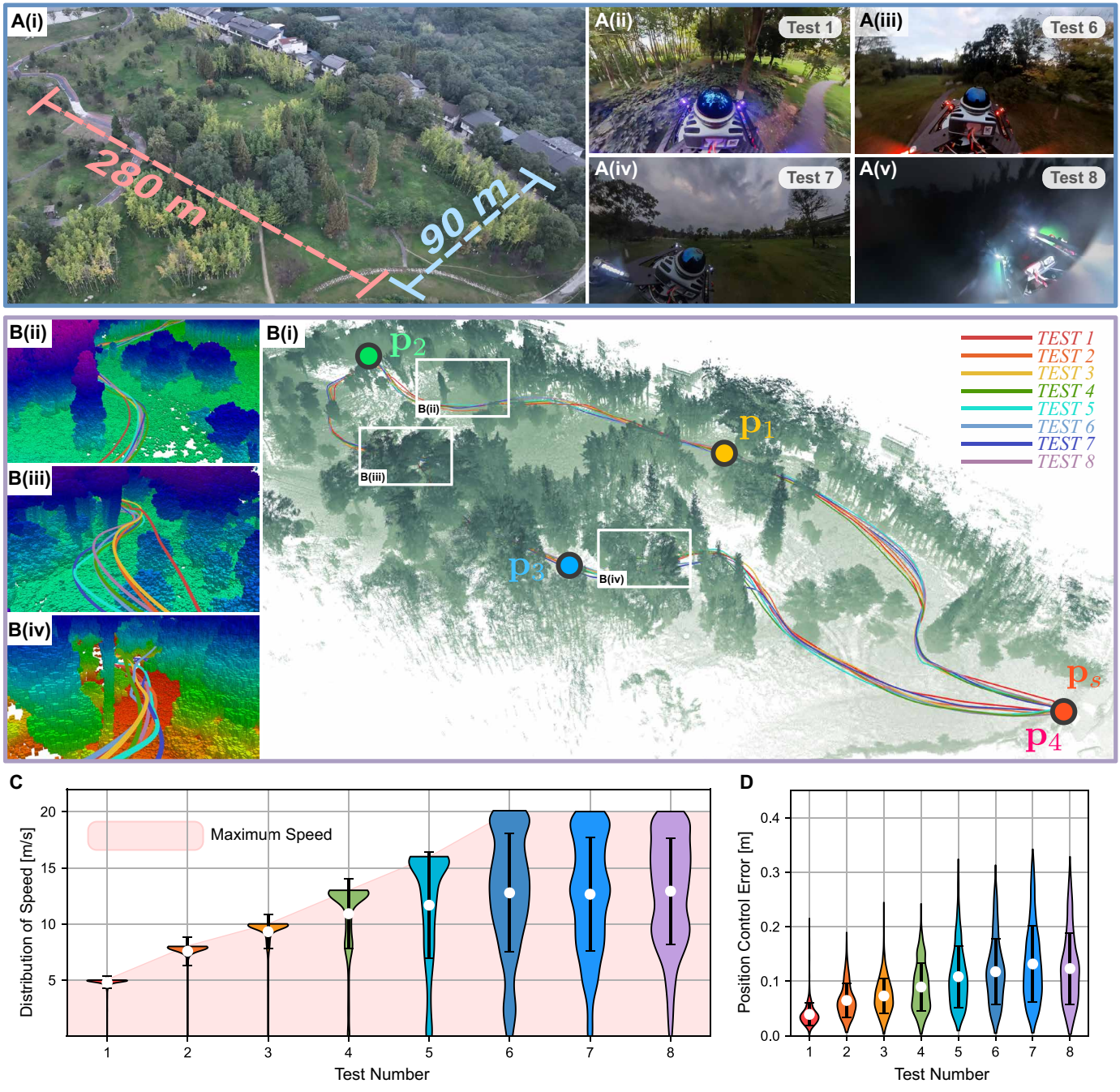

Fig. 3. Safe high-speed navigation in unknown environments. (**A**) Bird's-eye view of the experiment environment. (ii to v) Four representative lighting conditions from four of the eight experiment tests. Specifically, (ii) was captured during test 1, and (iii to v) were captured in tests 6 to 8, respectively. (**B**) (i) The executed trajectories during the eight tests are shown in curves with different colors. In all eight tests, the MAV flew through the waypoints $\mathbf{p}_s$, $\mathbf{p}_1$, $\mathbf{p}_2$, and $\mathbf{p}_3$ sequentially and stopped at position $\mathbf{p}_4$. Overlaid with the trajectories is the point cloud map, which is synthesized offline with point measurements collected during the flight. (ii to iv) Three detailed views of the environment on flight paths. (**C** and **D**) Violin plots illustrating the distribution of flight speed and position tracking error across the eight tests. The shape of the violin plot represents the density of the data distribution, with the white point indicating the mean value. The black error bar represents the SD. The data for these plots were obtained by uniformly sampling 2000 data points from the flight data of each test.

on these thin objects (Fig. 5D), and the point cloud map in our planning module effectively retains these points for trajectory planning. The robustness of SUPER when faced with small objects further strengthens its safety for real-world missions. A video illustration of this experiment is shown in movie S3.

Evaluation of safety, success rate, and efficiency

We evaluated the performance of SUPER's planning module through a series of controlled experiments conducted in simulation. The testing environments consisted of a set of randomly generated 3D forest-like scenarios at a fixed size of 110 m by 20 m (Fig. 6A). These

Fig. 4. Navigation in cluttered environments. (**A**) Trajectories of the target, the commercial drone, and SUPER during the target tracking experiment. After starting from position $\mathbf{p}_s^{target}$, the target passed sequentially through two wooded areas and ended at position $\mathbf{p}_g^{target}$. The commercial drone tracked the target closely until the target entered the first wooded area, where the commercial drone failed to find a collision-free path to track the object and stopped at position A. Once the target exited the first wooded area, the commercial drone resumed tracking by finding a path detouring the wooded area. However, the commercial drone failed the tracking mission again and stopped at position B when the target entered the second wooded area. In contrast, SUPER tracked the target well without stopping throughout the whole process. The point cloud map is synthesized offline with point measurements collected online during the flight of SUPER. (**B**) SUPER with extended size and the commercial drone (*27*). (**C**) Tracking of the target when she entered the first wooded area. The commercial drone failed to track the target because of the tight spaces, whereas SUPER could track the object closely. (**D**) Tracking of the target when she entered the second wooded area. The commercial drone failed the tracking again, whereas SUPER was able to fly through the tight space where the person being tracked had to lower her body to pass through.

scenarios encompassed six distinct obstacle densities ranging from 3.1 to 6.5 in terms of the traversability (*30*). Traversability is a metric describing the obstacle density normalized by the robot's size, specifically its maximum radius ($r = 0.2$ m in our simulation), where a higher traversability corresponds to a less challenging obstacle avoidance task. For each scenario, the positions and inclination angles of the obstacles were randomly generated. To ensure diversity, we used different random seeds to generate 10 different maps for each density, resulting in a total of 60 distinct maps for evaluation. We benchmarked the performance of SUPER with three state-of-the-art baseline planners: Bubble planner, an optimistic planning strategy (*13*); Raptor, a safety-aware method (*20*); and Faster, a safety-assured method (*23*). The implementation details of these four planners are summarized in table S1. Each planner was evaluated 18 times in each map, with maximum velocity varying from 1 to 18 m/s. The maximum acceleration was fixed at 20 m/s^2 for all experiments. In each experiment, a planner was given a goal 100 m away from the start position, and the experiment terminated at any of the three conditions. A "succeed" occurs when the MAV reaches the goal without any collisions and satisfies all kinematic constraints, including velocity and acceleration. A "collision" happens when the MAV collides with obstacles during the flight. An "unfinished" outcome is recorded if, in any replanning cycle before reaching the goal, the planner fails to return a trajectory within 30 s.

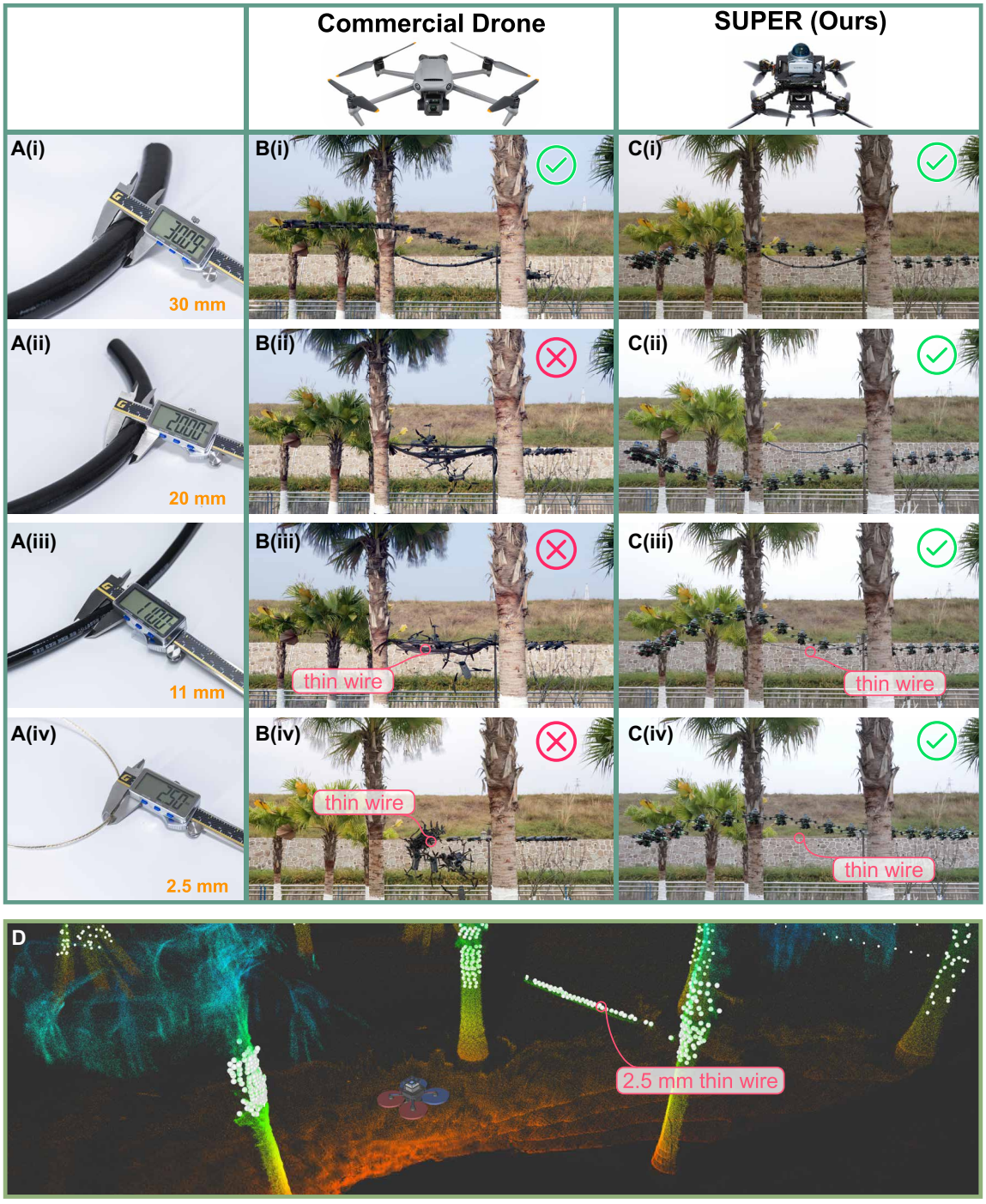

Fig. 5. Demonstration of thin object avoidance. (**A**) (i to iv) Four thin wires of varying diameters were used in the experiments. (**B**) (i to iv) Time-lapse images capturing the flights of DJI Mavic 3. It successfully avoided the thin wire of 30-mm diameter but failed to avoid wires with smaller diameters. The collision of the commercial drone caused the wire to swing, leading to some artifacts in the time-lapse images despite only one actual wire being present in each experiment. (**C**) (i to iv) Time-lapse images capturing the flights of SUPER. It successfully avoided all four types of thin wires. (**D**) Point cloud view of SUPER when facing a 2.5-mm thin wire. The wire was seen in the current scan measurements (white points) and more evident in the accumulated point cloud (colored points).

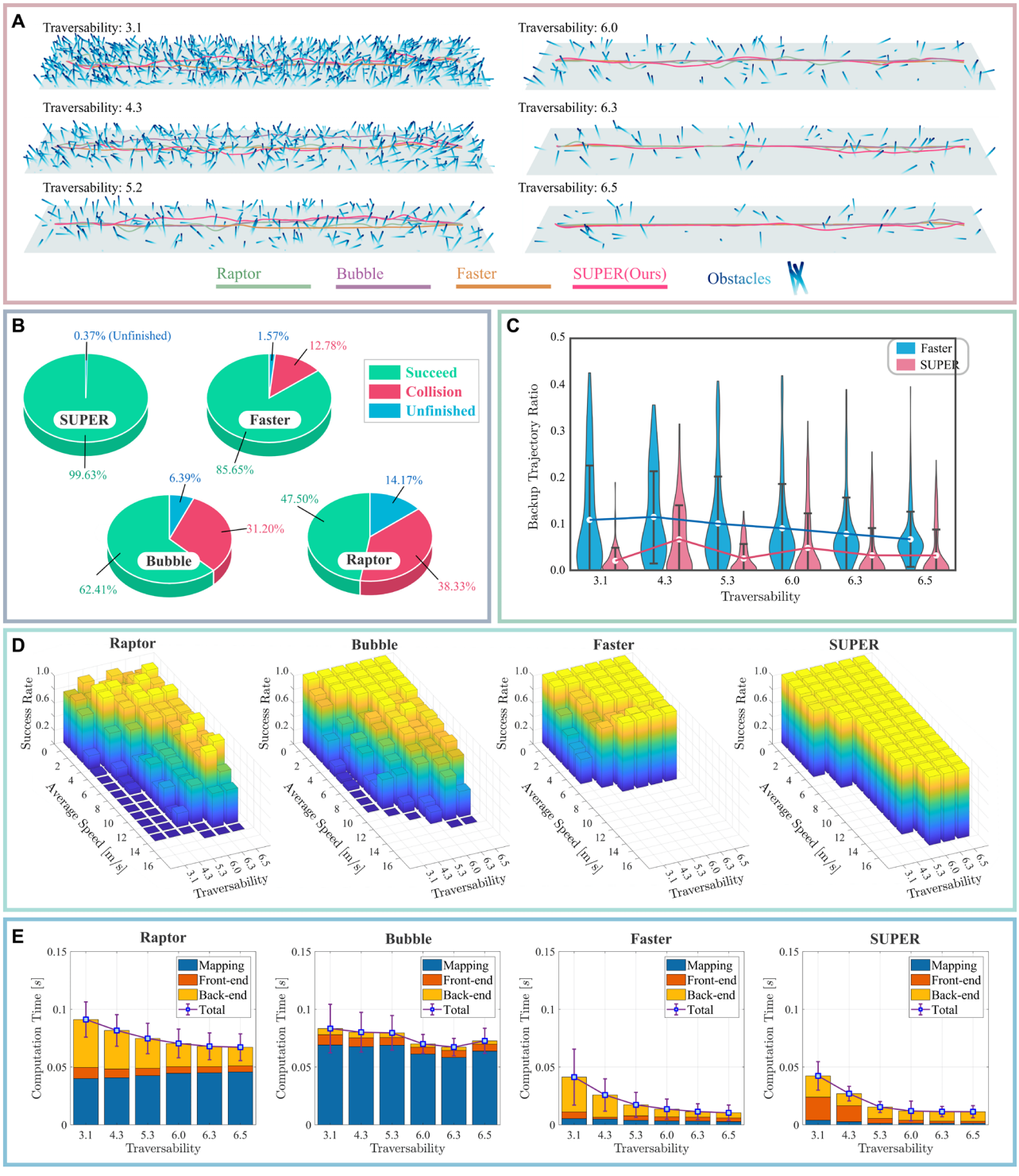

Fig. 6. Evaluation of safety, success rate, and efficiency. (**A**) Benchmarking environments with varying traversability. (**B**) Flight results of the benchmarked methods in 1080 experiments. (**C**) Distributions of the ratio of switching to backup trajectories for SUPER and Faster at varying obstacle densities, with each distribution computed from 180 tests. The white points represent the mean values, and the error bars indicate the SDs. (**D**) Success rate of the benchmarked methods at different obstacle densities and flight speeds. Empty columns indicate that the corresponding combination of speed and density was not achieved. (**E**) Time consumption of the benchmarked methods, with squares representing the mean values and error bars indicating the SDs of the total computation time. Each mean and SD was computed from 180 tests.

We evaluated the safety of all benchmarked methods in Fig. 6B in terms of the safe rate, which is the ratio of both succeed and unfinished cases over all experiments. Across all 1080 experiments with varying flight speed and obstacle density, SUPER had no collision or infeasible trajectory, achieving a perfect safe rate. Faster (*23*) is a safety-assured method in principle because of the planning of a backup trajectory in each replan cycle, similar to SUPER. However, we found that its backup trajectory was not constrained in the map region, where the unknown and occupied information is available, causing collisions when the map was not updated in time. The collision rate of Faster reached 12.78%, leading to an overall safe rate of 87.22%. Bubble (*13*) achieved a safe rate of 68.80%; the remaining 31.20% of collision cases were caused by its optimistic planning strategy that treats unknown spaces as free. Regarding Raptor (*20*), it suffered from a collision rate of 38.33%, which was primarily due to the planned trajectory frequently violating kinodynamic constraints during high-speed flights. These trajectories are challenging for the simulated MAV to track, resulting in collisions with obstacles. Consequently, the overall safe rate was merely 61.67%. To sum up, SUPER outperformed other benchmarked methods by achieving safe flights in all experiments, highlighting its exceptional safety-assured property.

Besides safety, whether the MAV achieved the task (for example, reaching the goal) and at which speed it achieved the task are also important. We analyzed the success rate, the ratio of succeed over all tests, and the flight speed of all benchmarked methods in Fig. 6D. SUPER achieved the highest average flight speed across almost all obstacle densities while maintaining a nearly perfect success rate of 99.63% (Fig. 6B). In contrast, Faster (*23*) had lower average speeds because of the short mapping range. Moreover, Faster formulated the trajectory optimization as an MIQP, which, on the one hand, had difficulties in finding high-speed trajectories that require intricate temporal optimization and, on the other hand, was time consuming (Fig. 6E). The high computation time limited the planning horizon of both exploratory and backup trajectories. Because both trajectories terminate at zero speed at each replan, the short planning horizon caused low-speed trajectories. Frequent timeouts in the MIQP also forced the MAV to execute backup trajectories (Fig. 6C) more often, which further reduced the overall flight speeds. Bubble (*13*), an optimistic planning strategy, and Raptor (*20*), a safety-aware planning method, had occasionally higher flight speeds than SUPER at certain obstacle densities but at the cost of a lower success rate. In sum, SUPER reduced the mission failure rate by 35.9 to 95.8 times compared with the other benchmarked methods while flying at higher or comparable speeds.

We analyzed the time consumption of all benchmarked methods as shown in Fig. 6E. Each method has its computation time broken into mapping and trajectory planning, which further breaks down into the front end and back end. SUPER and Bubble exhibited minimal time overhead for mapping (1 to 5 ms) because of their direct planning on point clouds, which can be obtained from LIDAR raw measurements with minimal processing, such as rigid transformation. Faster used an OGM, which has to enumerate a large number of map grids traversed by all points contained in a LIDAR scan, leading to a mapping time of up to 47 ms. Raptor used a Euclidean signed distance field (ESDF) map, which is built from OGM by further calculating the distance information of each grid (*31*), hence consuming an even longer time of 69 ms. For trajectory planning, SUPER, Bubble, and Raptor parameterized the trajectories by polynomials or B-splines, which are differentiable to the trajectory parameters and can be solved efficiently. In contrast, the MIQP problem solved in Faster for trajectory optimization is a nondifferentiable and hard to solve, taking a computation time of up to 41 ms. Consequently, Bubble and SUPER took the least total computation time (8 to 44 ms for Bubble and 10 to 47 ms for SUPER). The slightly higher time consumption of SUPER when compared with Bubble is due to the planning of an extra backup trajectory. Raptor took 67 to 83 ms, and Faster took 67 to 91 ms; both were higher than SUPER.

Applications

We demonstrated the use of SUPER in three representative applications: object tracking, autonomous exploration of unknown spaces, and waypoint navigation in changing environments (movie S4). In object tracking, we commanded SUPER to track an object by repeatedly assigning its local planning goal as a position behind the object. We conducted extensive experiments to evaluate SUPER's performance in tracking both a person and a car, as depicted in Fig. 2 (A to D). In all experiments, the target object was attached with a sheet of high reflectivity, enabling reliable detection within the LIDAR point cloud based on reflectivity measurements. The target was then tracked using a simple constant velocity mode. The results show that SUPER successfully tracked the target and avoided all obstacles in all experiments, regardless of variations in target speeds (ranging from 1 to 8 m/s), different indoor and outdoor environments, light conditions of daytime and nighttime, and different scene scales. In the application of autonomous exploration, the MAV was commanded to explore an unknown space. To fulfill this task, a global planner named Bubble Explorer (*32*) first generated a set of waypoints online. These waypoints were expected to observe the maximum unknown spaces at the minimum cost (such as the total flight distance). Then, SUPER was used as a local planner to reach these waypoints while ensuring obstacle avoidance. In the experiment, SUPER navigated to all of the waypoints without a collision failure and explored the entire 3000-m^2 environment within 120 s.

Figure 2 (E, i to iii), presents snapshots of the scene and the mapping results after the exploration. Last, we tested the flight capability of SUPER in changing environments (Fig. 2F). A typical problem for planning methods based on point clouds is that they falsely view spaces passed by moving objects as being occupied because of the points collected on the moving object. SUPER implemented a time-decaying point cloud map (Materials and Methods) and could hence avoid such a problem. In the experiment, the MAV starting from $\mathbf{p}_s$ was commanded to fly through a sequence of waypoints (from $\mathbf{p}_1$ to $\mathbf{p}_{11}$; see Fig. 2F, i); meanwhile, two people were moving randomly in the same area (Fig. 2F, ii). SUPER successfully arrived at all waypoints using the free spaces left by the moving objects without a collision with the moving object or static obstacles in the environments. These results underscored the superior navigation capabilities of SUPER in complex unknown environments.

DISCUSSION

Autonomous aerial robots

SUPER is designed for high-speed navigation in unknown environments by relying solely on its onboard sensing and computation. Agility and onboard computation are, therefore, two important factors. SUPER has a size of 280 mm and a TWR of 5.0, which characterizes its high agility. The onboard computer of SUPER is an Intel NUC 12 equipped with a 12-core 4.7-GHz central processing unit (CPU),

providing sufficient computation power for efficient, online optimization of both exploratory trajectories, attaining high-speed flights and backup trajectories and ensuring safety. The high agility and onboard computation power have enabled SUPER to achieve high-speed flights of greater than 20 m/s in unknown cluttered environments without a collision failure. The safety-assured and high-speed flying capability of SUPER places itself in a unique position among existing autonomous aerial robots (Fig. 1A). Agilicious (*33*) exhibits a high TWR of greater than 5.0 and a size similar to SUPER, but its imitation learning–based planning (*16*) achieves a lower flight speed of 10 m/s and lacks a safety guarantee, yielding a low success rate in high-speed flights (around 60% at 10 m/s). Moreover, the onboard computing unit, an Nvidia TX2 (*34*) with a four-core 1.9-GHz CPU, has limited computational power, which restricts the potential application of Agilicious for more computationally demanding tasks, such as autonomous exploration. On the other hand, ZJU Swarm (*35*) concentrates on designing small-size and lightweight autonomous MAVs for swarm applications, achieving an impressive size of a 110-mm wheelbase with a weight less than 300 g. However, this design has a low TWR (only 2.4) and limited onboard computation resources because of weight constraints, leading to a maximum speed of merely 3 m/s in autonomous flights (*35*). Being a representative commercial MAV (*27*), it has a larger wheelbase of 380 mm compared with SUPER, and its relatively low TWR (only 2.3) further restricts its ability to perform high-speed flights in cluttered environments. The commercial drone has an APAS, which can detect and avoid potential obstacles on the flight path. The maximum flight speed supported by APAS of the commercial drone is 15 m/s (*27*), which is still lower than the speed attained by SUPER in autonomous flights.

LIDAR-based MAV navigation

SUPER used a 3D LIDAR sensor to achieve its high-speed navigation. Compared with cameras that have been widely used in academic research (*14*–*16*, *18*–*21*, *23*) and commercial MAVs (*27*, *36*), LIDAR sensors provide key advantages that are crucial to enhance the flight speed, safety, and efficiency of MAV navigation. First, LIDAR sensors provide direct, accurate (centimeter-level), and long-range (tens to hundreds of meters) depth measurements. The long-range measurements allow the MAV to perceive and avoid obstacles far ahead, hence enabling a high-speed flight, and the accurate measurements allow the MAV to fly through small spaces in cluttered environments. In contrast, visual approaches estimate depths on the basis of disparity and triangulation (*37*), which is computationally demanding, and the accuracy decreases quadratically with the measuring distance. The state-of-the-art visual sensors, such as Intel RealSense (*38*), have an effective measuring range of merely 3 to 5 m. Second, LIDAR sensors measure 3D points at an extremely high frequency, from hundreds of thousands to millions of hertz. Using such high-frequency measurements would allow us to estimate extremely fast MAV motions (*39*), where traditional RGB or RGB-D cameras could suffer from severe motion blur because of the long exposure time and low dynamic range. Third, LIDAR sensors use TOF for depth measuring, enabling them to detect thin objects with diameters below even 1 cm. In contrast, visual-based (including RGB, RGB-D, and event camera–based) approaches prove inadequate in detecting typical thin objects like power lines because of the challenges of accurately computing disparities for such objects. Last, because of their active measuring mechanism, LIDAR sensors exhibit reliable performance even under low-light conditions, such as nighttime, surpassing visual sensors' capabilities in these scenarios.

Using 3D LIDAR sensors for MAV navigation is not new. For example, several works (*10*, *11*, *40*–*42*) used the VLP-16 on multirotor UAVs, and other works (*29*, *43*–*45*) used the Ouster OS1-128 (*46*). A crucial problem among these works lies in their LIDAR sensors. Existing LIDAR sensors are bulky, heavy, and costly: The prior two factors necessitate a UAV of a large size (wheelbase of 330 to 1133 mm) and low TWR [for example, typically around 2.0 (*11*, *29*, *40*, *41*, *43*–*45*, *47*, *48*)], which restrict the agility of the UAV. The high cost of LIDAR sensors has also prevented the wider adoption of them in UAV navigation, not even to speak of the high-speed agile flights that pose a much higher risk to the system. The reported flight speeds of existing LIDAR-based UAVs were merely 2 to 5 m/s (*40*, *41*, *44*). Zhang *et al.* (*10*, *11*, *42*) achieved a flight speed of 10 m/s with a LIDAR-based UAV, but their UAV was large (wheelbase of 1133 mm), heavy (more than 13 kg), and was only demonstrated in a static, sparsely wooded environment (for example, the distance between obstacles was more than 10 m) with very smooth flights. Our SUPER is enabled by the rapid recent advancements in LIDAR technology, which have enabled the commercialization and mass production of solid-state LIDARs with orders of magnitude–lower cost and weight. SUPER used a Livox MID360 LiDAR sensor (*25*), which features a weight of 265 g, size of 65 mm by 65 mm by 60 mm (length by width by height), and a power consumption of around 6.3 W. Although the LIDAR sensor is still larger and heavier and consumes more power than typical depth camera sensors [for example, the RealSense D435i (*49*) weighs only 72 g with a size of 90 mm by 25 mm by 25 mm and 1.6-W power consumption], given a compact design of airframe, SUPER has achieved a wheelbase of only 280 mm, which is even smaller than state-of-the-art visual-based MAVs such as the DJI Mavic 3 (380-mm wheelbase). Furthermore, SUPER achieves a TWR exceeding 5.0, surpassing the typical ratio of around 2.0 for existing LIDAR-based UAVs. The compact size and high TWR of SUPER lead to an agility that is seldom observed in existing works.

Although LIDAR sensors provide a superior sensing ability over depth sensors in terms of range, resolution, and accuracy, leveraging the LIDAR measurements to achieve safe, high-speed flights is still challenging, requiring a carefully designed map that truthfully represents the environments as measured by the LIDAR sensor. One mainstream approach in existing autonomous systems (*12*, *14*, *15*, *20*, *23*, *40*) uses OGMs (*50*, *51*) or ESDF maps (*31*). However, these maps are computationally demanding because of the ray-casting or distance field update processes. They are also incapable of mapping small thin objects because of the limited spatial resolution in the ray-casting process (see fig. S5). Another choice is the point cloud map (*10*, *11*, *13*, *41*, *42*, *52*), which is much more efficient to maintain and can effectively retain point measurements on small thin objects because of the avoidance of ray-casting. However, one inherent issue with a point cloud map is the difficulty in distinguishing known free space and unknown space. Therefore, existing point cloud–based planning methods (*10*, *11*, *13*, *41*, *42*, *52*) often treat the unknown area as free, which cannot ensure avoidance of occluded obstacles, and have exhibited low success rates in cluttered environments. We proposed a method (theorem 1 in Supplementary Methods) that distinguishes known free spaces directly from the point cloud map, allowing SUPER to plan a backup trajectory ensuring safe flights. Consequently, SUPER has achieved higher flight speeds and success rates than previous LIDAR-based navigation methods (*13*, *42*) (see Supplementary

Methods for a benchmark comparison). The accurate and high-resolution point map also enabled SUPER to avoid thin obstacles in the wild and navigate through tight spaces safely. Last, outdated points caused by moving objects or LIDAR false measurements present another problem for point cloud maps. We addressed this problem using a spatiotemporal sliding mechanism, which has enabled SUPER to use the areas traversed by moving objects.

Safety-assured high-speed trajectory planning

SUPER achieved high-speed and safe planning by calculating two trajectories at each time: an exploratory trajectory in both known free spaces and unknown spaces and a backup trajectory in known free spaces only. This two-trajectory strategy was first proposed in Faster (*23*, *28*). SUPER adopted such a two-trajectory strategy but had completely different designs for sensing, mapping, and planning, which are optimized for higher flight speeds, success rates, and computation efficiency. Ultimately, SUPER achieved flight speeds exceeding 20 m/s in the unknown wild using only its onboard sensing and computing devices.

From sensing, Faster (*23*) used a depth camera (*38*) for obstacle detection. The depth camera has a measuring range of merely 3 to 5 m, which limits the attainable flight speeds of the system. Moreover, Faster used an external motion capture system for state estimation, restricting its application in real-world environments that are often uninstrumented. In contrast, SUPER used a lightweight LIDAR sensor (*25*), which offers up to 70-m depth measurements, enabling detection and avoidance of obstacles at a distance. The long-range detection capability constituted the base for high-speed flights achieved in this work. SUPER further used the LIDAR measurements for state estimation, using our in-house developed LIDAR-inertial odometry, FAST-LIO2 (*39*). Consequently, the system relies completely on onboard devices and is adaptable to unknown environments without any prior instrumentation, such as Global Navigation Satellite System (GNSS)–denied environments.

A key requirement for the two-trajectory strategy used in Faster (*23*) and SUPER is to distinguish known free spaces of the environment. Faster (*23*) uses OGMs to maintain such information. However, updating OGM is rather computationally expensive because of the ray-casting operations (*50*). To constrain the overall computation time for real-time operation, Faster updates only 5 m of the OGM, attaining an average computation time of 47 ms. The limited mapping distance further constrains the planning horizon, leading to short trajectories that have rather low flight speeds. In contrast, SUPER distinguishes known free spaces directly on LIDAR point clouds (Materials and Methods), eliminating the time-consuming OGMs while retaining the full range of LIDAR points up to 70 m. The resultant map updating time is merely 1 to 5 ms.

For planning, Faster formulates the optimization of both exploratory and backup trajectories into MIQP problems. MIQP is nondifferentiable and often hard to solve, leading to a high computation time, occasional constraint violation, and local minimum trajectories with limited speeds. The high computation time, on the one hand, causes frequent timeouts that trigger the MAV to switch to backup trajectories of lower speeds and, on the other hand, limits the planning horizon that also constrains the flight speeds. In contrast, SUPER parameterizes the trajectory as multistage polynomials (Materials and Methods) following (*29*), in which spatiotemporal parameters could be efficiently optimized using gradient methods. SUPER's trajectory optimization required only half the computation time of Faster while adhering to all constraints. Another key difference between Faster and SUPER is the determination of switching time between backup and exploratory trajectories. Faster determines the switching time heuristically before the backup trajectory optimization on the basis of a wild assumption that the backup trajectory would decelerate along the exploratory trajectory. In contrast, SUPER optimizes the switching time along with the backup trajectory optimization. The optimized switching time in SUPER is usually much larger than the heuristically determined one in Faster, reducing the flight time on the backup trajectory and the chance of switching to it (fig. S6).

Limitations and future directions

At the hardware level, the emergence of smaller and lighter LIDAR sensors with longer sensing ranges and denser point clouds may open up new opportunities for autonomous aerial systems. Although the LIDAR adopted by SUPER is much smaller and lighter than previous ones, it remains larger and heavier than visual sensors, limiting the potential for further miniaturization of the MAV platform. This restricts the MAV's ability to operate in more confined spaces (such as narrow corridors with widths of less than 0.5 m). Adopting lighter LIDAR sensors could also enhance the TWR of the drone, thereby improving its agility further. In addition, the sensing range and point cloud density of LIDAR sensors can be further improved. Longer sensing ranges and denser point clouds can provide more detailed environmental information, which could enable SUPER to avoid thinner objects at higher speeds. The aerodynamic design of the drone can also be further optimized. The current SUPER system only considers aerodynamic drag effects at the software level by identifying drag coefficients and compensating it in the control module. However, with careful aerodynamics reducing air resistance, the drone's flight speed and stability could be increased even further.

At the algorithmic level, although SUPER has demonstrated the ability to handle dynamic obstacles, the current implementation does not actively detect moving objects or predict their movement, limiting its flight capability in highly dynamic environments. By actively identifying the moving objects (*53*) and predicting their motion, the planning module could generate smoother, safer trajectories, enhancing flight safety and reducing energy consumption. In addition, further parallelization of the SUPER software implementation and leveraging hardware accelerators, such as GPUs (graphics processing units), could enhance the system's efficiency, reduce latency, and further improve flight speed.

In the future, SUPER holds potential across a wide range of applications. Its key strengths lie in its emphasis on flight safety, high-speed capabilities, and relatively low computational demands. These features could progress various autonomous drone applications, including autonomous exploration, logistics transportation, infrastructure inspection, and search and rescue operations, where the ability to reach destinations safely and promptly is crucial. Another standout attribute of SUPER is its robustness, versatility, and scalability. It can operate in diverse, unstructured environments and is adaptable to all lighting conditions, enabling continuous, round-the-clock operation. From both hardware and software perspectives, SUPER effectively addresses the challenges of navigation in complex real-world environments, offering valuable insights for advancing autonomous robots from laboratory settings to real-world applications.

MATERIALS AND METHODS

Agile LIDAR-based MAV platform

The design objective of SUPER is to achieve a high TWR while minimizing its size. After a thorough investigation, we finalized the design of SUPER, as depicted in Fig. 7A. The platform has a takeoff weight of 1.5 kg and a compact wheelbase measuring only 280 mm. For the motors, we selected the T-Motor F90 type (*54*), with each motor capable of generating a maximum thrust of 23.6 N. This configuration results in a maximum total thrust of 94.4 N and an estimated ideal TWR of ~6.3. Because of factors like airflow interference and blockage, the actual TWR is slightly above 5.0, enabling highly agile maneuvers. For computation, the MAV is equipped with a flight control unit (FCU) running PX4 Autopilot (*55*) and an Intel NUC onboard computer that features a 12-core 4.7-GHz i7-1270P CPU. For sensing, the MAV equipped a lightweight 3D LIDAR (*25*) enabling object detection at ranges exceeding 70 m with a rapid point rate of 200,000 Hz. The MAV also equipped an IMU on the FCU, providing acceleration and angular rate measurements at a frequency of 200 Hz. The LIDAR point clouds and IMU measurements are input to the navigation software, which consists of a LIDAR-based perception module performing state estimation and obstacle mapping, a planning module for generating safety-assured and high-speed trajectories, and a controller to track the planned trajectories, all running on the onboard computer (Fig. 7). The outputs of the trajectory tracking controller are the desired angular velocity (consisting of pitch, roll, and yaw rate) and thrust acceleration, which are tracked by the lower-level controllers running on the FCU.

LIDAR-based perception

The perception module (Fig. 7B) consists of two main parts: state estimation and mapping. The state estimation part is based on our in-house designed LIDAR-inertial odometry, FAST-LIO2 (*39*), which provides low-latency nine–degrees of freedom state estimation (position, velocity, and attitude) at a frequency of 200 Hz and with a latency of less than 1 ms. It also produces world-frame registered point clouds at a scan rate of 50 Hz, with around 4000 points per scan, resulting in a total of about 200,000 points per second at a latency of less than 10 ms.

For the mapping module, we propose a point cloud–based spatiotemporal sliding point cloud map to represent the occupied spaces of the environment. To prevent increasingly densified points over time accumulation, we downsampled the points in the map using a uniform grid data structure. This structure divides the space into uniform grids with a resolution of 0.05 to 0.3 m and retains only the center point of each occupied grid cell. Also, to prevent overlarge maps that are not necessary for local planning, we only maintained a local map of 50 to 100 m around the MAV using a zero-copy map sliding strategy following our previous work (*51*). Furthermore, a well-known problem of a point cloud map is that LIDAR points belonging to both the static environment and moving objects in past locations are considered to be occupied straightly, which will falsely classify the space traveled by moving objects as occupied. To address this issue, a temporal sliding strategy was used to cull points outside a temporal window t_d. Specifically, within each grid cell, we tracked the time t_{hit} when it was last hit by a LIDAR point. If at current time t_c, the $\Delta t = t_c - t_{hit}$ was less than a threshold t_d, the grid cell was considered occupied; otherwise, it was considered free.

The point cloud map was constructed as follows. Upon the arrival of a new LIDAR scan, we located the grid cells hit by new points and updated their t_{hit} as the timestamp of the new point in it. On the other hand, to reduce the latency for map updates, we used a lazy-update strategy for the outdated map grids, which are culled only when they are queried by the trajectory planning module. The resultant computational complexity of map update depends solely on the number of LIDAR points in a scan while being independent of the sensing range. Consequently, the proposed mapping module can

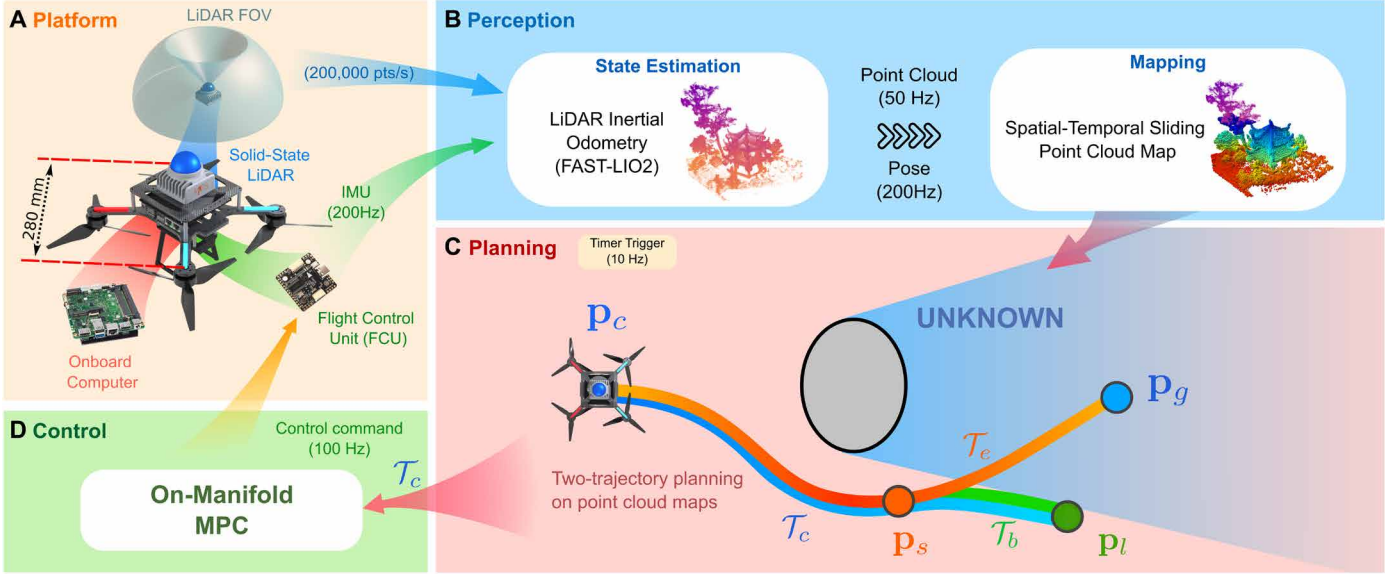

Fig. 7. System overview of SUPER. (**A**) Hardware configuration of SUPER. (**B**) Perception module comprises two components: state estimation and mapping. (**C**) The planning module is a safety-assured planner based on point cloud maps. The planned trajectory includes an exploratory trajectory $\mathcal{T}_e$, which spans both known free space and unknown space, and a backup trajectory $\mathcal{T}_b$, which starts from a state on $\mathcal{T}_e$ and lies entirely within known free space. Part of $\mathcal{T}_e$ and the entirety of $\mathcal{T}_b$ form the committed trajectory $\mathcal{T}_c$, which is then executed by the MAV. (**D**) The control module is an accurate and efficient OMMPC.

effectively leverage the extended sensing range of LIDAR to accomplish long-range updates. In contrast, volume mapping methods such as OGMs (*50*) or ESDF-based maps (*31*) have to update all grid cells in the measuring range, leading to a much higher computation load and short mapping range, typically around 5 to 10 m.

Safety-assured high-speed trajectory planning

Framework overview

To achieve safe and high-speed trajectory planning, we used a two-trajectory planning strategy where two trajectories are planned at each replan cycle (Fig. 7C). The first trajectory, called the exploratory trajectory $\mathcal{T}_e$: $\mathbf{p}_c \rightarrow \mathbf{p}_g$, which expands from the MAV current position $\mathbf{p}_c$ to a goal position $\mathbf{p}_g$. With the aim to achieve high-speed flight, the exploratory trajectory treats unknown space as free and is planned in both known free spaces and unknown spaces. The second trajectory, referred to as the backup trajectory $\mathcal{T}_b$: $\mathbf{p}_s \rightarrow \mathbf{p}_l$, begins from a position $\mathbf{p}_s$ on the exploratory trajectory at time t_s and ends at a position $\mathbf{p}_l$ with zero speed. With the aim to ensure safe flights, the backup trajectory and the segment of the exploratory trajectory from $\mathbf{p}_c$ to $\mathbf{p}_s$ must remain in known free spaces. In cases where the whole exploratory trajectory is already in the known free spaces (for example, when the MAV is in close proximity to the goal), no backup trajectory needs to be planned. Last, the committed trajectory $\mathcal{T}_c$: $\mathbf{p}_c \rightarrow \mathbf{p}_s \rightarrow \mathbf{p}_l$ consists of the exploratory trajectory from $\mathbf{p}_c$ to $\mathbf{p}_s$ and the backup trajectory from $\mathbf{p}_s$ to $\mathbf{p}_l$. The committed trajectory is then transmitted to the control module for execution.

The two trajectories above are replanned at a constant frequency of 10 Hz. At each replan, we determine the goal position $\mathbf{p}_g$ used for exploratory trajectory planning from the goal position $\mathbf{G}$ specified externally, such as the next waypoint to pursue in a waypoint navigation task and the next viewpoint in an exploration task. Specifically, if the distance between the current MAV position $\mathbf{p}_c$ and the goal position $\mathbf{G}$ exceeds the planning horizon H, we project $\mathbf{G}$ onto a sphere centered at $\mathbf{p}_c$ with a radius of H to obtain $\mathbf{p}_g$; otherwise, $\mathbf{p}_g$ is set to $\mathbf{G}$. A replan is considered successful if both the exploratory and backup trajectories are generated successfully or if an exploratory trajectory that remains entirely within known free spaces is generated. Upon a successful replan, a new committed trajectory is formed and updated to the control module. Otherwise, the replan is considered to have failed, and the MAV would execute the last committed trajectory. Given that the last committed trajectory remains in the known free space, executing it would ensure collision avoidance and guarantee flight safety, even at planning failure.

Both the exploratory and backup trajectories are planned via corridor-constrained trajectory optimization. The method consists of generating a flight corridor and then optimizing a smooth trajectory within the corridor (Fig. 8A). The flight corridor represents collision-free spaces from the start position to the goal position by a series of convex shapes, such as cubes (*56*), balls (*13*, *41*), and polyhedra (*40*, *23*, *12*). In our two-trajectory planning framework, two polyhedra corridors need to be generated: the exploratory corridor, which encompasses both known free spaces and unknown spaces, and the backup corridor, which encompasses only known free spaces. Then, the exploratory and backup trajectories are planned by optimizing two smooth trajectories in the respective flight corridors. In the following sections, we will present how to generate exploratory and backup corridors directly on LIDAR point clouds and how to optimize smooth trajectories within the respective corridors.

Flight corridor generation in configuration space

We represent both exploratory and backup corridors as a set of overlapping polyhedra and extract them directly from LIDAR point cloud inputs to enhance the computation efficiency. For the exploratory corridor, the objective is to find a set of overlapping polyhedra that connect the start points $\mathbf{p}_s$ to the goal position $\mathbf{p}_g$. To achieve this, we first use an A* path search algorithm (*57*) to find a collision-free path $\mathcal{G}$ connecting $\mathbf{p}_c$ and $\mathbf{p}_g$ (Fig. 8B, i); the resulting path consists of a set of discrete positions with a step equal to the map resolution (such as 0.1 m). Then, we search for a series of line segments along the path $\mathcal{G}$ by repeatedly finding the furthest visible point from the end of the last line segment. Specifically, starting from $\mathbf{p}_c$, the first point of $\mathcal{G}$, we find the furthest point $\mathbf{s}_1$ that is visible from $\mathbf{p}_c$ and form a line segment ($\mathbf{p}_c$, $\mathbf{s}_1$). Then, starting from $\mathbf{s}_1$, we find its furthest visible point $\mathbf{s}_2$ and form the line segment ($\mathbf{s}_1$, $\mathbf{s}_2$). This process is repeated until the goal position $\mathbf{p}_g$ is visible (Fig. 8B, ii). Last, each of the found line segments is used as a seed for convex decomposition, which extracts a polyhedron that contains the seed but not any obstacle points input to it (Fig. 8B, iii).

The convex decomposition has to be performed in the robot configuration space to consider the UAV size. Existing methods, such as the fast iterative regional inflation (FIRI) method (*58*), the regional inflation with line search method (*40*), or their predecessor the iterative regional inflation with semidefinite programming method (*59*), address robot size through point inflation or plane shrinkage. We introduce a new method for the convex decomposition in configuration space, termed configuration-space iterative regional inflation (CIRI). CIRI inherits the iterative optimization process and the efficient second-order cone program–based maximum inscribed ellipsoid volume algorithm from the prior work FIRI (*58*). It distinguishes itself from previous works (*59*, *40*, *58*) by directly operating on input point clouds and extracting polyhedra in the MAV configuration space without prior map inflation (*3*, *12*, *60*) or polyhedral shrinking (*40*, *61*). In addition, CIRI uses a plane adjustment strategy to ensure that the polyhedron generated in the configuration space also contains the line seed. Compared with point inflation and plane shrinkage strategies used in existing methods (*62*, *58*), the polyhedron obtained by CIRI achieves a larger volume in less computation time while ensuring a complete containment of line seed in the configuration space (see fig. S8). Details of the CIRI method are provided in the Supplementary Materials.

Backup corridor generation

Leveraging CIRI, we can efficiently generate exploratory corridors directly from LIDAR point clouds. However, generating backup corridors, which indicate known free spaces, from point clouds is challenging because of the lack of information regarding known and unknown areas. To overcome this limitation, we introduce a method that unifies the generation of exploratory and backup corridors within the same CIRI framework, relying solely on point cloud data and eliminating the need for ray cast–based occupancy mapping. Our approach leverages the dense and high-precision depth measurements obtained from LIDAR sensors and is fundamentally based on theorem 1 (see Supplementary Methods), which states that a polyhedron $\mathcal{S}$ extracted by CIRI will lie in the known free space if the input point cloud makes a sufficiently dense depth image and the input seed contains the current position $\mathbf{p}_c$. This is because $\mathcal{S}$ is convex, ensuring that it satisfies the convexity condition of theorem 1. It also does not contain any points from the input point cloud (or the depth image $\mathcal{D}$), fulfilling the requirement to exclude points

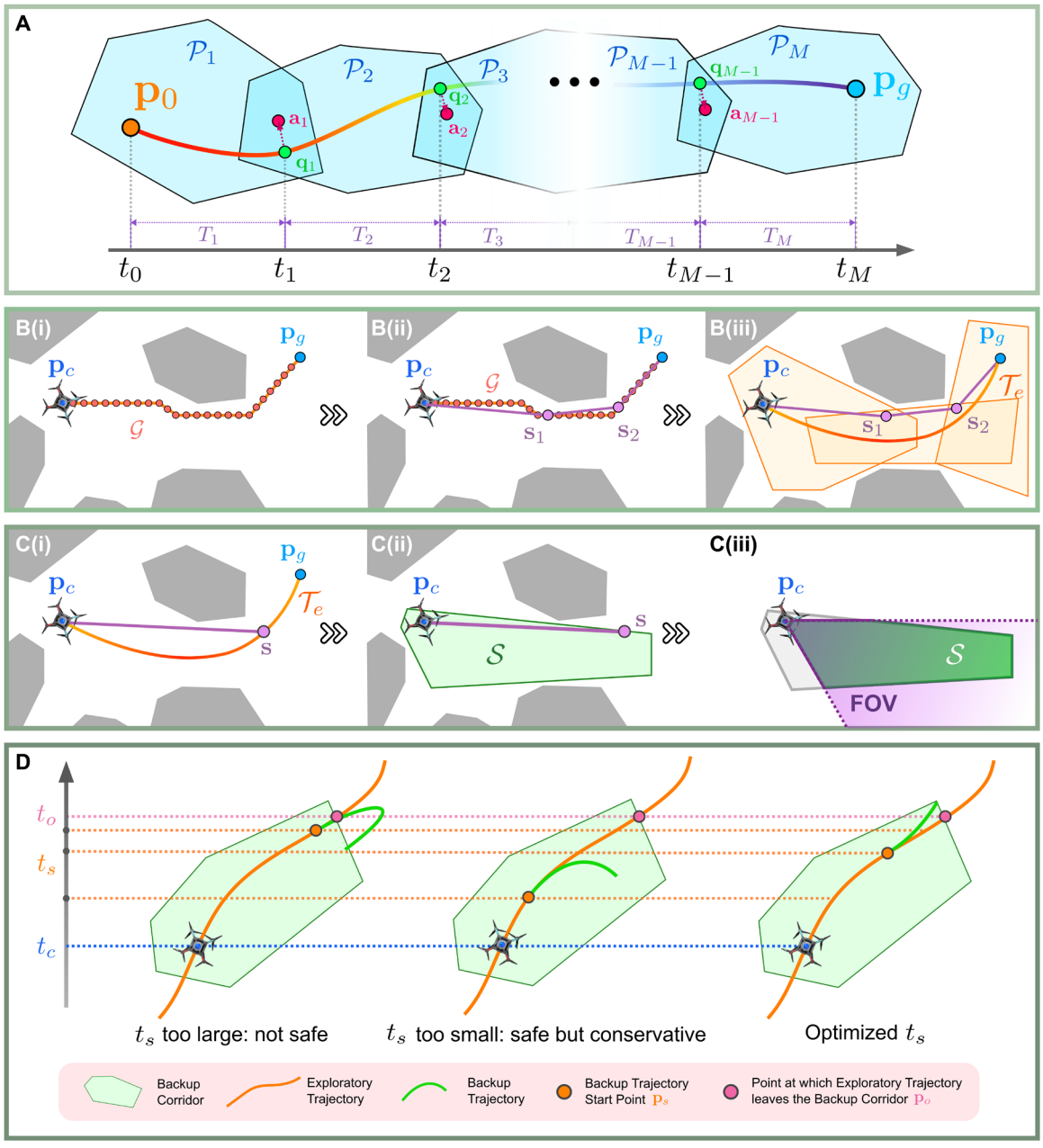

Fig. 8. Two-trajectory planning strategy. (**A**) Illustration of the corridor-based trajectory generation method that involves generating a corridor, represented by a series of overlapping polyhedra P_i, and then optimizing a smooth trajectory within the corridor. The trajectory, which begins at $\mathbf{p}_0$ and terminates at $\mathbf{p}_g$, is parameterized by a multistage polynomial trajectory through the intermediate position $\mathbf{q}_i$ connecting consecutive trajectory segments and the time duration T_i for each segment. $\mathbf{a}_i$ denotes the centers of the overlapping regions between two adjacent polyhedra. (**B**) The exploratory trajectory generation process includes (i) an A*-based path search, (ii) line seed generation, (iii) and convex decomposition followed by trajectory optimization. (**C**) The backup corridor generation process involves (i) searching for a seed on the exploratory trajectory, (ii) convex decomposition, and (iii) optionally cutting the corridor on the basis of the LIDAR field of view. (**D**) Effect of switching times t_s and the corresponding switching position $\mathbf{p}_s$ between the exploratory and backup trajectories. The switching time t_s is optimized within the constraint $t_c < t_s < t_o$.

from the map, and includes the seed, ensuring that the current position $\mathbf{p}_c$ is within the corridor as required. As a result, this polyhedron can serve as the backup corridor.

On the basis of theorem 1, the detailed process is illustrated in Fig. 8C. For the seed, we select the line segment that starts from the MAV's current position, $\mathbf{p}_c$, and ends at the furthest point $\mathbf{s}$ on the exploratory trajectory $\mathcal{T}_e$ that is visible from $\mathbf{p}_c$ (Fig. 8C, i). The purpose is that the backup corridor, which is extracted by the CIRI algorithm and should contain the seed ($\mathbf{p}_c$, $\mathbf{s}$), is aligned with the exploratory trajectory and can hence contain a substantial portion of the exploratory trajectory. Meanwhile, considering that practical LIDAR sensors are not infinitely dense, to make a sufficiently dense depth image, we accumulate recent LIDAR scans (for example, 1 to 2 s) and input them to CIRI, along with the line seed ($\mathbf{p}_c$, $\mathbf{s}$), to obtain a convex

polytope $\mathcal{S}$ (Fig. 8C, ii). The area within the generated polytope $\mathcal{S}$ is guaranteed to be known free according to theorem 1. If the LIDAR has a limited FOV, the intersection between the polytope $\mathcal{S}$ obtained by CIRI and a convex subset of the LIDAR FOV region can be used (Fig. 8C, iii). This intersection forms another polyhedron within the known free space. Moreover, the first polytope of the exploratory corridor must also be intersected with the MAV's FOV to ensure that the initial portion of the exploratory trajectory lies in known free space, thereby enhancing the feasibility of the backup trajectory optimization. A detailed method for selecting the convex subset of the FOV and an analysis of the FOV's influence are provided in the Supplementary Materials.

Trajectory optimization

Given the exploratory and backup corridors generated in the previous section, we consider how to optimize smooth trajectories within them. For exploratory trajectory generation, optimizing high-speed trajectories is challenging because it requires not only spatial optimization of the trajectory shape but also temporal optimization for time allocation. The objective is to find a smooth trajectory $\mathbf{p}(t)$: $[0, t_M] \to \mathbb{R}^3$ that minimizes the flight duration t_M and control efforts represented by the sth order derivative $||p^{(s)}||^2$ ($s = 4$ in this work) while satisfying all kinodynamic and collision avoidance constraints. The dynamic constraints encompass maximum speed and acceleration limitations, and collision avoidance is achieved through positional constraints enforced by flight corridors. Considering the composition of the flight corridors, we parameterize the trajectory as a multistage polynomial as proposed in (*29*). Specifically, given the exploratory corridor $\mathcal{C}_e$ composed of M consecutively overlapped polyhedra $\mathcal{P}_i$, both the trajectory $\mathbf{p}(t)$ and the flight duration t_M are divided into M segments, where the ith ($i = 1, 2, \cdots, M$) segment $\mathbf{p}_i(t)$ has a duration T_i and is contained in the polyhedra $\mathcal{P}_i$ (Fig. 8A). Two consecutive trajectory segments $\mathbf{p}_i(t)$ and $\mathbf{p}_{i+1}(t)$ connect at position $\mathbf{q}_i \in \mathbb{R}^3$ and are continuous up to the $(s-1)$th order derivative. Then, given the initial state $\mathbf{s}_0 = \mathbf{p}^{(0:s-1)}(0)$, which is the current MAV state, and the terminal state $\mathbf{s}_f = \mathbf{p}^{(0:s-1)}(t_M)$, which is the front-end path end position with zero derivatives, the ith polynomial trajectory can be parameterized by the intermediate point $\mathbf{Q} = (\mathbf{q}_1, \ldots, \mathbf{q}_{M-1})$ and time allocation $\mathbf{T} = (T_1, \ldots, T_M)$ as follows:

$$\mathbf{p}_i(t) = \mathbf{C}_i(\mathbf{Q}, \mathbf{T}, \mathbf{s}_0, \mathbf{s}_f)\beta(t), t \in [0, T_i], i = 1, 2, \cdots, M \quad (1)$$

where $\beta(t) = [1, t, \ldots, t^{2s-1}]^T$ is the basis function and $\mathbf{C}_i$ is the coefficient matrix of the polynomial that can be obtained from $\mathbf{Q}$ and $\mathbf{T}$ by solving a linear system (*29*). Moreover, $\mathbf{C}_i$ [and hence the trajectory $\mathbf{p}_i(t)$] is differentiable to both $\mathbf{Q}$ and $\mathbf{T}$ with the partial differentiation derived in (*29*).

With the parameterization in Eq. 1, the trajectory optimization is then formulated as

$$\min_{\mathbf{Q},\mathbf{T}} \int_0^{t_M} \left\|\mathbf{p}^{(s)}(t)\right\|_2^2 \mathrm{d}t + \rho_T t_M + \rho_u \mathcal{U} \quad (2)$$

$$\text{s. t.} \quad \mathbf{p}(t) = \mathbf{C}(\mathbf{Q}, \mathbf{T}, \mathbf{s}_0, \mathbf{s}_f)\beta(t) \quad (3)$$

$$t_M = \sum_{i=1}^{M} T_i \quad (4)$$

$$\|\dot{\mathbf{p}}\|_2^2 \le v_{\max}^2, \ \ \|\ddot{\mathbf{p}}\|_2^2 \le a_{\max}^2 \quad (5)$$

$$\mathbf{p}_i(t) \in \mathcal{P}_i, \forall i \in [1, M], t \in [0, T_i] \quad (6)$$

where $\rho_T > 0$ is the weight used to penalize the total flight time to achieve high flight speed and $\rho_u > 0$ is the penalty weight for the soft penalty term. Constraint Eq. 5 guarantees compliance with kinodynamic constraints, whereas constraint Eq. 6 constrains the position trajectory within the flight corridor. The term $\mathcal{U}$ in the objective function is defined as

$$\mathcal{U} = \sum_{i=1}^{M} \mathcal{L}_\mu\left(\left\|\mathbf{q}_i - \mathbf{a}_i\right\|_2^2\right) \quad (7)$$

where $\mathbf{a}_i$ is the center location of the overlapping area between $\mathcal{P}_i$ and $\mathcal{P}_{i+1}$. This term is used to prevent the trajectory from being too proximate to the boundaries of the flight corridor, which is often where the obstacles are located. Maintaining a distance from obstacles would enhance the visibility of the onboard sensor and hence allows generation of larger known free flight corridors for backup trajectory planning. $\mathcal{L}_\mu$ is a C^2 continuous barrier function as illustrated in fig. S9. We chose μ to be a larger value (such as 0.5) to make the penalty softer. This choice ensures that the intermediate point $\mathbf{q}_i$ is attracted toward $\mathbf{a}_i$ while allowing for some flexibility and avoiding excessive constraints on the optimization problem. To solve the optimization problem in Eq. 2, we adopted the method in (*29*), which reformulates the constrained problem into an unconstrained one. Because of the differentiability of the trajectory (and hence the objective and constraints in Eqs. 2 through 6) to the optimization variables $\mathbf{Q}$ and $\mathbf{T}$, the unconstrained optimization problem can be solved efficiently using a gradient-based method, the limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) solver (*63*). The solved exploratory trajectory is denoted as $\mathcal{T}_e(t)$.

The backup trajectory should start from an initial state $\mathbf{s}_0$ that lies on the exploratory trajectory $\mathcal{T}_e(t)$ at time t_s [where $\mathbf{s}_0 = \mathcal{T}_e(t_s)$] and end at a terminal position $\mathbf{p}_l$ in the known free space with zero speed and acceleration [where $\mathbf{s}_f = (\mathbf{p}_l, \mathbf{0}, \cdots)$]. To maximize the average flight speed, it is desirable to minimize the proportion of executing the backup trajectory because it is typically a deceleration trajectory due to the zero termination speed. That being said, the switching time t_s from the exploratory trajectory should be as late as possible so that the MAV can spend more time on the high-speed exploratory trajectory. In case of replanning failure or timeout, the late switching time also allows more attempts for the planning module to replan new trajectories to minimize the chance of executing the backup trajectory. On the other hand, t_s cannot be arbitrarily large. A t_s that is too large will cause the starting point of the backup trajectory $\mathbf{p}_s$ to be too close to the boundary of the backup corridor, which may render the backup trajectory optimization problem infeasible (Fig. 8D).

To determine the best switching time t_s, we optimized it along with the backup trajectory $\mathcal{T}_b(t)$. We adopted the same formulation as in the exploratory trajectory optimization (Eq. 2) but replaced the corridor constraints with the backup corridor $\mathcal{C}_b$ to ensure that the whole backup trajectory lies in the known free space. We also replaced the cost item $\mathcal{U}$ with $\mathcal{U} = -t_s$ to maximize the switching time t_s. In addition to modification of the objective function and

constraints, two additional optimization variables are added: the switching time t_s and the terminal position $\mathbf{p}_l$. The terminal position $\mathbf{p}_l$ replaces the goal position $\mathbf{p}_g$ in Eq. 2 and should be determined within the optimization because the MAV could rest at any position in the known free space. Moreover, because the backup trajectory lies within the backup corridor, the start point $\mathbf{p}_s$ should lie in the backup corridor as well, necessitating an additional constraint $t_c < t_s < t_o$, where t_c is the current time and t_o is the time when the exploratory trajectory exits the backup corridor, respectively (see Fig. 8D). To eliminate the inequality constraints in the optimization, we parameterize t_s as another variable $\eta \in \mathbb{R}$ as follows:

$$t_s = (t_o - t_c)\frac{1}{1+e^{-\eta}} + t_c \tag{8}$$

where η is completely unconstrained (see fig. S10). As a result, the complete optimization variables are ($\mathbf{Q}$, $\mathbf{T}$, $\mathbf{p}_l$, t_s). Combining the fact that the trajectory $\mathbf{p}_i(t)$ is differentiable to the $\mathbf{Q}$, $\mathbf{T}$, the initial state $\mathbf{s}_0$, and the terminal state $\mathbf{s}_f$ containing $\mathbf{p}_l$ [as proven in (*29*)] and the fact that $\mathbf{s}_0 = \mathcal{T}_e(t_s)$ is differentiable to t_s (and hence η), the whole optimization problem containing the objective and constraints is differentiable and can hence be solved by the gradient method, the L-BFGS solver (*63*), again.

On-manifold model predictive controller

The position trajectory $\mathbf{p}(t)$ generated by the planning module is then transmitted to the control module for tracking (Fig. 7D). The position trajectory is first transformed into a state trajectory $\mathbf{s}(t)$ on the basis of the differential flatness of the MAV subject to drag effects (*64*). Then, to accurately track this state trajectory at high speeds, we implement an on-manifold model predictive control (OMMPC) developed in our previous work (*65*). The method linearizes the quadrotor system by mapping its states from the state manifold to the local coordinates of each point along the trajectory $\mathbf{s}(t)$, leading to a linearized system that is minimally parameterized while avoiding kinematic singularities, allowing the MAV to perform aggressive maneuvers with large attitude angles, such as banked turns at sharp corners. The linearized system is a linear time-varying system, which is then controlled by a usual MPC. To enhance the control accuracy, we further consider the rotor drag compensation in the control. Detailed derivations of the OMMPC with rotor drag compensation and a quantitative analysis of the compensation effectiveness are provided in Supplementary Methods and fig. S13.

Statistical analysis

We used three types of statistical analyses: error bars, violin plots, and box plots. Error bars were used to represent the SD around the mean, reflecting the variability in the data. The SD σ was calculated as

$$\sigma = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \mu)^2} \tag{9}$$

where n is the number of data points, x_i is each data point, and μ is the mean value.

Violin plots, used in Figs. 3 and 6 and fig. S12, were used to illustrate the distribution of various performance metrics (e.g., flight speed, trajectory tracking error, and the ratio of backup trajectory execution) across multiple tests. These plots combine features of error bars and kernel density estimates, providing a depiction of the data's distribution, including its density and range. The kernel density estimate $\hat{f}(x)$ was calculated as

$$\hat{f}(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right) \tag{10}$$

where n is the number of data points, x_i is the data point, h is the bandwidth parameter, and K is the kernel function (a Gaussian kernel was used in our analysis). In each violin plot, the width corresponds to the density of data points at different values, and the central dot represents the mean value. The error bars, which indicate the SD, are shown alongside the mean to convey the data's variability.

Box plots, used in fig. S8, summarize the data distribution by displaying the median and interquartile range (IQR). The whiskers extend to 1.5 times the IQR, and data points beyond this range were considered outliers and are not shown in the plots.

Supplementary Materials

The PDF file includes:

Supplementary Methods
Figs. S1 to S13
Table S1
References (*66*–*75*)

Other Supplementary Material for this manuscript includes the following:

Movies S1 to S4

REFERENCES AND NOTES

1. J. Verbeke, J. D. Schutter, Experimental maneuverability and agility quantification for rotary unmanned aerial vehicle. *Int. J. Micro Air Veh.* **10**, 3–11 (2018).
2. B. Mishra, D. Garg, P. Narang, V. Mishra, Drone-surveillance for search and rescue in natural disaster. *Comput. Commun.* **156**, 1–10 (2020).
3. S. M. S. M. Daud, M. Y. P. M. Yusof, C. C. Heo, L. S. Khoo, M. K. C. Singh, M. S. Mahmood, H. Nawawi, Applications of drone in disaster management: A scoping review. *Sci. Justice* **62**, 30–42 (2022).
4. B. Rabta, C. Wankmüller, G. Reiner, A drone fleet model for last-mile distribution in disaster relief operations. *Int. J. Disaster Risk Reduct.* **28**, 107–112 (2018).
5. Y. Song, A. Romero, M. Müller, V. Koltun, D. Scaramuzza, Reaching the limit in autonomous racing: Optimal control versus reinforcement learning. *Sci. Robot.* **8**, eadg1462 (2023).
6. P. Foehn, A. Romero, D. Scaramuzza, Time-optimal planning for quadrotor waypoint flight. *Sci. Robot.* **6**, eabh1221 (2021).
7. A. Romero, R. Penicka, D. Scaramuzza, Time-optimal online replanning for agile quadrotor flight. *IEEE Robot. Autom. Lett.* **7**, 7730–7737 (2022).
8. A. Romero, S. Sun, P. Foehn, D. Scaramuzza, Model predictive contouring control for time-optimal quadrotor flight. *IEEE Trans. Robot.* **38**, 3340–3356 (2022).
9. E. Kaufmann, L. Bauersfeld, A. Loquercio, M. Müller, V. Koltun, D. Scaramuzza, Champion-level drone racing using deep reinforcement learning. *Nature* **620**, 982–987 (2023).
10. J. Zhang, R. G. Chadha, V. Velivela, S. Singh, P-cal: Pre-computed alternative lanes for aggressive aerial collision avoidance, paper presented at the 12th International Conference on Field and Service Robotics (FSR), Tokyo, Japan, 31 August 2019.
11. J. Zhang, R. G. Chadha, V. Velivela, S. Singh, P-cap: Pre-computed alternative paths to enable aggressive aerial maneuvers in cluttered environments, in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2018), pp. 8456–8463.
12. Y. Ren, S. Liang, F. Zhu, G. Lu, F. Zhang, Online whole-body motion planning for quadrotor using multi-resolution search, in *2023 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2023), pp. 1594–1600.
13. Y. Ren, F. Zhu, W. Liu, Z. Wang, Y. Lin, F. Gao, F. Zhang, Bubble planner: Planning highspeed smooth quadrotor trajectories using receding corridors, in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2022), pp. 6332–6339.
14. X. Zhou, Z. Wang, H. Ye, C. Xu, F. Gao, Ego-planner: An esdf-free gradient-based local planner for quadrotors. *IEEE Robot. Autom. Lett.* **6**, 478–485 (2020).

15. H. Ye, X. Zhou, Z. Wang, C. Xu, J. Chu, F. Gao, Tgk-planner: An efficient topology guided kinodynamic planner for autonomous quadrotors. *IEEE Robot. Autom. Lett.* **6**, 494–501 (2020).
16. A. Loquercio, E. Kaufmann, R. Ranftl, M. Müller, V. Koltun, D. Scaramuzza, Learning high-speed flight in the wild. *Sci. Robot.* **6**, eabg5810 (2021).
17. H. D. Escobar-Alvarez, N. Johnson, T. Hebble, K. Klingebiel, S. A. Quintero, J. Regenstein, N. A. Browning, R-advance: Rapid adaptive prediction for vision-based autonomous navigation, control, and evasion. *J. Field Robot.* **35**, 91–100 (2018).
18. L. Quan, Z. Zhang, X. Zhong, C. Xu, F. Gao, Eva-planner: Environmental adaptive quadrotor planning, in *2021 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2021), pp. 398–404.
19. L. Wang, Y. Guo, Speed adaptive robot trajectory generation based on derivative property of b-spline curve. *IEEE Robot. Autom. Lett.* **8**, 1905–1911 (2023).
20. B. Zhou, J. Pan, F. Gao, S. Shen, Raptor: Robust and perception-aware trajectory replanning for quadrotor fast flight. *IEEE Trans. Robot.* **37**, 1992–2009 (2021).
21. H. Oleynikova, Z. Taylor, R. Siegwart, J. Nieto, Safe local exploration for replanning in cluttered unknown environments for microaerial vehicles. *IEEE Robot. Autom. Lett.* **3**, 1474–1481 (2018).
22. S. Liu, M. Watterson, S. Tang, V. Kumar, High speed navigation for quadrotors with limited onboard sensing, in *2016 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2016), pp. 1484–1491.
23. J. Tordesillas, B. T. Lopez, M. Everett, J. P. How, Faster: Fast and safe trajectory planner for navigation in unknown environments. *IEEE Transact. Robot.* **38**, 922–938 (2021).
24. E. Mueggler, H. Rebecq, G. Gallego, T. Delbruck, D. Scaramuzza, The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and slam. *Int. J. Robot. Res.* **36**, 142–149 (2017).
25. Livox Technology Company Limited, Livox Mid-360 User Manual (2023); https://livoxtech.com/mid-360.
26. Wikipedia, Lockheed Martin F-35 Lightning II stealth multirole combat aircraft (2024); https://en.wikipedia.org/wiki/Lockheed_Martin_F-35_Lightning_II.
27. DJI, DJI Mavic 3 (2024); https://dji.com/mavic-3.
28. J. Tordesillas, B. T. Lopez, J. P. How, Faster: Fast and safe trajectory planner for flights in unknown environments, in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2019), pp. 1934–1940.
29. Z. Wang, X. Zhou, C. Xu, F. Gao, Geometrically constrained trajectory optimization for multicopters. *IEEE Trans. Robot.* **38**, 3259–3278 (2022).
30. C. Nous, R. Meertens, C. De Wagter, G. De Croon, Performance evaluation in obstacle avoidance, in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2016), pp. 3614–3619.
31. L. Han, F. Gao, B. Zhou, S. Shen, Fiesta: Fast incremental Euclidean distance fields for online motion planning of aerial robots, in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2019), pp. 4423–4430.
32. B. Tang, Y. Ren, F. Zhu, R. He, S. Liang, F. Kong, F. Zhang, Bubble explorer: Fast uav exploration in large-scale and cluttered 3D-environments using occlusion-free spheres. arXiv:2304.00852 [cs.RO] (2023).
33. P. Foehn, E. Kaufmann, A. Romero, R. Penicka, S. Sun, L. Bauersfeld, T. Laengle, G. Cioffi, Y. Song, A. Loquercio, D. Scaramuzza, Agilicious: Open-source and open-hardware agile quadrotor for vision-based flight. *Sci. Robot.* **7**, eabl6259 (2022).
34. NVIDIA Corporation, Nvidia Jetson TX2 embedded AI computing device (2024); https://developer.nvidia.com/embedded/jetson-tx2.
35. X. Zhou, X. Wen, Z. Wang, Y. Gao, H. Li, Q. Wang, T. Yang, H. Lu, Y. Cao, C. Xu, F. Gao, Swarm of micro flying robots in the wild. *Sci. Robot.* **7**, eabm5954 (2022).
36. Skydio 2 Plus Enterprise, An intelligent flying robot powered by AI (2024); https://skydio.com/skydio-2-plus-enterprise.
37. A. Harmat, M. Trentini, I. Sharf, Multi-camera tracking and mapping for unmanned aerial vehicles in unstructured environments. *J. Intell. Robot. Syst.* **78**, 291–317 (2015).
38. Intel, Intel RealSense Depth Camera D435i (2024); https://intelrealsense.com/depth-camera-d435i.
39. W. Xu, Y. Cai, D. He, J. Lin, F. Zhang, Fast-lio2: Fast direct lidar-inertial odometry. *IEEE Trans. Robot.* **38**, 2053–2073 (2022).
40. S. Liu, M. Watterson, K. Mohta, K. Sun, S. Bhattacharya, C. J. Taylor, V. Kumar, Planning dynamically feasible trajectories for quadrotors using safe flight corridors in 3-d complex environments. *IEEE Robot. Autom. Lett.* **2**, 1688–1695 (2017).
41. F. Gao, W. Wu, W. Gao, S. Shen, Flying on point clouds: Online trajectory generation and autonomous navigation for quadrotors in cluttered environments. *J. Field Robot.* **36**, 710–733 (2019).
42. J. Zhang, C. Hu, R. G. Chadha, S. Singh, Falco: Fast likelihood-based collision avoidance with extension to human-guided navigation. *J. Field Robot.* **37**, 1300–1313 (2020).
43. D. Cheng, F. C. Ojeda, A. Prabhu, X. Liu, A. Zhu, P. C. Green, R. Ehsani, P. Chaudhari, V. Kumar, TreeScope: An agricultural robotics dataset for lidar-based mapping of trees in forests and orchards, in *2024 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2024), pp. 14860–14866.
44. P. De Petris, H. Nguyen, M. Dharmadhikari, M. Kulkarni, N. Khedekar, F. Mascarich, sK. Alexis, Rmf-owl: A collision-tolerant flying robot for autonomous subterranean exploration, in *2022 International Conference on Unmanned Aircraft Systems (ICUAS)* (IEEE, 2022), pp. 536–543.
45. Y. Cai, F. Kong, Y. Ren, F. Zhu, J. Lin, F. Zhang, Occupancy grid mapping without raycasting for high-resolution LiDAR sensors. *IEEE Trans. Robot.* **40**, 172–192 (2023).
46. Ouster, Ouster OS1-128 sensor (2024); https://ouster.com/products/scanning-lidar/os1-sensor.
47. DJI, DJI Matrice 300 RTK (2024); https://enterprise.dji.com/zh-tw/matrice-300.
48. LiDAR USA, LiDAR USA surveyor 32 (2024); https://lidarusa.com/.
49. Intel, Product Family D400 Series Datasheet (2024); https://intelrealsense.com/download/21345/?tmstv=1697035582.
50. A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, W. Burgard, Octomap: An efficient probabilistic 3d mapping framework based on octrees. *Autonom. Robot.* **34**, 189–206 (2013).
51. Y. Ren, Y. Cai, F. Zhu, S. Liang, F. Zhang, Rog-map: An efficient robocentric occupancy grid map for large-scene and high-resolution lidar-based motion planning. arXiv:2302.14819 [cs.RO] (2023).
52. F. Kong, W. Xu, Y. Cai, F. Zhang, Avoiding dynamic small obstacles with onboard sensing and computation on aerial robots. *IEEE Robot. Autom. Lett.* **6**, 7869–7876 (2021).
53. H. Wu, Y. Li, W. Xu, F. Kong, F. Zhang, Moving event detection from LiDAR point streams. *Nat. Commun.* **15**, 345 (2024).
54. T-MOTOR, T-MOTOR F90 series racing drone motor specifications (2024); https://store.tmotor.com/goods-1064-F90.html.
55. Dronecode Foundation, PX4: Open source autopilot for drone developers (2024); https://px4.io.
56. J. Chen, K. Su, S. Shen, Real-time safe trajectory generation for quadrotor flight in cluttered environments, in *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)* (IEEE, 2015), pp. 1678–1685.
57. P. Hart, N. Nilsson, B. Raphael, A formal basis for the heuristic determination of minimum cost paths. *IEEE Trans. Syst. Sci. Cybern.* **4**, 100–107 (1968).
58. Q. Wang, Z. Wang, C. Xu, F. Gao, Fast iterative region inflation for computing large 2-d/3-d convex regions of obstacle-free space. arXiv:2403.02977 [cs.RO] (2024).
59. R. Deits, R. Tedrake, Computing large convex regions of obstacle-free space through semidefinite programming, in *Algorithmic Foundations of Robotics XI: Selected Contributions of the Eleventh International Workshop on the Algorithmic Foundations of Robotics* (Springer, 2015), pp. 109–124.
60. L. Yin, F. Zhu, Y. Ren, F. Kong, F. Zhang, Decentralized swarm trajectory generation for lidar-based aerial tracking in cluttered environments, in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2023), pp. 9285–9292.
61. J. Ji, N. Pan, C. Xu, F. Gao, Elastic tracker: A spatio-temporal trajectory planner for flexible aerial tracking, in *2022 International Conference on Robotics and Automation (ICRA)* (IEEE, 2022), pp. 47–53.
62. S. Liu, N. Atanasov, K. Mohta, V. Kumar, Search-based motion planning for quadrotors using linear quadratic minimum time control, in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2017), pp. 2872–2879.
63. ZJU FAST Lab, LBFGS-Lite: A header-only L-BFGS unconstrained optimizer (Github, 2024); https://github.com/ZJU-FAST-Lab/LBFGS-Lite.
64. M. Faessler, A. Franchi, D. Scaramuzza, Differential flatness of quadrotor dynamics subject to rotor drag for accurate tracking of high-speed trajectories. *IEEE Robot. Autom. Lett.* **3**, 620–626 (2017).
65. G. Lu, W. Xu, F. Zhang, On-manifold model predictive control for trajectory tracking on robotic systems. *IEEE Trans. Ind. Electron.* **70**, 9192–9202 (2022).
66. A. Koubaa, Ed., *Robot Operating System (ROS): The Complete Reference (Volume 1)*, vol. 625 of *Studies in Computational Intelligence* (Springer, 2017).
67. W. Liu, Y. Ren, F. Zhang, Integrated planning and control for quadrotor navigation in presence of suddenly appearing objects and disturbances. *IEEE Robot. Autom. Lett.* **9**, 899–906 (2023).
68. C. Toumieh, A. Lambert, Voxel-grid based convex decomposition of 3D space for safe corridor generation. *J. Intell. Robot. Syst.* **105**, 87 (2022).
69. X. Zhong, Y. Wu, D. Wang, Q. Wang, C. Xu, F. Gao, Generating large convex polytopes directly on point clouds. arXiv:2010.08744 [cs.RO] (2020).
70. C. D. Toth, J. O'Rourke, J. E. Goodman, *Handbook of Discrete and Computational Geometry* (CRC Press, 2017).
71. D. Eberly, Distance from a point to an ellipse, an ellipsoid, or a hyperellipsoid (Geometric Tools, 2013); https://geometrictools.com/Documentation/DistancePointEllipseEllipsoid.pdf.
72. Z. Wang, "A geometrical approach to multicopter motion planning," thesis, Zhejiang University, Hangzhou, China (2022).

73. Y. Cui, D. Sun, K.-C. Toh, On the r-superlinear convergence of the KKT residuals generated by the augmented lagrangian method for convex composite conic programming. *Math. Program.* **178**, 381–415 (2019).
74. W. Wu, mockamap, https://github.com/HKUST-Aerial-Robotics/mockamap.
75. Dronecode Foundation, PX4 software in the loop simulation with Gazebo (2024); https://docs.px4.io/v1.12/en/simulation/gazebo.html.

Acknowledgments
Funding: This work was supported by the Hong Kong Research Grants Council (RGC) General Research Fund (GRF) 17204523 and a DJI donation. **Author contributions:** The research was initiated by Y.R. and F. Zhang. Y.R. was responsible for the design and manufacture of the UAV prototype, as well as the implementation of the planning modules for the UAV, with assistance from G.L. and L.Y. F. Zhu and Y.C. were responsible for implementing the perception model, whereas G.L. and Y.R. implemented the control module with the assistance of N.C. The experiments were designed by Y.R. and F. Zhu and conducted by Y.R., F. Zhu, G.L., N.C., and F.K. Y.R. and J.L. performed all data analyses. The manuscript was written by Y.R. and F. Zhang, with input and advice from all authors. F. Zhang provided funding for the research and supervised the project. **Competing interests:** The authors declare that they have no competing interests. **Data and materials availability:** The data and source code supporting the conclusions of this study are available at https://doi.org/10.5281/zenodo.14528604 and https://github.com/hku-mars/SUPER.

Submitted 13 February 2024
Accepted 23 December 2024
Published 29 January 2025
10.1126/scirobotics.ado6187