https://doi.org/10.1038/s44182-024-00012-1

High-speed aerial grasping using a soft drone with onboard perception

Samuel Ubellacker[1] ✉, Aaron Ray[1], James M. Bern[2], Jared Strader[1] & Luca Carlone[1]

Contrary to the stunning feats observed in birds of prey, aerial manipulation and grasping with flying robots still lack versatility and agility. Conventional approaches using rigid manipulators require precise positioning and are subject to large reaction forces at grasp, which limit performance at high speeds. The few reported examples of high-speed aerial grasping rely on motion capture systems, or fail to generalize across environments and grasp targets. We describe the first example of a soft aerial manipulator equipped with a fully onboard perception pipeline, capable of robustly localizing and grasping visually and morphologically varied objects. The proposed system features a novel passively closed tendon-actuated soft gripper that enables fast closure at grasp, while compensating for position errors, complying to the target-object morphology, and dampening reaction forces. The system includes an onboard perception pipeline that combines a neural-network-based semantic keypoint detector, a state-of-the-art robust 3D object pose estimator, and a fixed-lag smoother to estimate the pose of known objects. The resulting pose estimate is passed to a minimum-snap trajectory planner, tracked by an adaptive controller that fully compensates for the added mass of the grasped object. Finally, a finite-element-based controller determines optimal gripper configurations for grasping. Experiments on three different targets confirm that our approach enables dynamic, high-speed, and versatile grasping, all of which are necessary capabilities for tasks such as rapid package delivery or emergency relief. We demonstrate fully onboard vision-based grasps of a variety of objects, in both indoor and outdoor environments, and up to speeds of 2.0 m/s—the fastest vision-based grasp reported in the literature. Finally, we take a major step in expanding the utility of our platform beyond stationary targets, by demonstrating motion-capture-based grasps of targets moving up to 0.3 m/s, with relative speeds up to 1.5 m/s.

Current quadrotor platforms can fly at high speeds, maneuver with great agility, and carry a wide range of payloads, but their ability to *interact* with the world remains limited. Most quadrotor platforms sense their environment for applications such as navigation, inspection, or videography[1–4]. However, many tasks require interacting with the environment, and quadrotors built to accomplish such tasks must handle complicated time-varying contact dynamics and have the ability to perceive task-relevant features of the environment. Recent work has increasingly focused in this direction, including a number of perching mechanisms[5–8], and full systems for applications such as contact-based inspection[9], and tree eDNA collection[10]. These tasks require contact with the environment, but not necessarily changing the environment. Our work enables quadrotors to take a more active role in interacting with the world around them by picking up and moving objects without the need for external motion capture infrastructure, expanding the kinds of tasks that can be accomplished by such platforms.

Building an aerial grasping system requires solving problems at the intersection of mechanical design, perception, motion planning, control, and manipulation. There is extensive prior work toward solving each of these problems individually. Many works focus on gripping or grasping mechanisms for attaching to conventional drones[5–7,11,12] or helicopters[13]. Recent work has examined a closer coupling between the drone and gripper design[5,14]. The control of aerial manipulators has been studied in[15–18], and recently for related tasks such as catching balls in flight[19] and grabbing other drones out of the air[20]. Vision-based object pose estimation and tracking are widely studied problems in robotics, and many existing methods are relevant for estimating the target's position for aerial manipulation; notable examples applied to aerial grasping include[21]. While each of these works

[1]Massachusetts Institute of Technology, Cambridge, MA, USA. [2]Williams College, Williamstown, MA, USA. ✉e-mail: subella@mit.edu

reports promising advances in the subsystems that aerial grasping platforms are built upon, they do not directly seek to achieve high-speed vision-based aerial manipulation.

Combining perception, control, and manipulation to achieve high-speed and versatile aerial manipulation presents several challenges. First, conventional approaches using rigid manipulators require precise positioning and are subject to large reaction forces at grasp, which limit performance at high speeds. Second, in contrast to systems that passively sense the environment, a quadrotor manipulating objects in the environment must adapt to changing dynamics, in particular since the added mass of the grasped object might be non-negligible compared to the mass of the drone itself. Third, if such systems are to be generally useful, they must also function without perfect state information from external motion capture systems; therefore they must rely on onboard perception and their perception system has to be resilient to the manipulator partially occluding the onboard camera. These problems must be solved in real-time, to enable efficient operation, and with the limited onboard computation, to circumvent the need for an external infrastructure.

The few reported examples of high-speed aerial grasping rely on motion capture systems, or fail to generalize across environments or grasp targets. For instance, Thomas et al.[15] demonstrate high-speed grasping of up to 3 m/s. But, the system relies on a motion capture system for both the state of the drone and target and the manipulator is specifically designed for grasping a cylindrical object weighing only 27 g. They use a rigid manipulator and ignore the risk of contact interaction with the ground by suspending the target in the air. Fishman et al.[22] demonstrate dynamic grasping with a soft aerial gripper, but the system requires external localization for the quadrotor and target. The system was only shown to grasp one type of object and with a grasp speed of only 0.2 m/s. Several works take strides towards vision-based aerial manipulation but reduce errors by using motion-capture for drone localization, flying at slow speeds, and/or making strong assumptions that limit the types of objects that can be grasped. Bauer et al.[23] use a segmentation network and point clouds to determine the pose of arbitrary targets, but perform the computations using an offboard computer. Furthermore, the state of the drone relies on a motion capture system. Ramon-Soria et al.[24] employ a multi-link rigid gripper and a CAD model of the object to precisely identify grasp points. As a result, the system can only execute very careful, stationary grasps. Lin et al.[25] are capable of performing onboard vision-based grasps in outdoor environments, but also employs a rigid gripper which requires careful gripper positioning and is limited to stationary grasps. Other works make strong assumptions that the target must be cylindrical, rely on motion capture for the drone state, and are unable to grasp with any notable speed[26,27]. None of these works combines vision-based drone localization and target pose estimation, the ability to pick up targets with meaningful forward velocity, and fully onboard computation.

We posit that aerial grasping can be improved by focusing on *soft* grasping mechanisms rather than trying to achieve extremely precise positioning with a rigid manipulator. We draw inspiration from biological systems (e.g., birds), which utilize a combination of rigid and soft tissues to achieve unparalleled agility and robustness[28]. Soft robotic grippers can passively conform to the grasped object, reduce the need for explicit grasp analysis, and weaken the coupling of flight and manipulation dynamics; this is an example of *morphological computation*, i.e., the exploitation of passive mechanical elements to supplement explicit control[29]. In this work, we combine a rigid quadrotor platform with a soft robotic gripper (Fig. 1A, B). Rigid and soft robots can operate together synergistically[30] as hybrid systems that get the best of worlds: our system combines the speed and agility of a rigid quadcopter with the natural compliance and robustness of a soft robotic gripper, and harnesses control methodologies that enable its rigid and soft components to work together.

Contribution. Our first contribution is to develop a *soft, quadrotor-mounted gripper* with passively closed and tendon-actuated foam fingers. Our design enables fast closure at grasp, while compensating for positions errors, complying to the target-object morphology, and dampening reaction forces. Our second contribution is to develop a real-time and fully onboard perception pipeline for drone and target state estimation, using two cameras and an onboard GPU (Fig. 1C, D). Our pipeline requires prior knowledge of the target's geometry and visual features. The pipeline combines a neural-network-based semantic keypoint detector with a state-of-the-art method for robust 3D object pose estimation and a fixed-lag smoother. Finally, we integrate a minimum-snap grasp trajectory planner with an adaptive controller for the drone and a finite-element-based approach for the soft gripper control. The adaptive controller is able to adjust to quickly changing dynamics, caused by the grasped object and related aerodynamic effects (e.g., ground effect and down-wash resulting from the grasped object); the finite-element-based approach computes optimal configurations for the soft gripper that are key for high-speed grasping.

We evaluate our soft drone with onboard perception when grasping three different objects in both indoor and outdoor environments. We present extensive experimental results on these objects across 180 flight tests. In contrast to prior work, our system performs all computation on board, can fly with or without a motion capture system, and can perform high-speed grasping of static targets at up to 2.0 m/s. Moreover, we show that when operating within a motion capture system the soft drone can also grasp objects from *moving* platforms: a quadruped robot carrying a med-kit moving forward at 0.3 m/s, and a rotating turntable with a relative grasp speed of 1.5 m/s.

Results

In this section, we provide an overview of our system (Section "System overview") and then discuss the results of our flights with both static (Section "Grasping static targets") and moving targets (Section "Grasping moving targets"). More technical details are postponed to the Materials and methods section, and visualizations of the system and experiments are shown in Movie 1.

System overview

A successful object grasp requires the drone to detect and estimate the object's pose, plan a feasible grasping trajectory, and track that trajectory while facing complicated contact dynamics and other external disturbances.

Our system (Fig. 2) is designed to perform fully onboard perception, using a RealSense D455 color and depth camera for object pose estimation and a RealSense T265 stereo fisheye camera for visual-inertial odometry (VIO) for drone state estimation[31]. After estimating the target's and drone's pose in the global frame, (While some aerial grasping approaches based on visual servoing avoid the need for an explicit world-frame target estimate[26,27], we form a global estimate of the target's pose to remove the constraint that the target must always be visible in the camera's field of view, which is a restricting assumption in particular right before grasping.) the quadrotor plans a minimum-snap polynomial trajectory[32] connecting its initial hover point, a grasp point directly above the target, and a terminal hover point. This reference trajectory is then tracked by an adaptive flight control law[33] which learns to compensate for the additional mass after grasp and for unmodeled aerodynamic effects. The gripper's finger positions are controlled along the trajectory using a finite-element-based approach[28,34] governed by an objective functions that balances target object visibility and grasp robustness. The object pose estimation and trajectory planning algorithms run on a Jetson Xavier NX, the low-level flight control runs on a Pixhawk micro-controller, and the visual-inertial odometry pipeline runs on the RealSense T265 camera module.

Below, we provide an overview of our soft gripper design and the onboard perception system, while we postpone the details to Materials and methods.

Soft gripper design and modeling. Eliminating all estimation and tracking errors from the perception and control systems is infeasible, so the grasp mechanism must be robust to imprecise positioning to enable reliable grasps. Collisions between the fingers and the ground present a particular challenge. Small amounts of error in the planned or executed

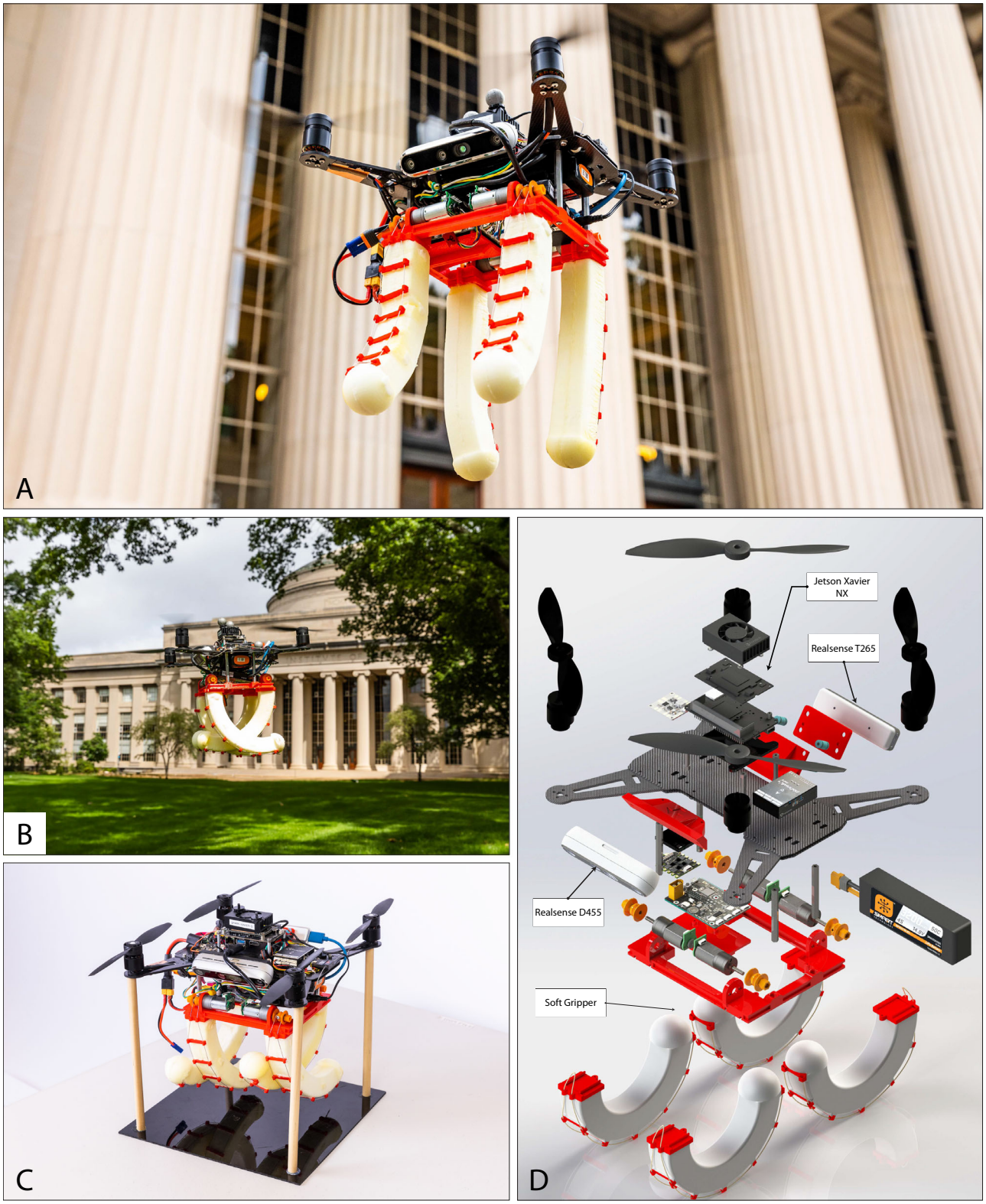


Fig. 1 | Overview of the Soft Drone. A, **B** The proposed Soft Drone platform flying outdoors. **C** Front view of the Soft Drone platform, with the soft gripper in the passively closed state. **D** Exploded CAD view of the quadrotor and gripper, including onboard sensing and computation.

trajectory lead to the manipulator contacting the ground. With a traditional rigid manipulator, these unexpected contact forces induce a moment on the quadrotor that results in it rapidly pitching forward, leading to a crash or failed grasp[22].

We propose to solve this issue with *compliant* fingers for picking up the object (Fig. 3A). Such fingers weaken the coupling between ground contact and drone dynamics, enabling steadier flight in the event of ground contact (Movie 1). Soft fingers also provide robustness for grasping the object: they conform to the object morphology and do not require precisely chosen grasp points.

Our finger design is made of foam molded in a closed position. Soft foams, especially castable expanding polyurethane, have been recently

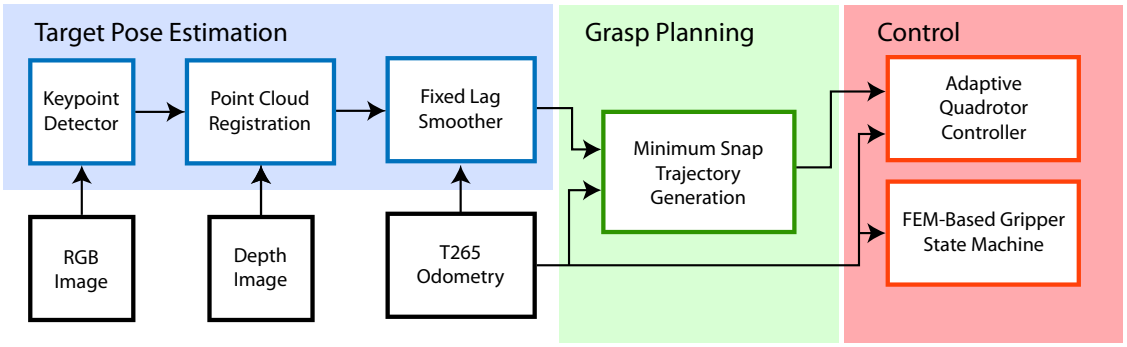


Fig. 2 | Grasping pipeline overview. An accurate estimate of the target and drone's pose is first identified and used to plan the polynomial grasp trajectory. This trajectory is tracked with an adaptive quadrotor controller, and the soft gripper states are dictated by an FEM-based optimization.

harnessed to produce highly-deformable soft robots with favorable strain-stress ratios[35]. A cable connected via eyelets along the outside edge of the finger controls its shape (Fig. 3B). As the motor at the base of the finger turns, the cable shortens and pulls the finger into an open position. When the motor turns in the opposite direction, the cable loosens and the foam's elasticity drives the finger back to the closed position. This passively closed finger design has the advantage that the closing speed is limited only by the stiffness of the finger and no-load speed of the motor, in addition to lower power draw when the fingers are closed. The resulting finger design is mechanically quite simple compared to more traditional rigid finger designs that require actuation at each joint. In comparison to our previous soft finger design[22] that required two motors working in synchronization to open and close each finger, our current design has half the number of motors and is much easier to fabricate and control. Now each finger only has a single degree of freedom, yet it can assume nontrivial shapes (Fig. 3C).

The positioning of the fingers is important for a successful grasp. We model the body of the gripper as a finite element mesh, and the cables as unilateral springs running through via points in the mesh. Given a feasible choice of real-world control inputs, e.g., motor angles $\boldsymbol{u}$, this approach to soft robotic modeling can accurately predict the real finger's deformed shape. This is done by minimizing the total energy of the system, to find a statically stable deformed mesh position, following the approach in[28,34]:

$$\boldsymbol{x}(\boldsymbol{u}) = \arg\min_{\boldsymbol{x}} E(\boldsymbol{u}, \boldsymbol{x}). \tag{1}$$

In addition to helping us predict the gripper's motion, the FEM-based model can be harnessed in a nested optimization to do control. In this work, we find optimal control inputs by minimizing a suitable objective function:

$$\boldsymbol{u}^* = \arg\min_{\boldsymbol{u}} \left(\mathcal{O}_{\text{grasp}}(\boldsymbol{x}(\boldsymbol{u})) + \mathcal{O}_{\text{FOV}}(\boldsymbol{x}(\boldsymbol{u})) \right), \tag{2}$$

where the objective is designed to produce control inputs that deform the gripper into a shape $\boldsymbol{x}(\boldsymbol{u}^*)$ that maximizes the area enclosed by the fingertips and the target's centroid (as quantified by the term $\mathcal{O}_{\text{grasp}}(\boldsymbol{x}(\boldsymbol{u}))$ in Eq. (2), see Materials and methods for a complete mathematical expression) while also not occluding the quadcopter's field of view (as quantified by the term $\mathcal{O}_{\text{FOV}}(\boldsymbol{x}(\boldsymbol{u}))$). The exact choice of objective depends on the different phases of the grasp, as discussed below.

When the drone is planning the grasp trajectory (Fig. 5A-1), the front fingers should not be in the front camera's field of view or they may obscure the object (Fig. 3D, Fig. S7). The back fingers also need to stay out of the navigation camera's field of view, to avoid interfering with the drone localization system. We refer to this configuration as *target observation state* (Fig. 5B-1), and set $\mathcal{O}_{\text{FOV}}(\boldsymbol{x}(\boldsymbol{u})) = \mathcal{O}_{\text{FOV}_R}(\boldsymbol{x}(\boldsymbol{u})) + \mathcal{O}_{\text{FOV}_F}(\boldsymbol{x}(\boldsymbol{u}))$ as the sum of the FOV constraints for the rear (T265) and front (D455) cameras.

Right before grasping the target (what we call the *pre-grasp state*, Fig. 5B-2), we can remove the front camera FOV constraint because we have global knowledge of the target's location. Therefore, $\mathcal{O}_{\text{FOV}}(\boldsymbol{x}(\boldsymbol{u})) = \mathcal{O}_{\text{FOV}_R}(\boldsymbol{x}(\boldsymbol{u}))$, and the front fingers can reach the unconstrained grasp configuration, which maximizes the area between the gripper's fingertips and the target's centroid, greatly improving our position error margins (Fig. 3E). We note that this objective function also results in the rear fingers pointing downward; if the fingers are closed too late, the target will still be caught up by the back fingers and a successful grasp may still be possible (Fig. 3A). Our objective function depends on the position of the target relative to the drone. However, computational constraints prevent us from performing this optimization in real-time. Instead, we perform an offline optimization that substitutes the true relative position between the grasp target and the drone with a constant, tuned offset. The offset is tuned to approximate the actual relative position as the drone enters the pre-grasp state and is about to grasp. This is a reasonable simplification, as all trajectory shapes are locally similar near the grasp target, and the soft gripper compensates for any induced generalization inaccuracies.

To initiate the grasp closure, the tendons are simply commanded back to their rest positions and the fingers return to the passively closed state, thus enclosing the target (Figs. 3F, 5B-3).

Perception System Design. Our perception system enables global position and orientation estimation of the target object (Fig. 4). It is important to estimate the target's orientation in addition to position, as some objects are much more difficult to pick up along certain axes (e.g., the two-liter bottle in Fig. 4B). The perception pipeline assumes that the shape (more precisely, the CAD model) of the objects to be picked up is known ahead of time, but the framework is generalizable to arbitrary objects.

A deep-neural-network-based keypoint detector can be rapidly trained with an automated data collection tool (Fig. 4A) and predicts a set of keypoints in each color image collected by the onboard Realsense 455 camera (Fig. 4C); each semantic keypoint corresponds to a specific 3D point on the object CAD model. The detected 2D points are mapped to 3D keypoints based on the image's depth channel. The object's pose can be then estimated by aligning the detected keypoints to the corresponding points in the known object CAD model (Fig. 4B). As there may be outliers in the semantic keypoint detection process, we employ a robust registration algorithm, namely TEASER++[36], to find the target pose. Combining the camera-relative object pose with the quadrotor's own odometry estimate gives a global pose estimate for the target object (Fig. 4C). In our implementation, we use the visual-inertial odometry computed by the T265 RealSense camera as the quadrotor's state estimate.

Figure 4D–F shows the pixel-space, translation, and rotation errors for the pose estimates produced by out perception pipeline for a med-kit on a test dataset held out from training. As the distance to the target increases, the translation estimate degrades while the keypoint and rotation errors remain roughly constant, indicating that our range is mostly limited by the depth camera's accuracy. Following the analysis in Supplementary Text S1, we note that the maximum error our pose detection can incur while still planning successful grasps depends on the target's geometry and speed at grasp, and report an estimated maximum planning distance for each object

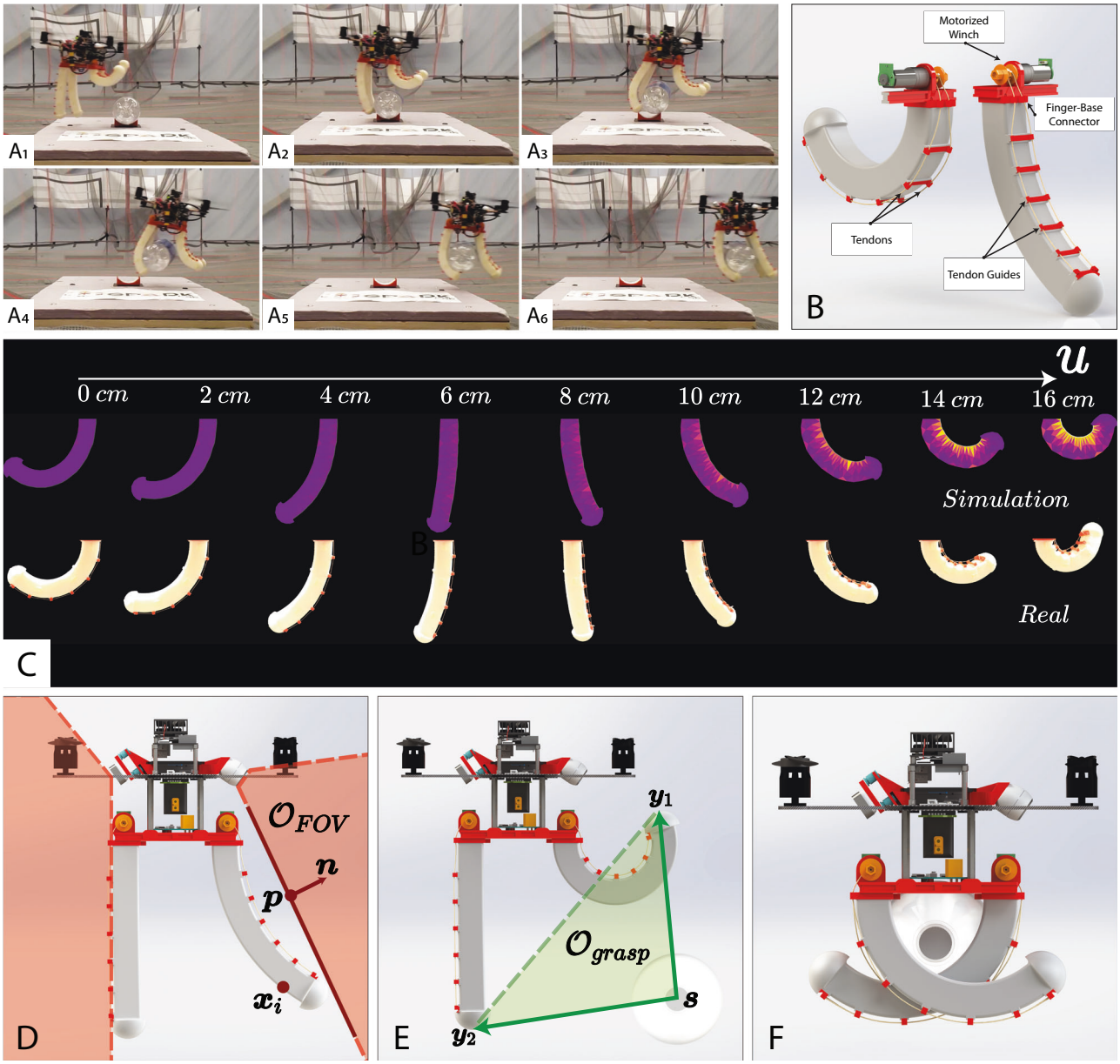


Fig. 3 | FEM modeling and objective functions. A Still frames from a grasp with over 2 m/s forward velocity. The compliance of the soft fingers mitigates the high reaction forces faced at this speed ($\mathbf{A_3}$) and the gripper configuration reduces the sensitivity of the gripper closing timing. **B** Soft finger in its passively closed state (left) and partially open state (right). The finger opening is actuated by a motorized winch contracting a tendon fixed to the tip of the finger and routed through guides. **C** The finger shape predicted by the FEM model based on cable retraction length u (top) matches the actual finger shape for the same cable length (bottom). **D** Target observation state: the field-of-view objective, $\mathcal{O}_{\text{FOV}}$, drives the mesh nodes, including the node with position $\mathbf{x}_i$, out of the half space bounded by the plane defined by point $\boldsymbol{p}$ and normal $\boldsymbol{n}$ and corresponding to the cameras' frustum. **E** Pre-grasp state: the grasp margin objective, $\mathcal{O}_{\text{grasp}}$, maximizes the area enclosed by the grippers fingertips $\mathbf{y}_1$ and $\mathbf{y}_2$ and the target point $\mathbf{s}$. **F** Gripper closed configuration: the fingers return to their passively closed state.

at 0.5 m/s in Fig. S8. In our flight tests, we select approximately 1.25 m as our nominal planning distance for all experiments. Additionally, these raw global pose estimates are refined using a fixed-lag smoother to further improve accuracy; see Materials and methods for details.

Our system is capable of estimating the full 3D rotation of the target, which is necessary to plan grasps for targets on inclines or for determining the feasibility of a grasp. However, for most grasping applications, the target can be assumed to rest on a flat horizontal plane. Thus, only the target's yaw is used to plan trajectories.

Grasping static targets

We conducted grasps of three distinct stationary objects (a med-kit, a two-liter bottle, and a cardboard box, Fig. 5C–E) to demonstrate the versatility of the proposed soft gripper and vision system and quantify the effects of various sources of error. In our tests, upon takeoff, the drone flies to a randomized starting point from which it can see the target, as shown in Fig. 5A. Once the perception system has an estimate of the target's pose, it flies to a point such that the forward direction of the drone is aligned with a predefined grasp direction for the target and the drone is 0.9 m in front of and above the target. (Fig. 5A-1, B-1). From that starting point, the quadrotor flies a trajectory through the grasp point (Fig. 5A-2, B-2) that terminates at an arbitrary endpoint away from the target (Fig. 5A-3, B-3). The drone stops updating its estimate of the target pose once it has started the grasp trajectory, relying on its initial estimate of target pose and updated estimate of its own state in the global frame in order to fly to the grasp point.

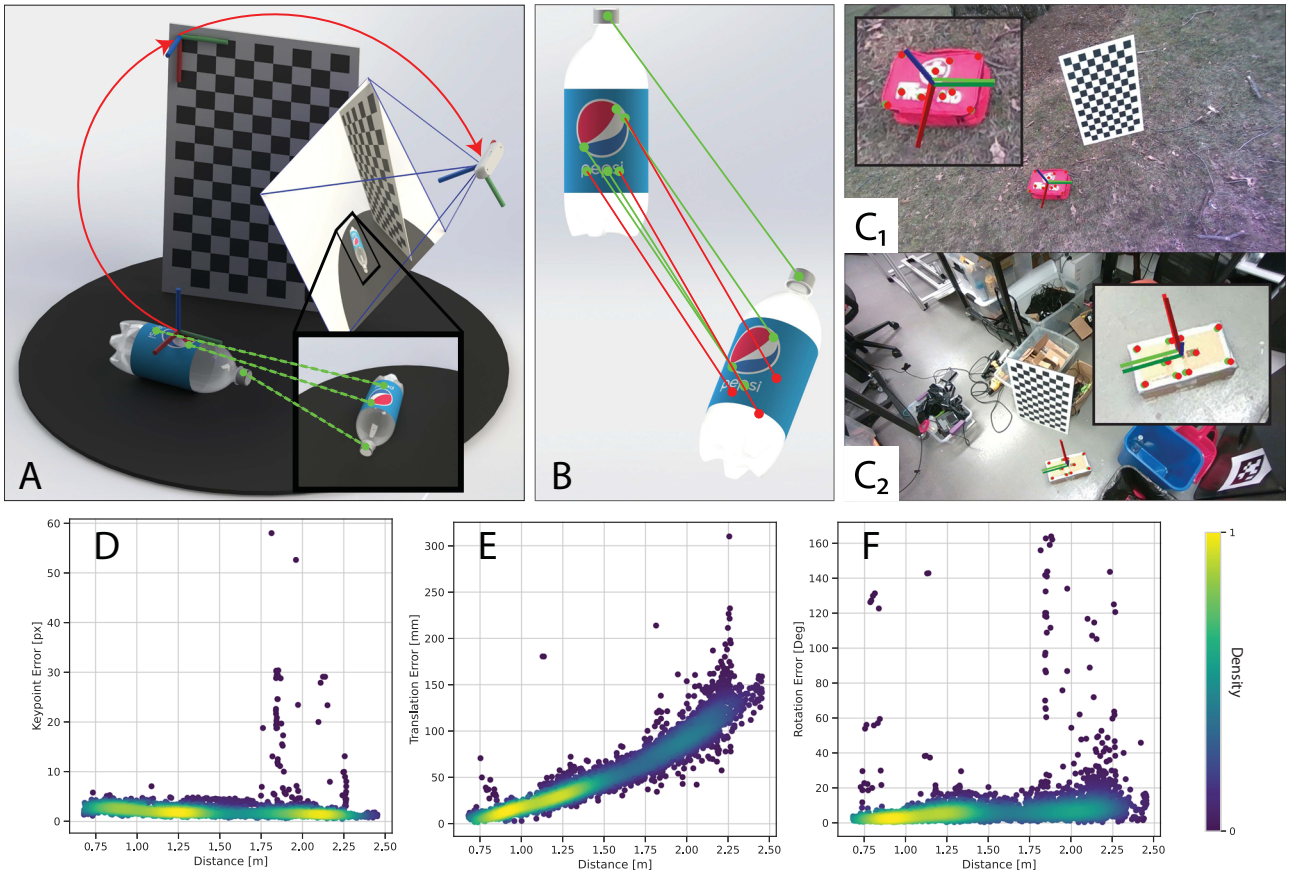


Fig. 4 | Keypoint-based object pose estimation. A Keypoints are labeled to generate training data for the semantic keypoint detector, for each object of interest. A checkerboard in the background of the collection area enables calculation of camera pose, which accelerates and partially automates the labeling process. A Resnet-18 neural network[43] is trained to predict object keypoints based on this dataset. **B** The 3D keypoints predicted by the perception system are matched against the corresponding keypoint locations on the object CAD model. A rigid transformation computed between the observed and CAD keypoints gives an estimate of the relative pose between the object and the camera. **C** The keypoint and pose detection system generalizes across environments and targets. The red points are predicted keypoints, the green points are ground-truth keypoints, the dark colored axes indicate the estimated pose, and the light colored axes denote the ground-truth pose. **D–F** Error characteristics for the proposed perception system, showing pixel-space, translation, and rotation error depending on distance between the target object and the camera for a med-kit object.

Grasp success rate. In this first set of tests, the reference trajectories were chosen to have a forward velocity of 0.5 m/s at the grasp point. A grasp is considered a success if the target remains grasped until the drone lands. For all tests, the target is secured to a horizontal surface with weak magnets to prevent the downdraft of the quadrotor displacing the target pre-grasp. Under these conditions, the system achieves a success rate of 9/10, 6/10, and 10/10 for the med-kit, cardboard box, and two-liter bottle, respectively, and with fully onboard vision (Fig. 6A). To determine the extent that our perception system affects grasp success, we repeated these experiments with a motion capture system that provides accurate states for both the drone and target. The motion capture baseline performs similarly to the vision-based approach (Fig. 6A), slightly underperforming for the med-kit, improving for the cardboard box, and performing equally for the bottle. Our results show that our vision system is able to consistently match the motion capture results at this speed, indicating that the gripper mechanism is robust against the additional error caused by the perception pipeline.

The biggest contributors to grasp performance difference among objects are their mass and surface morphology. The two-liter bottle's cylindrical geometry and low mass (60 g) enable consistent enveloping grasps. The uneven, roughly concave edges of the med-kit provide good grasp surfaces, but the high mass (148 g) makes sustaining the grasp more difficult. The cardboard box's tall, straight walls combined with its relatively high mass (115 g) force the gripper to rely on large pinching force to secure the grasp, making it the most difficult target to grasp.

In isolated tests, our gripper can hold up to a 2 kg target when the grasp is completely enveloping. For grasps relying on pinching force, the maximum payload decreases significantly; we observe the gripper failing consistently to hold the cardboard box when its mass is increased to 250 g. We note that the material properties also play a role here, where it becomes harder to maintain a secure, pinching grasp on targets with a low frictional coefficient.

The target's dimensions are another key factor in the success rate. We provide a detailed analysis on how the target's geometry affects grasp success rate in Supplementary Text S1.

Given a secure grasp connection with a target, there is still the possibility the drone cannot exert enough thrust to successfully carry the target. The target's mass is a key factor here, but the target's surface area about the top face is also an important consideration. Targets with high surface area occlude more of the drone's thrust envelope, leading to reduced efficiency. In our experiments, we determined a target with mass 250 g (13% of the drone's total weight) and a top plane surface area of 546 cm^2 was the limit of our system. For reference, the surface areas of the two-liter bottle, med-kit, and cardboard box are 310, 411, and 319 cm^2, respectively.

Error analysis. We collected ground-truth pose data for the drone and target with a motion-capture system in order to understand the relative contribution of errors from the target estimation pipeline, the VIO estimate, and the trajectory tracking to the system's performance.

The refined pose estimate for each target (Fig. 6B) at the start point of the trajectory has at most 5 cm translation error and less than 10 degrees of

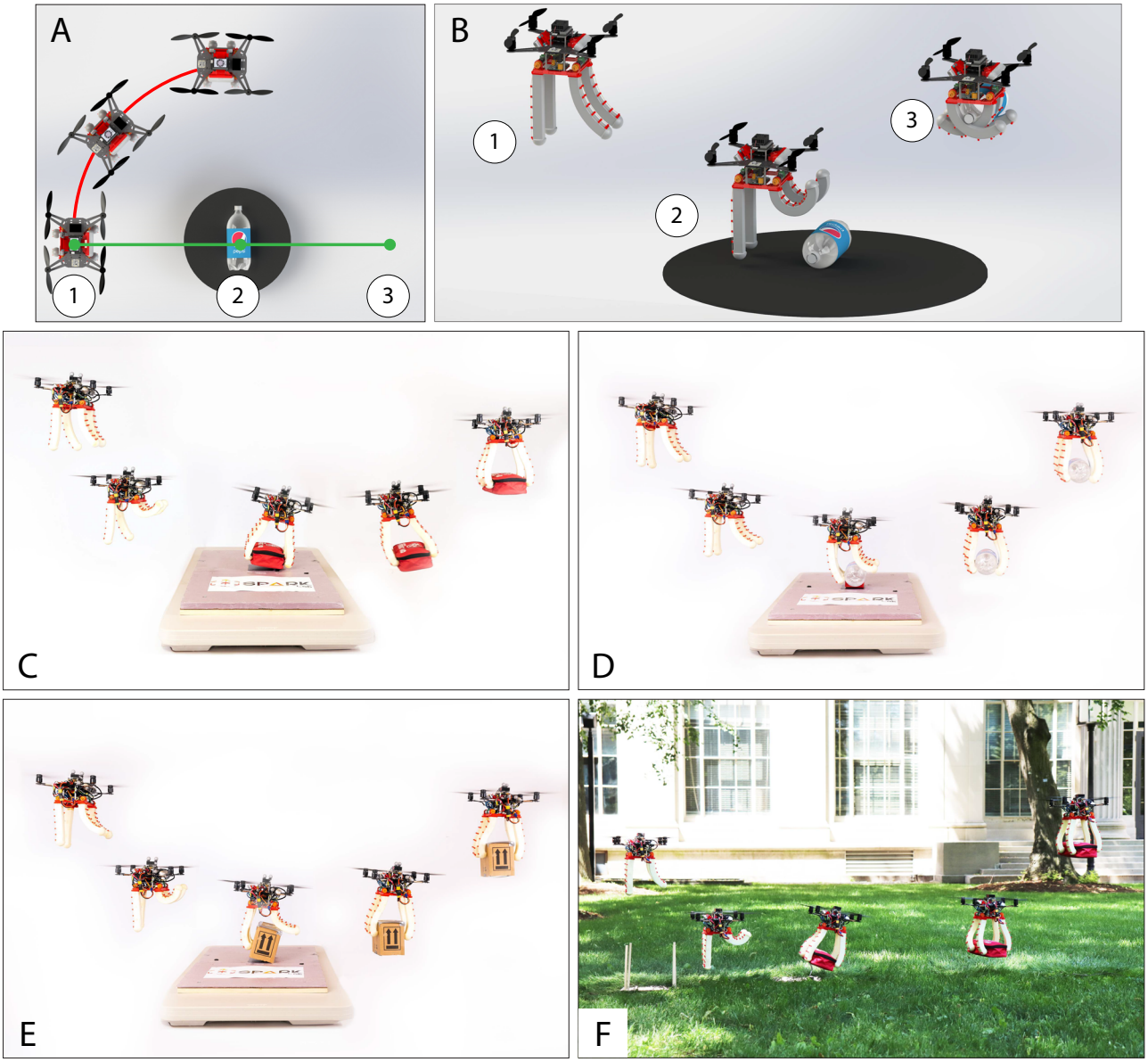


Fig. 5 | Overview and timelapses of experiments with static targets. **A** Experimental setup for the vision-based experiments. The red curve depicts the distribution of random start points, the green line the grasp trajectory; the numbers denote the grasp planning point ①, the grasp point ②, and the terminal point ③. **B** Configurations of the gripper states through the course of the grasp trajectory: ① target observation state, ② pre-grasp state, and ③ post-grasp state. **C**–**E** Static vision-based grasps at 0.5 m/s for med-kit, two-liter bottle, and cardboard box, respectively. **F** Outdoor vision-based grasped of the med-kit at 0.5 m/s.

rotation error. Furthermore, only the target's yaw is needed for grasp planning and at most has roughly 6 degrees of error.

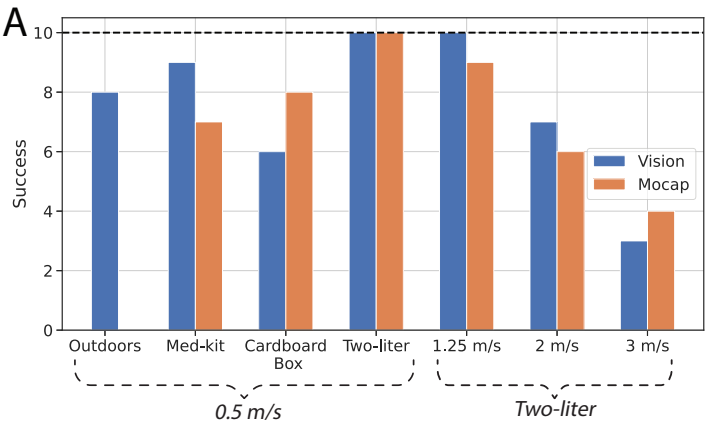
The average error between the quadrotor's actual position and VIO-based estimate (Fig. 6C) up until the grasp point is about 2 cm, for a forward speed of 0.5 m/s.

Finally, the trajectory tracking performance (Fig. 6D, E) before the grasp point is comparable for all three objects, with approximately 5 cm position error, and 0.05 m/s velocity error. One cause of tracking error before the grasp time is the presence of the *ground effect*, where the drone experiences greater thrust efficiency when approaching a horizontal surface[37], thus raising the drone vertically above its desired setpoint (Fig. S1). This is partially compensated by the adaptive controller, and the ability of the soft fingers to safely contact the ground reduces the risk from overcompensation. Apart from specific aerodynamic effects, our platform has a low thrust-to-weight ratio, which limits its control authority and invokes slight tracking errors even in normal flight conditions.

After the grasp, the higher mass of the med-kit and cardboard box lead to increased tracking errors primarily in the vertical direction (Fig. S1), which reduce over time as the adaptive controller learns to compensate. It is important that both pre- and post-grasp errors are minimal for a successful grasp: high pre-grasp errors may cause the drone to be misaligned and miss the target, while high post-grasp errors could cause unintended collisions with the environment (e.g., floor) and destabilize the drone.

Further breakdown of errors across separate axes is presented in Fig. S1, Tables S1–3, and Supplementary Text S1, and information on how these error metrics are measured is provided in Section "Measuring sources of error". A discussion on the maximum error bound that can be tolerated for successful grasps is included in Supplementary Text S1.

Grasp speed analysis. We further evaluate the performance of our system by increasing the desired forward grasp velocity to 1.25, 2, and 3 m/s for the two-liter bottle. We conducted ten flights for each speed with the full vision-based system, and repeated another ten flights for



B

Target	Position Error [cm]	Rotation Error [deg]	Yaw Error [deg]
Med-kit	4.26	7.39	2.16
2-liter	4	7.28	6.27
Cardboard Box	5.02	9.69	5.64

C

Commanded Speed [m/s]	Actual Speed [m/s]	VIO Drift [cm]
0.5	0.47	2
1.25	1.04	2.69
2	1.58	4.39
3	2.15	7.08

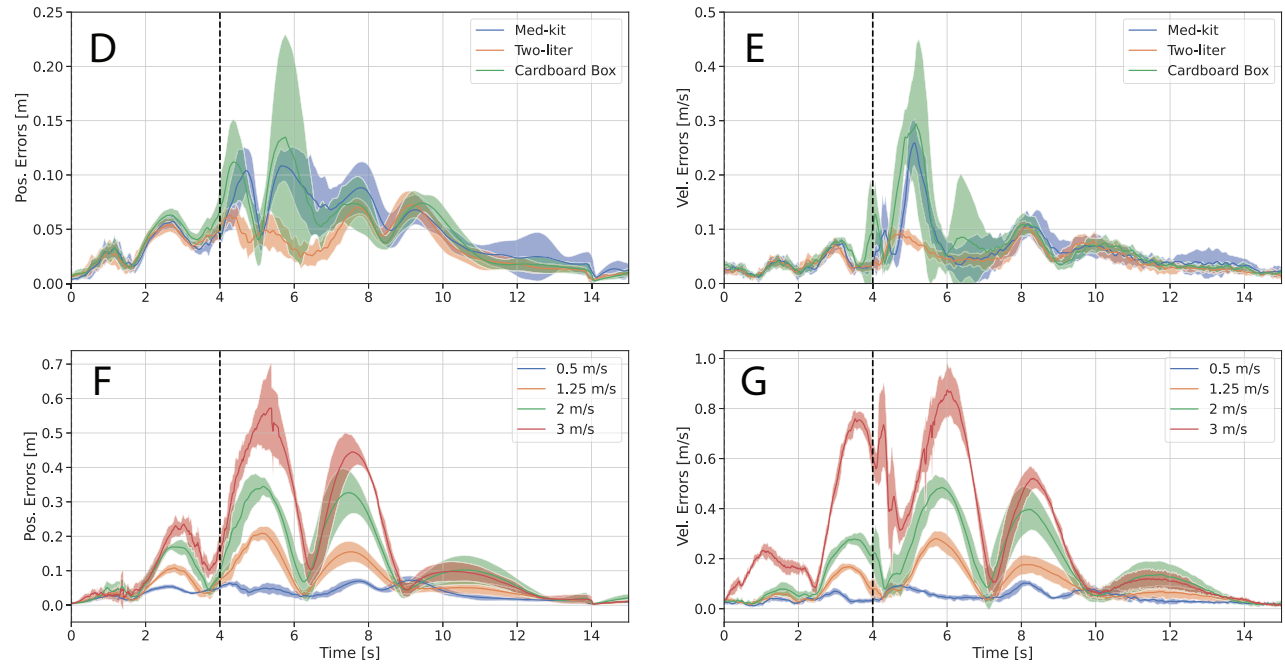


Fig. 6 | Static grasp experiments results. A Success rate for each experiment, comparing motion-capture-based performance (when available) with onboard-vision-based performance. Ten flights were conducted for each experiment, as indicated by the horizontal dashed line. **B** Refined target pose estimation errors for each object at the time of trajectory planning, averaged across all runs. **C** Actual forward velocity as measured by motion capture and VIO drift for varying commanded speeds, averaged across runs. **D**, **E** Position and velocity tracking errors over the grasp trajectory for the three objects. **F**, **G** Position and velocity errors for four different speeds. The vertical dashed line denotes the time of grasp and the shaded regions represent one standard deviation from the mean. The errors here are computed as the mismatch between the vision-based estimate and desired setpoint.

each speed in motion capture as a baseline. Our system is capable of grasping the bottle with full vision at over 2 m/s with a success rate of 3/10 (Fig. 6A). To our knowledge, this is the fastest fully vision-based aerial grasp reported in the literature.

The increases in speed are accompanied by larger tracking errors in both position and velocity (Fig. 6F, G). Because there exists a large discrepancy between the desired and actual speeds at grasp, we report the drone's average actual grasp speed as measured by the motion capture system in (Fig. 6C). This error can be attributed to lower control authority at higher speeds; our platform operates near its maximum payload capacity and struggles to accelerate quickly. However, the error is mostly along the longitudinal direction (i.e., grasp or forward axis), with the lateral and vertical errors remaining low regardless of speed. Longitudinal tracking errors have little effect on our grasp success as the gripper control is informed by the current position of the drone, not its desired position. As long as the drone's state estimate remains accurate up until the time of grasp, the grasp will be triggered at the appropriate time. We validate this by analyzing the VIO drift for the varying speeds (Fig. 6C). VIO drift does increase with grasp speed, remaining manageable for desired speeds less than 2 m/s, but reaching a substantial 7 cm for a desired speed of 3 m/s. As reflected in Fig. 6A, this increase in VIO error proves to have a non-negligible effect on grasp performance. Tables S1–S3 further break down VIO error per axis for each speed.

Outdoor grasps. A central motivation for our onboard perception system is the ability to operate in places where a localization infrastructure is unavailable. To emphasize this capability, we demonstrate the quadrotor picking up the med-kit in an outdoor field (Fig. 5F) at 0.5 m/s. Grasping objects in outdoor environments presents several additional challenges, including unpredictable wind disturbances, varying illumination conditions, and visually diverse surroundings. Furthermore, the rough, sustained ground plane introduces an additional challenge compared to the indoor flights where the target was placed on a smooth, raised pedestal. Post-grasp, the delay in adapting to the target's mass and thrust occlusion causes the drone to loose altitude temporarily. The experimental setup of our indoor environment allows the target to slide

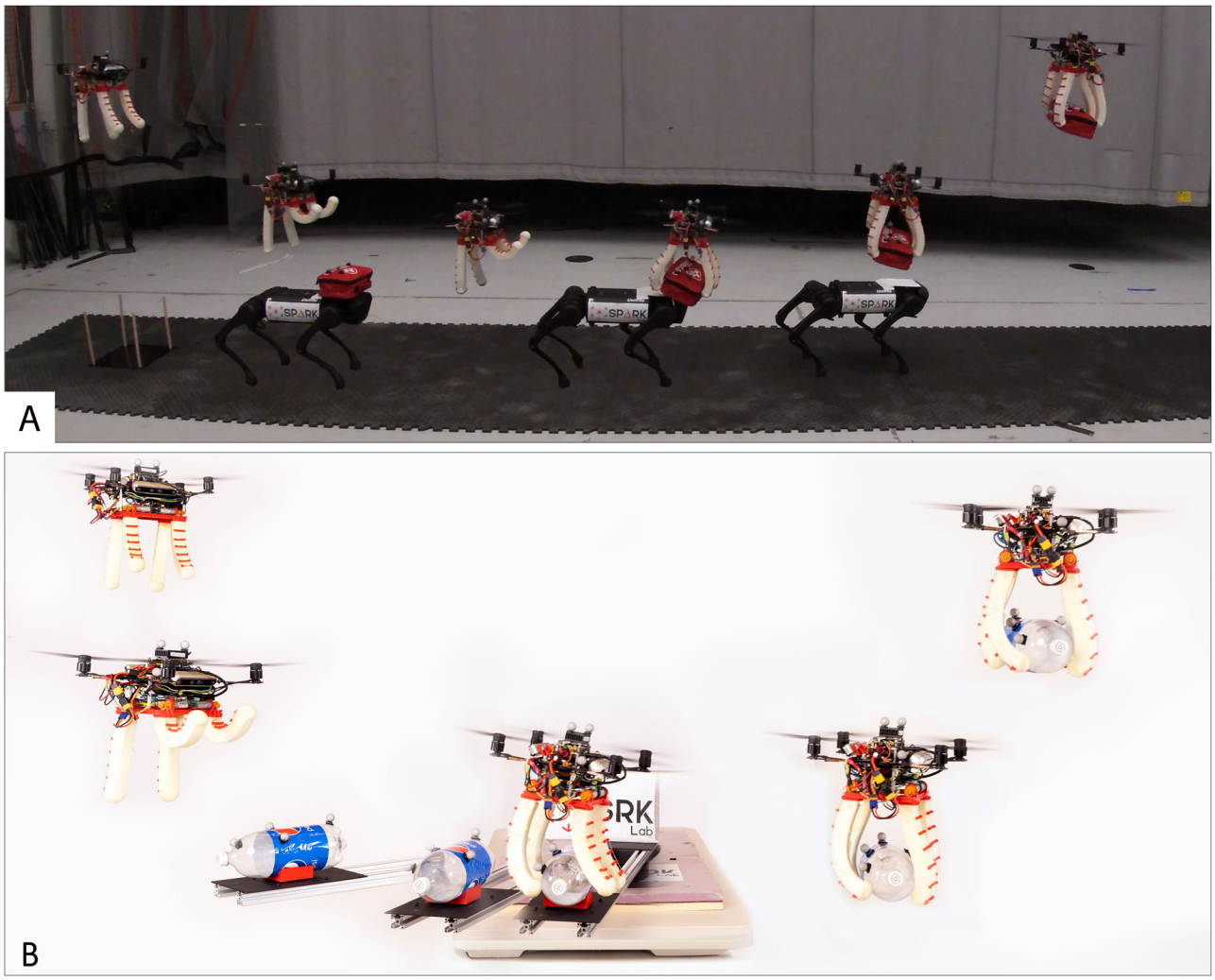


Fig. 7 | Timelapses of experiments with moving targets. A Moving grasps of med-kit attached to a quadruped robot that has forward velocity of 0.3 m/s and with a relative grasp speed of 0.1 m/s. **B** Moving grasps of two-liter bottle on a turntable moving with a tangential speed of 0.08 m/s and with a relative grasp speed of 0.5 m/s; see Movie 1 for more visualizations.

smoothly as it briefly drags against the platform. The duration of this sliding contact is also limited by the platform's length; this setup was designed to accommodate a safety net which would protect the drone's components during crashes and landing. In the unconstrained, outdoor environment, we noticed that the grass inflicts a much higher resistance over a longer duration and induces a large disturbance that can destabilize the drone or weaken the grasp connection (Movie 1). To combat this, we introduce a feedforward acceleration impulse that counteracts the immediate vertical disturbance post-grasp and briefly raises the drone up from the ground plane. Afterwards, the drone continues tracking the nominal trajectory. We implement this term by directly adding a constant, positive vertical acceleration command to our controller slightly after the drone grasps the target, for a duration of 0.6 seconds. We also note that the starting point of the drone was left fixed, rather than randomized as in the indoor experiments. We demonstrate a success rate of 8/10, and, to our knowledge, the first instance of dynamic manipulation in an outdoor environment.

Grasping moving targets

In this section, we investigate grasping *moving* targets at high velocities, or with high relative velocity between the drone and the target. These maneuvers are common in biological systems, where birds of prey must quickly capture their prey before it can escape. In robotics applications, this capability can increase efficiency by removing the need to decelerate before grasping as well as minimize the time the drone must spend near the ground in dangerous environments.

Grasping moving targets presents additional challenges on top of the ones observed with static targets. Because our trajectory optimization does not directly consider camera FOV constraints, the target may not remain in frame during the grasp. Additionally, motion blur and synchronization issues further corrupt the target estimate during the grasp trajectory. Because of this, we cannot rely on measurements to update the target's state and instead must predict its trajectory by rolling out the dynamics. However, this would necessitate strong assumptions of the target's dynamics. Our system is unable to simultaneously handle these errors from perception and dynamics, so we demonstrate the ability to grasp moving objects with the aid of motion capture for drone and target localization.

Our first set of experiments with moving targets consists of the med-kit mounted on top of a Unitree A1 quadruped robot (Fig. 7A). A human operator steers the quadruped in a roughly linear motion (Fig. 8B). Even though the trajectory is nominally linear and constant velocity, there is noticeable variation. We conducted 10 flights at a slow quadruped forward velocity (roughly 0.15 m/s) and a fast forward velocity (roughly 0.3 m/s) (Fig. 8C). The drone was commanded to grasp the med-kit with a relative forward velocity of 0.1 m/s. The drone achieved 10/10 success for the slow speed, and 6/10 for the fast (Fig. 8A). This difference can primarily be

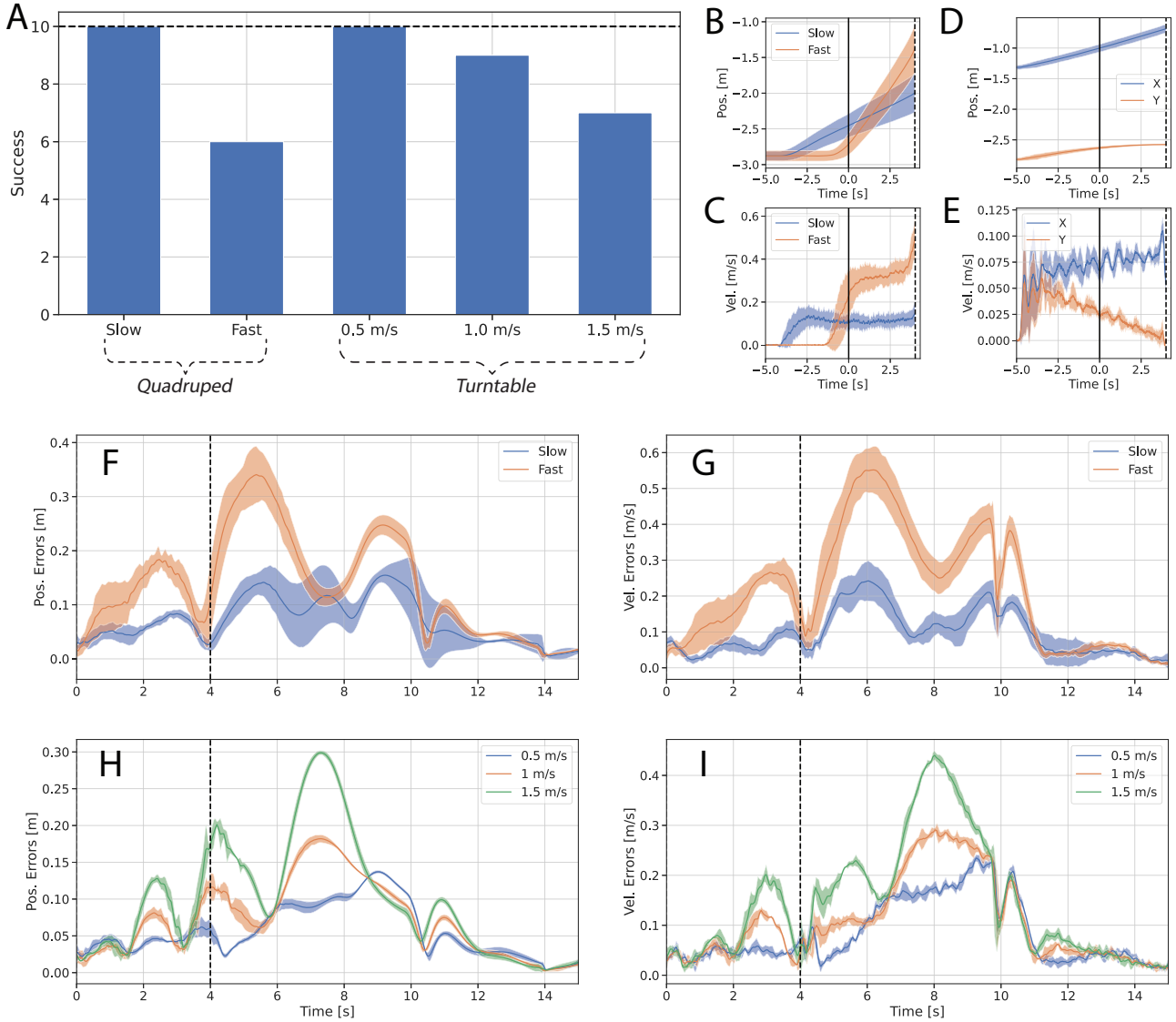


Fig. 8 | Moving grasp experiments results. A Success rates for each experiment, including different moving platforms (Quadruped and Turntable) and increasing speeds. **B**, **C** Position and velocity of the target for the quadruped experiments. **D**, **E** Position and velocity of the target for the turntable experiments. Here, *X*, *Y* represent the axes in the inertial floor plane. **F**, **G** Position and velocity tracking errors for the quadruped experiments. **H**, **I** Position and velocity tracking errors for the turntable experiments. For all plots, the solid black line denotes the start of the grasp trajectory, and the dashed black line is the time of grasp (after which the target pose is unavailable).

attributed to the increased tracking errors seen at grasp point for the higher speed (Fig. 8F, G), with noticeable increases in lateral and vertical errors (Fig. S2, Tables S2 and S3).

For our second set of dynamic experiments, the two-liter bottle sits on a lever arm attached to a motorized turntable (Fig. 7B). With a fixed rotation speed of the turntable, we perform 10 tests for each desired relative grasp speeds of 0.5, 1.0, and 1.5 m/s. For all experiments, the two-liter bottle follows a circular arc (Fig. 8D) with a tangential speed of approximately 0.08 m/s (Fig. 8E). Circular motion is challenging for our approach; because the drone tracks the target at a lever arm, the angular velocity of the bottle gets amplified into large translational velocities at the drone's setpoint. Despite this, the drone exhibited high success rates for the three speeds (Fig. 8A). Similar to the the static speed analysis, the tracking errors increase with relative speed (Fig. 8H, I). However, these errors remain mostly concentrated in the longitudinal axis (Fig. S2, Tables S2 and S3), which has little effect on the grasp success. The rotational motion does induce higher lateral errors with speed (Table S2), which causes a misalignment that contributes to failed grasps at high speeds.

Discussion

We advance the state of the art towards deployable, high-speed, and versatile aerial manipulation. We accomplish this by combining the desirable properties of a soft gripper with an advanced quadrotor platform and a state-of-the-art perception system. The gripper's reliable performance—even with imperfect grasp placement—and its ability to decouple the quadrotor dynamics from contact constraints enable high speed grasps. The system's ability to operate outside of a controlled motion-capture room is vital in nearly every practical application, enabling aerial vehicles to perform manipulation tasks such as emergency supply distribution, package pickup and delivery, and warehouse automation.

The compliant fingers of our soft gripper are key to enabling the full system to grasp at high speeds. As the drone flies faster, its position estimate suffers and tracking errors increase. Furthermore, the detections of the target's pose have several centimeters of error and VIO drift adds additional error. In spite of these error sources, the system successfully picks up objects at over 2 m/s, with full onboard vision.

The gripper mechanism is mechanically simple, which we have found to be a great advantage in manufacturability and repairability. Each finger assembly requires only a handful of 3D-printed parts, fishing line, off-the-shelf foam, and a small motor. The fingers themselves have been surprisingly resilient when attached to the drone. Their most common failure modes are tears that develop near the base, which we have been able to easily repair with common super glue. They also provide cushion to the drone during hard landings; during our development we have seen the drone fall over six feet onto solid surfaces and not require any repair. Even in these cases, the fingers did not break and were used successfully for subsequent grasps.

Controlling soft fingers can be challenging, as they are difficult to model. In comparison to rigid manipulators, which have a well-defined finite dimensional state space, soft fingers are a continuum and have an infinite-dimensional state space. This makes it difficult to relate control input to finger state, which is necessary for our gripper to open its fingers for object grasping while staying out of the cameras' field of view when necessary. We have demonstrated that FEM-based modeling and control enable us to accurately approximate the continuum mechanics of our soft fingers, enabling reliable grasp performance. Our approach is readily generalizable to different, task-specific gripper configurations, which could be designed to exploit the morphology of a specific target, following the geometric analysis in Supplementary Text S1.

We see several potential avenues for future work. One limitation of our pose estimation pipeline is its assumption of prior knowledge of the target object's geometry and visual features. Leveraging recent advances in category-level keypoint detection, e.g.,[38], would enable the system's perception pipeline to generalize to other instances of the same semantic category (e.g., multiple types of packages or soda bottles). Our drone is also currently limited to picking up light objects. We expect that scaling up the drone platform relative to the gripper would lead to a wider range of graspable objects without having to change the rest of the system, as a larger drone would have more control authority and more reliable trajectory tracking both before and after grasp. Finally, full vision-based grasps of moving targets is an immediate next step for our system. Grasping a moving target using only vision-based inputs with our current system requires predicting the object's motion when it goes out of frame before the grasp point, inevitably adding error to the object pose estimate. Timing offsets between the drone's pose estimate and the camera image frame also manifest as error in the object pose estimate. These two additional sources of error currently prevent our drone from performing vision-based grasps of moving targets. More sophisticated trajectory prediction techniques or constraints on the target's possible motion might be needed to enable these moving grasps.

Materials and methods

Hardware design

Drone design. Our drone platform (Fig. 1C) is a quadrotor inspired by the *Intel Aero Ready to Fly* (RTF) drone. It has a custom waterjet carbon fiber frame, but uses the same Yuneec brushless DC motors and propellers as the RTF drone. A Pixhawk 4 Mini flight controller running custom PX4 firmware handles the low-level adaptive controller. The drone is powered by a four-cell Lithium Polymer battery, which provides about three minutes of flight time (enough for three grasp trajectories). The total weight of the drone (excluding the gripper) is 1442g.

The drone features two complementary Realsense cameras: a D455 for estimating the target pose and a T265 for VIO estimation. The D455 captures RGB-D images at high resolution and with accurate depth information. The RGB images are ideal for inference with our keypoint detector, as the high resolution color images aid feature detection and the accurate depth is necessary for transforming keypoints from 2D pixel locations to 3D coordinates in meters. The T265 camera has wide FOV, fisheye lenses which help strengthen the VIO estimate by significantly increasing the amount of detected visual features. Furthermore, the T265 performs VIO on-chip, which alleviates the computational demands on our onboard computer. We note that using these cameras in tandem is necessary for the performance of our system. Using a single T265 would have poor target estimation as the images are grayscale, low resolution, and there is no direct depth estimation. Using a single D455 would require running a VIO pipeline on the computer, which would exceed our computational limits.

We position the D455 in the front, pitched down at 35 degrees to provide a good viewing angle for detecting the target. The T265 is positioned in the rear, where it can observe environmental features without facing significant occlusions from raising the front fingers in preparation for grasp. The T265 is also pitched down slightly, since we expect it to detect many visual features on the ground. Because both cameras are downward facing and, intuitively, the gripper must be mounted on the bottom of the drone, the unconstrained movement of the fingers will naturally cover parts of the camera frames. Finger movement can cover up visibility of the target or corrupt our VIO estimate. Because of the T265's wide FOV, it is difficult to create a drone-gripper geometry where the gripper does not occlude the cameras. However, by adhering to the FOV constraints in Section "Soft gripper design and modeling", our design can be compact and durable without sacrificing performance.

We observed the T265's odometry estimation failing under high-vibration conditions, so we added vibration damping silicone grommets between the camera and drone. The camera required physical masking on the bottom of the camera lens to match the field of view constraint used in the gripper's tendon length optimization, as the camera does not support software-defined region masking. Ensuring that the fingers were not in the camera frame was vital, as the visual odometry estimate when the fingers entered the frame proved to be quite poor.

The visual target estimation pipeline and trajectory planner are run on a Jetson Xavier NX. Both Realsense cameras are connected to the Xavier via USB 3.0 and a powered USB hub. The Xavier communicates with the Pixhawk over UART. When using motion-capture pose estimates instead of visual navigation, the Xavier receives pose updates over Wifi from a base station connected to the motion capture system. See Table S4 for detailed component specifications.

Gripper design. The drone interfaces with the gripper through standoffs connected to a 3D printed base plate, which houses the gripper components (Fig. 1D). A custom-made, lightweight PCB embeds an ATMega32U4 microcontroller to handle logic control, 2 DRV8434 motor controllers which actuate four 31:1 gear ratio 12V DC motors (49 g), and a voltage regulator and connector to convert the Lipo battery's voltage into a stable 12V. Jumper cables are wired from GPIO pins on the Xavier to the PCB to connect the gripper state machine with the low level gripper controller.

The gripper features four passively closed, cable actuated fingers (Fig. 3B). Each finger is molded using Smooth-On *FlexFoam-iT! X*[39] and a custom 3D-printed mold, which follows a circular arc (Fig. S4). The initial state of the finger is then its closed position, which is maintained by the elasticity of the mesh. A 3D-printed finger-base connector is super glued to the top of the finger and can be slotted into the base plate for quick replacement. The tendon guides are positioned between evenly placed foam extrusions and adhered with super glue. Braided fishing line is fixed to the last tendon guide and the motorized winch, passing through the interior tendon guides. As the winch tightens, the tendon contracts, compressing the outer edge of the finger and forcing it open. As the winch loosens, the elastic force of the mesh returns the finger to its default closed position. Each finger weighs 54 g and the total weight of the gripper is 544 g.

We position the four fingers in a symmetric, rectangular configuration. The front two fingers are commanded identically, as are the rear two fingers. The spacing between the fingers is designed to yield sufficient positioning error leniency for the three grasp targets (Fig. S3). Using four fingers offers a degree of redundancy; we have noticed several instances of successful grasps with only three of the fingers making contact. Placing the fingers in this orientation also enables us to exploit the pre-grasp state depicted in Section

"Soft gripper design and modeling", which is necessary for robust, high-speed grasps.

We measure the power consumption of our gripper by measuring the change in battery capacity over a set duration. The gripper uses 0.294 W of power in the pre-grasp state, 0.104 W in the target observation state, and 0.01 W while closed. Over the course of a grasp, the gripper approximately maintains the pre-grasp state for 2 s, the target observation state for 11 s, and the closed state for 34 s. From this, we estimate the gripper consumes 2.072 J of energy over the course of one grasp. The energy consumption is comparable to other bi-stable grippers. Wang et al.[40] report a total energy consumption of 0.1386 J for a similar grasp operation, but our gripper is capable of grasping targets with over 10 times the mass.

Target object perception and pose estimation

We present a robust pose estimation system that combines a keypoint detector with a state-of-the-art algorithm for keypoint-based 3D pose estimation, namely Teaser++[36]. Estimated poses are further refined through a fixed-lag smoother (Fig. 2). We also developed a semi-automated data annotation tool to rapidly train the keypoint detector.

Semi-automated keypoint annotation. Keypoint annotation can be a costly process, requiring multiple points to be labeled in each image of a large image dataset (e.g., with tens of thousands of images). We draw inspiration from prior methods of speeding up keypoint labeling by using known camera poses at training time[41,42], enabling a small number of manually labeled keypoints to be projected into many images. Figure 4A presents an overview of the annotation pipeline. The target object and calibration board are positioned such that both are visible in the recorded camera frames, and they are fixed for the entirety of the recording process. We move the D455 camera at varying distances and angles from the object and record all RGB and depth images. The pose of the camera with respect to the calibration board for each image can be readily determined by computing the camera extrinsics using the calibration board. Known camera poses enable reasoning about the 3D position of each keypoint across the whole trajectory sequence. First, pixel locations for the keypoints in user-selected images are manually annotated using the Matlab tool GUI (Fig. S5). The 3D position for each keypoint relative to the camera is determined by back-projecting the pixel coordinates using the depth of each keypoint, given by the D455. The keypoint positions can be further refined by minimizing the reprojection error in a small number of manually labeled training images, and then reprojected into all other images based on the camera model.

In our tests, we only annotate keypoints on the top face of the object and assume that no keypoints are occluded in our training set; this is a valid assumption considering the viewing angle of our drone. Proper annotation of keypoint visibility is important to prevent the neural net from associating features blocking the view of the keypoint with the true keypoint features. The labeling method does support keypoints placed on all sides of the object, but self-occlusions require visibility detection. This can be done by sampling the depth of each keypoint using the depth image and comparing that against the expected depth obtained by transforming the keypoint relative to the camera. If there is a large discrepancy, the keypoint is assumed to be occluded. This method requires a highly accurate depth sensor and the D455 proved to be inadequate.

We add variety in the training set to improve generalization of the keypoint detector by collecting data in several environments: a motion capture room, a cluttered workshop, a lounge space, a foyer, and an outdoor field (Fig. S6). For each location, the data collection process was repeated multiple times with different poses of the target object. A randomly selected subset of the runs were added to the validation set. Across the three targets, 39,548 training images and 15,172 validation images were collected and annotated, within only a few days of manual labor.

Keypoint detection. We use the architecture provided by a state-of-the-art human keypoint detector, *trt-pose*[43], and train the entire model for our tasks. The network is comprised of a ResNet-18 backbone, followed by a keypoint prediction head. The training data is augmented using standard techniques (perturbations of translation, rotation, scale, and contrast) to further improve diversity of the training set. We present the full set of parameters we used for trt-pose's training pipeline in Table S5. Training takes around seven hours on an Nvidia GeForce GTX 1080 Ti GPU, with the loss converging after 75 epochs.

The original RGB images have size 1280×720 and are cropped by removing the top half and resized into 912×256 images to improve inference speed. Due to the geometry of the camera mount, cropping the top half of the images does not reduce performance as objects in that region are too far away to detect reliably anyway. The trained network is compiled to leverage the Jetson Xavier's hardware using TensorRT, and runs at approximately 14 Hz on the Xavier.

Figure 4D shows the results of the keypoint detector on the med-kit test set, along with visual representations in Fig. 4C. Keypoint error is defined as the pixel distance between the estimated and ground-truth keypoints, averaged across all keypoints, and only for keypoints that were valid detections (an invalid detection occurs when the keypoint isn't detected and the network predicts $(0, 0)$). The error remains roughly constant with distance, but we note that pixel error is influenced by the scale of the target. At far distances, the target's perceived area is smaller and a unit of pixel error translates to higher error in world coordinates than a unit of pixel error at close distances.

Robust point cloud registration. From the depth image provided by the D455 camera and the detected keypoint pixel coordinates, we create an estimated point cloud of keypoints, denoted as $\mathcal{A} = \{\boldsymbol{a}_i\}_{i=1}^{N}$. Furthermore, the corresponding keypoints are also annotated on the known object CAD model through the data annotation tool, leading to a second point cloud, namely $\mathcal{B} = \{\boldsymbol{b}_i\}_{i=1}^{N}$. Solving for the transformation between these two point clouds allows us to estimate the pose of the target relative to the camera. We seek to find the rotation $\boldsymbol{R} \in SO(3)$ and translation $\boldsymbol{t} \in \mathbb{R}^3$ which minimize the following *truncated nonlinear least squares* problem[36]:

$$\min_{\boldsymbol{R}\in SO(3),\boldsymbol{t}\in\mathbb{R}^3} \sum_{i=1}^{N} \min\left(||\boldsymbol{b}_i - \boldsymbol{R}\boldsymbol{a}_i - \boldsymbol{t}||^2, \bar{c}^2\right), \tag{3}$$

where $\bar{c}$ represents the upper bound on the residual error for a measurement to be considered an inlier, and is set to be 10 mm. In the truncated least squares problem in Eq. (3), measurements with residual error smaller than $\bar{c}$ are used to compute a least-squares estimate, while residuals greater than $\bar{c}$ have no effect on the estimate and are discarded as outliers. In our application, this is useful for filtering out outliers caused by spurious keypoint or depth detections. In practice, we utilize Teaser++[36] to solve this problem, which takes on average 2 ms to solve a single registration problem on the Xavier NX. Fig. 4E, F demonstrate the results of our approach for the med-kit object. Translation error is defined as the Euclidean distance between the estimated and ground-truth translations, while rotation error is computed as the geodesic distance between the estimated and ground-truth rotations. Additionally, we only report data for detections that had at least one inlier point. We nominally plan trajectories at a distance of 1.25 m from the target, which equates to roughly 5 cm of translation error and 15 degrees of rotation error. Our experiments have validated that this error is well tolerated when combined with the fixed-lag smoother and the robustness of the soft gripper.

Fixed-lag smoother. Estimates of the target's relative pose from individual images are fused together in a fixed-lag smoother to reduce noise in the estimate and provide estimates for linear and angular velocity. Knowledge of the target's velocity is necessary for grasping moving targets at the desired relative speed. We rely on the estimation process described below to infer velocity.

We model the target's motion as a nearly constant velocity model[44], with the target's pose $\boldsymbol{T}_k \in SE(3)$ and linear and angular velocity $\boldsymbol{\xi}_k \in \mathbb{R}^6$ at time k:

$$\begin{aligned} \boldsymbol{T}_{k+1} &= \boldsymbol{T}_k \boxplus (\boldsymbol{\xi}_k dt) \\ \boldsymbol{\xi}_{k+1} &= \boldsymbol{\xi}_k + \boldsymbol{\nu}_k dt, \end{aligned} \tag{4}$$

where $\boldsymbol{\nu}_k \in \mathbb{R}^6 \sim \mathcal{N}(\boldsymbol{0}, \Sigma)$ represents white-noise angular and linear acceleration, and $\boxplus$ maps a vector in $\mathbb{R}^6$ (on the right-hand-side of the operator) to $SE(3)$ (e.g., via an exponential map or, more generally, a *retraction*[45]) and then composes it with the pose on the left-hand-side of the operator. Given observations of the target's pose over the last τ timesteps, namely $\hat{\boldsymbol{T}}_{T-\tau:T}$, we estimate $\boldsymbol{T}_{T-\tau:T}$ and $\boldsymbol{\xi}_{T-\tau:T}$ as

$$\begin{aligned} \min_{\boldsymbol{T}_{T-\tau:t_T}, \boldsymbol{\xi}_{T-\tau:T}} \quad & \sum_{k=T-\tau}^{T} ||\boldsymbol{T}_k \boxminus \hat{\boldsymbol{T}}_k||^2_{\Omega_1} \\ & + ||(\boldsymbol{T}_k \boxplus (\boldsymbol{\xi}_k dt)) \boxminus \boldsymbol{T}_{k+1}||^2_{\Omega_2} \\ & + ||\boldsymbol{\xi}_k - \boldsymbol{\xi}_{k+1}||^2_{\Omega_3} + ||\boldsymbol{\xi}_k||^2_{\Omega_4}, \end{aligned} \tag{5}$$

where $\boxminus$ denotes a tangent space representation of the relative pose between two poses. The first term encourages the estimate to match the target pose observations, the second term constrains the estimated velocity to be consistent with the poses, and the third term forces changes in velocity between timesteps to be small. The final term provides regularization on the velocity estimate. We found a small amount of velocity regularization to be necessary for consistent initialization of the filter. $\Omega_1, \Omega_2, \Omega_3, \Omega_4$ are tunable weight matrices to trade off among the four terms (see Table S6). The optimization is implemented in C++ using the GTSAM library[46]. The incremental iSAM2 algorithm[47] is used to optimize (5), as it enables reuse of computation between timesteps.

Trajectory optimization and control

To enable our grasps, we compute a smooth polynomial trajectory for the drone, passing at a set distance over the target (the *grasp point*). This nominal trajectory is tracked with an adaptive controller which compensates for the added mass after grasping and for other external disturbances. Throughout the grasp procedure, the drone's current position is fed into a gripper state machine, which sets the finger winch angle based on the results of an offline FEM-based optimization (Fig. 2). The trajectory optimization and control are based on prior implementations for motion-capture based grasping[22], with added functionality enabling trajectory planning during flight and updating of the reference trajectory to account for moving targets.

Minimum-snap polynomial trajectory optimization. We compute a fixed-time polynomial aligned with the target's grasp axis and constrained to start at the drone's initial position, pass through the grasp point, and end at an arbitrary final position. The grasp point is defined as the target's position, plus a tunable offset to account for the length of the fingers and any misalignment between the target frame center and the geometric center. The end point can be selected arbitrarily and was chosen to be 1 m above the target and 1.7 m behind the target. We seek to find a polynomial that minimizes the 4th derivative of position, or snap, of the drone. As shown in[32], polynomials of this form enable smooth, continuous motion which allows for stable, consistent grasps. The key idea which enables grasping moving targets is that the polynomial is planned with respect to the *target's* frame, rather than the inertial frame. As the target moves the polynomial moves with it, inducing a disturbance which is compensated for by our low-level tracking controller. The polynomial provides a series of position, velocity, and acceleration setpoints that are tracked by the drone's flight controller. The drone's yaw setpoint is set to always point toward the estimated object position. Supplementary Text S2 provides further exposition on our specific trajectory optimization implementation.

Adaptive quadrotor control. The drone is subject to various forms of disturbances that could impede both grasp success and post-grasp flight performance. After the target is grasped, the effective mass of the quadrotor increases and the thrust efficiency decreases, as the grasped object partially obstructs the propellers. Estimates of system model parameters are also imperfect, and benefit from online adaptation. Additionally, we observe that our system is also sensitive to variable disturbances such as wind in outdoor environments or the ground effect. To mitigate these disturbances, we adopt the geometric adaptive control law presented in[33], that we briefly review below.

The external disturbances in our system primarily affect the translational dynamics of the drone, with negligible impact on the rotational dynamics. Hence, we model the quadrotor dynamics as:

$$\begin{aligned} m\ddot{\boldsymbol{p}} &= m\boldsymbol{g} + f\boldsymbol{b}_z + \boldsymbol{\theta}_f \\ \dot{\boldsymbol{R}} &= \boldsymbol{R}\hat{\boldsymbol{\Omega}} \\ \boldsymbol{J}\dot{\boldsymbol{\Omega}} &= -\boldsymbol{\Omega} \times \boldsymbol{J}\boldsymbol{\Omega} + \boldsymbol{\tau} \end{aligned} \tag{6}$$

where m is the drone's mass, $\boldsymbol{J}$ is the moment of inertia, $\boldsymbol{g}$ is the gravity vector, $\boldsymbol{p} \in \mathbb{R}^3$, $\dot{\boldsymbol{p}} \in \mathbb{R}^3$, $\boldsymbol{R} \in \mathrm{SO}(3)$, $\boldsymbol{\Omega} \in \mathbb{R}^3$ are the position, velocity, rotation, and angular velocity of the drone, respectively, f is the scalar thrust force applied along the local vertical direction $\boldsymbol{b}_z$, $\boldsymbol{\tau}$ is the torque applied by the quadrotor, and $\boldsymbol{\theta}_f$ is the unknown translational disturbance. The *hat* operator $\hat{\cdot}$ maps a vector in $\mathbb{R}^3$ to an element of the Lie algebra $\mathfrak{so}(3)$, i.e., a 3×3 skew symmetric matrix; see[33].

We then compute the quadrotor's thrust and moment as:

$$f = -\boldsymbol{b}_z^\top (k_p \boldsymbol{e}_p + k_v \boldsymbol{e}_v + m\boldsymbol{g} - m\ddot{\boldsymbol{p}}_d + \bar{\boldsymbol{\theta}}_f) \tag{7}$$

$$\begin{aligned} \boldsymbol{\tau} = & -k_r \boldsymbol{e}_r - k_\Omega \boldsymbol{e}_\Omega + \boldsymbol{J}\boldsymbol{R}^\top \boldsymbol{R}_d \dot{\boldsymbol{\Omega}}_d \\ & + (\boldsymbol{R}^\top \boldsymbol{R}_d \boldsymbol{\Omega}_d)^\wedge \boldsymbol{J}\boldsymbol{R}^\top \boldsymbol{R}_d \boldsymbol{\Omega}_d \end{aligned} \tag{8}$$

where $\boldsymbol{p}_d$, $\boldsymbol{R}_d$, $\boldsymbol{\Omega}_d$ are the desired position, attitude, and angular velocity, respectively, $\boldsymbol{e}_p$, $\boldsymbol{e}_v$, $\boldsymbol{e}_r$, $\boldsymbol{e}_\Omega$ are the position, velocity, rotation, and angular velocity errors, $\bar{\boldsymbol{\theta}}_f$ is the estimated disturbance on the translation dynamics, and k_p, k_v, k_r, k_Ω are user-specified control gains. We refer the reader to[33] for the definition of the errors and a more comprehensive discussion.

The estimated disturbance is adjusted online using the following law:

$$\frac{d\bar{\boldsymbol{\theta}}_f}{dt} = \Pi\left(\gamma_f (\boldsymbol{e}_v + k_{af} \boldsymbol{e}_p)\right) \tag{9}$$

where γ_f, k_{af} are user-specified gains and Π is a suitable projection function.

This formulation ensures that the tracking errors asymptotically go to zero even in the presence of unknown disturbances[33].

FEM-based gripper control. To optimally actuate the gripper, the controller must deform its fingers such that they effectively enclose the target object, while simultaneously ensuring that they do not obstruct the view of the onboard cameras. To balance these two goals, we employ an optimization-based control method. At the core of our control method is a finite element-based (FEM) model.

The gripper's volume is discretized into a finite element mesh with nodal positions $\boldsymbol{x}$, and simulated cables are routed through this mesh. Given cable contractions $\boldsymbol{u}$, we can solve for a corresponding statically stable pose

$$\boldsymbol{x}(\boldsymbol{u}) = \arg\min_{\boldsymbol{x}} E(\boldsymbol{u}, \boldsymbol{x}) \tag{10}$$

by minimizing the total potential energy of the system E using Newton's method. This general approach can be elegantly extended to account for dynamics, and is done in[28,48].

Our control optimization is a *nested optimization*. Its objective $\mathcal{O}(\boldsymbol{u}, \boldsymbol{x}(\boldsymbol{u}))$ is written in terms of the physically valid pose $\boldsymbol{x}(\boldsymbol{u})$, found by

solving Eq. (10). We minimize this objective using gradient descent to find optimal control inputs for a desired configuration

$$\boldsymbol{u}^* = \arg\min_{\boldsymbol{u}} \mathcal{O}(\boldsymbol{u}, \boldsymbol{x}(\boldsymbol{u})). \tag{11}$$

Computing the gradient of our control objective requires the Jacobian $\frac{d\boldsymbol{x}}{d\boldsymbol{u}}$ relating changes in control inputs $\boldsymbol{u}$ to changes in the corresponding physically-valid mesh shape $\boldsymbol{x}(\boldsymbol{u})$, which we solve for using direct sensitivity analysis, as described in[49].

Our specific control objective $\mathcal{O}$ is the weighted sum of two sub-objectives, which are illustrated in Fig. 3D–E:

$$\mathcal{O} = \mathcal{O}_{\text{grasp}} + \mathcal{O}_{\text{FOV}} \tag{12}$$

The first sub-objective, $\mathcal{O}_{\text{grasp}}$, drives the gripper's fingers toward a configuration where they can effectively grasp an object. The sub-objective

$$\mathcal{O}_{\text{grasp}} = ||(\boldsymbol{s} - \boldsymbol{y}_1(\boldsymbol{u})) \times (\boldsymbol{s} - \boldsymbol{y}_2(\boldsymbol{u}))||^2 \tag{13}$$

works by maximizing the area enclosed by the gripper's fingertips and a *target point**s*** (Fig. 3E[22]). Here, $\boldsymbol{s}$ represents the centroid of the target shifted from the drone by a nominal offset which approximates the relative position between the drone and target at the moment the gripper begins to close. The fingertip positions $\boldsymbol{y}_1(\boldsymbol{u})$ and $\boldsymbol{y}_2(\boldsymbol{u})$ are defined in terms of the physically-valid mesh position $\boldsymbol{x}(\boldsymbol{u})$ using barycentric coordinates.

The second sub-objective, $\mathcal{O}_{\text{FOV}}$, ensures the the gripper's fingers do not obstruct the camera's field of view. This is accomplished by defining a plane with point $\boldsymbol{p}$ and normal $\boldsymbol{n}$, analogous to the bottom face of a camera's view frustum (Fig. 3D). To prevent the finger from entering into the camera's field of view, the sub-objective

$$\mathcal{O}_{\text{FOV}} = \sum_i f\big((\boldsymbol{p} - \boldsymbol{x}_i(\boldsymbol{u})) \cdot \boldsymbol{n}\big) \tag{14}$$

sums over the squared penetration depths of all nodes in the mesh. Here, the function

$$f(z) = \begin{cases} z^2 - \varepsilon z + \frac{\varepsilon^2}{3}, & \text{if } \varepsilon < z, \\ \frac{z^3}{3\varepsilon}, & \text{if } 0 < z \le \varepsilon, \\ 0, & \text{otherwise} \end{cases} \tag{15}$$

is a smooth one-sided quadratic, driving the mesh outside the camera's field of view while ensuring the optimizer is well-behaved. Depending on the grasp phase, $\mathcal{O}_{\text{FOV}}$ is added for both the front and the rear camera (to obtain the *target observation state*) or only for the rear camera (to obtain the *pre-grasp state*).

Measuring sources of error

The success of our vision system depends on three sources of error: target estimation error, VIO drift, and trajectory tracking error. We collect motion capture data alongside the vision flights to benchmark these errors, although the quadrotor only uses vision-based data for its own control.

To quantify the amount of target estimation error (Fig. 6B), we placed motion capture markers near the target, but far from the key visual features. The target's motion capture frame was manually adjusted until it visually appeared aligned with the expected target's vision frame. Estimation error is then reported as the difference between our vision based estimate and the motion capture ground truth. VIO drift (Fig. 6C) was measured by aligning the grasp trajectory as reported by the vision system with the recorded motion capture data. This alignment was done using *evo*[50], which performs a least squares optimization to find the optimal rigid transformation between trajectories. Here, we are only concerned with aligning part of the trajectory up until the grasp point, as post-grasp drift generally has little effect on grasp success. Therefore, we align trajectories starting from 9 s before the grasp time up until the grasp time. We found this range to be suitable to be sure that the resulting alignment errors were due to trajectory drift and not because of a lack of data points. Trajectory tracking errors (Fig. 6D–G) were analyzed by extracting the setpoint and pose estimate from the flight controller.

When generating the plots and tables for these metrics, we exclude runs that resulted in catastrophic failures (i.e., the drone crashed after grasp). In total, we discarded two runs: one vision-based run with the cardboard box and one vision-based run with the two-liter bottle at 3 m/s desired speed. These runs were counted as failures. Additionally, for each of the turntable experiments, five trials were conducted with the turntable spinning clockwise and five were conducted counterclockwise. Because there is large variation between these cases, the error plots only use the clockwise case. The success rates use both cases.

Data availability

All data necessary for the conclusions of this paper are in the main text or Supplementary Materials. The flight data, keypoint detector training data, videos of each experiment, code for analyzing and plotting the data, the automated keypoint annotator, the keypoint detector training pipeline, CAD of our system, and our full flight stack can be found at: https://datadryad.org/stash/share/sf9Us4eZZiVLbog6iqXh4vpxVfcOXmI3ia_7NtF96Hw.

Received: 9 November 2023; Accepted: 17 May 2024;
Published online: 26 August 2024

References

1. Idrissi, M., Salami, M. R. & Annaz, F. A review of quadrotor unmanned aerial vehicles: applications, architectural design and control algorithms. *J. Intell. Robot. Syst.* **104**, 22 (2022).
2. Joubert, N. et al. Towards a Drone Cinematographer: Guiding Quadrotor Cameras using Visual Composition Principles. *ArXiv* abs/1610.01691 (2016).
3. Hassanalian, M. & Abdelkefi, A. Classifications, applications, and design challenges of drones: a review. *Prog. Aerosp. Sci.* **91**, 99–131 (2017).
4. Gupte, S., Mohandas, P. & Conrad, J. A survey of quadrotor Unmanned Aerial Vehicles. in *2012 Proceedings of IEEE Southeastcon* 1–6 (2012).
5. Nguyen, P. H., Patnaik, K., Mishra, S., Polygerinos, P. & Zhang, W. A soft-bodied aerial robot for collision resilience and contact-reactive perching. *Soft Robot.* **10**, 838–851 (2023).
6. Zufferey, R. et al. How ornithopters can perch autonomously on a branch. *Nat. Commun.* **13**, 7713 (2022).
7. Roderick, W. R., Cutkosky, M. R. & Lentink, D. Bird-inspired dynamic grasping and perching in arboreal environments. *Sci. Robot.* **6**, eabj7562 (2021).
8. Stewart, W., Guarino, L., Piskarev, Y. & Floreano, D. Passive perching with energy storage for winged aerial robots. *Adv. Intell. Syst.* **5**, 2100250 (2023).
9. Bodie, K. et al. An Omnidirectional Aerial Manipulation Platform for Contact-Based Inspection. in *Proceedings of Robotics: Science and Systems* (2019).
10. Aucone, E. et al. Drone-assisted collection of environmental dna from tree branches for biodiversity monitoring. *Sci. Robot.* **8**, eadd5762 (2023).
11. Xu, M., Huang, S., He, R., Yu, D. & Wang, H. Aerial shooting manipulator for distant grasping. *IEEE Robot. Autom. Lett.* **8**, 1991–1998 (2023).
12. Appius, A. X. et al. Raptor: Rapid aerial pickup and transport of objects by robots. in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 349–355.
13. Pounds, P. E., Bersak, D. R. & Dollar, A. M. The Yale Aerial Manipulator: Grasping in flight. in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2011.

14. Ryll, M. and Katzschmann, R. K. Smors: a soft multirotor uav for multimodal locomotion and robust interaction. in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 2010–2016.
15. Thomas, J., Polin, J., Sreenath, K. & Kumar, V. Avian-inspired grasping for quadrotor micro uavs. in *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, vol. 55935. American Society of Mechanical Engineers, 2013, p. V06AT07A014.
16. Spica, R., Franchi, A., Oriolo, G., Bülthoff, H. H. & Giordano, P. R. Aerial grasping of a moving target with a quadrotor uav. in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. (IEEE, 2012) pp. 4985–4992.
17. Rossi, R., Santamaria-Navarro, A., Andrade-Cetto, J. & Rocco, P. Trajectory generation for unmanned aerial manipulators through quadratic programming. *IEEE Robotics and Automation Letters* (2017).
18. Kannan, S., Quintanar-Guzman, S., Dentler, J., Olivares-Mendez, M. A. & Voos, H. Control of aerial manipulation vehicle in operational space. in *2016 8th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*, 2016, pp. 1–4.
19. Yu, H. et al. Catch Planner: Catching High-Speed Targets in the Flight. in *IEEE/ASME Transactions on Mechatronics*, Vol. PP, 1–12 (2023).
20. Chen, T. G. et al. Aerial grasping and the velocity sufficiency region. *IEEE Robot. Autom. Lett.* **7**, 10 009–10 016 (2022).
21. Sun, J. et al. Ick-track: a category-level 6-dof pose tracker using inter-frame consistent keypoints for aerial manipulation. in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 1556–1563.
22. Fishman, J., Ubellacker, S., Hughes, N. & Carlone, L. Dynamic grasping with a "soft" drone: From theory to practice. in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. https://arxiv.org/pdf/2103.06465.pdf (2021).
23. Bauer, E., Cangan, B. G. & Katzschmann, R. K. Autonomous vision-based rapid aerial grasping. Preprint at *arXiv* https://doi.org/10.48550/arXiv.2211.13093 (2022).
24. Ramon-Soria, P., Arrue, B. C. & Ollero, A. Grasp planning and visual servoing for an outdoors aerial dual manipulator. *Engineering* **6**, 77–88 (2020).
25. Lin, L., Yang, Y., Cheng, H. & Chen, X. Autonomous vision-based aerial grasping for rotorcraft unmanned aerial vehicles. *Sensors* **19**, 3410 (2019).
26. Seo, H., Kim, S. & Kim, H. J. Aerial grasping of cylindrical object using visual servoing based on stochastic model predictive control, in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 6362–6368.
27. Thomas, J., Loianno, G., Sreenath, K. & Kumar, V. Toward image based visual servoing for aerial grasping and perching. in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 2113–2118.
28. Bern, J., Banzet, P., Poranne, R. & Coros, S. Trajectory optimization for cable-driven soft robot locomotion. in *Robotics: Science and Systems (RSS)* (2019).
29. Rus, D. & Tolley, M. Design, fabrication and control of soft robots. in *Nature*, Vol. 521, 467–75 (2015).
30. Stokes, A. A., Shepherd, R. F., Morin, S. A., Ilievski, F. & M. Whitesides, G. A hybrid combining hard and soft robots. *Soft Robot.* **1**, 70–74 (2014).
31. Huang, G. Visual-inertial navigation: a concise review. in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 9572–9582.
32. Mellinger, D. & Kumar, V. Minimum snap trajectory generation and control for quadrotors. in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2011, pp. 2520–2525.
33. Goodarzi, F. Lee, D. & Lee, T. Geometric Adaptive Tracking Control of a Quadrotor Unmanned Aerial Vehicle on SE(3) for Agile Maneuvers. in *Journal of Dynamic Systems, Measurement, and Control*, Vol. 137, 091007 (2015).
34. Bern, J. M., Kumagai, G., & Coros, S. Fabrication, modeling, and control of plush robots. in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)* (2017).
35. Kastor, N. et al. Design and manufacturing of tendon-driven soft foam robots. *Robotica* **38**, 88–105 (2020).
36. Yang, H., Shi, J. & Carlone, L. TEASER: fast and certifiable point cloud registration. *IEEE Trans. Robot.* **37**, 314–333 (2020).
37. Del Cont Bernard, D., Giurato, M., Riccardi, F. & Lovera, M. Ground effect analysis for a quadrotor platform. in *Advances in Aerospace Guidance, Navigation and Control*, 2018, pp. 351–367.
38. Manuelli, L., Gao, W., Florence, P. & Tedrake, R., kpam: Keypoint affordances for category-level robotic manipulation. in *The International Symposium of Robotics Research*. Springer, 2019, pp. 132–157.
39. Smooth On, FlexFoam-iT! X. https://www.smooth-on.com/products/flexfoam-it-x/ (2023).
40. Wang, Y., Ujjaval, G., Nachiket, P. & Jian, Z. A soft gripper of fast speed and low energy consumption. *Sci. China Technol. Sci.* **62**, 31–38 (2019).
41. Blomqvist, K. et al. Semi-automatic 3D Object Keypoint Annotation and Detection for the Masses. in *2022 26th International Conference on Pattern Recognition (ICPR)*, 3908–3914 (2022).
42. Liu, X. et al. KeyPose: Multi-View 3D Labeling and Keypoint Estimation for Transparent Objects. in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 11599–11607 (2019).
43. Nvidia, trt_pose. https://github.com/NVIDIA-AI-IOT/trt_pose (2023).
44. Bar-Shalom, Y., Li, X. R. & Kirubarajan, T. *Estimation with Applications To Tracking and Navigation*. (John Wiley and Sons, 2001).
45. Forster, C., Carlone, L., Dellaert, F. & Scaramuzza, D. On-manifold preintegration for real-time visual-inertial odometry. *IEEE Trans. Robot.* **33**, 1–21 (2017).
46. Dellaert, F. & Contributors, G. borglab/gtsam. https://github.com/borglab/gtsam (2022).
47. Kaess, M. et al. iSAM2: incremental smoothing and mapping using the Bayes tree. *Int. J. Robot. Res.* **31**, 217–236 (2012).
48. Li, M. et al. Incremental Potential Contact: Intersection-and Inversion-free, Large-Deformation Dynamics. in *ACM Transactions on Graphics*, Vol. 39, 20 (2020).
49. Bern, J. M. & Rus, D. Soft ik with stiffness control. in *2021 IEEE 4th International Conference on Soft Robotics (RoboSoft)*. (IEEE, 2021) pp. 465–471.
50. Grupp, M. evo: Python package for the evaluation of odometry and SLAM. https://github.com/MichaelGrupp/evo (2017).

Acknowledgements

We would like to thank M. Mohamoud for initial quadrotor design, B. Evans for work assembling and manufacturing the gripper electrical and mechanical components, W. Menken and N. Mondal for electrical design of the gripper circuit board, and J. Fishman for useful discussion on soft aerial manipulation. We would also like to thank V. Murali, P. Lusk, and N. Hughes for feedback on initial drone design and system troubleshooting. *Funding*: This work has been partially sponsored by an MIT Research Support Committee (RSC) award, Carlone's Amazon Research Award, and by MathWorks. In addition, A.R. was supported by a National Defense Science and Engineering Graduate Fellowship.

Author contributions

S.U. designed and fabricated the soft gripper, implemented the perception system, lead the experimental evaluation, helped with system integration, and lead the preparation of the paper. A.R. did substantial design work of the quadrotor, implementedthe fixed-lag smoother and moving polynomial planning, helped with the experimental evaluation, paper preparation,

system integration, and with both hardware and software debugging. J.B. provided insights into control of soft manipulators, implemented the FEM-based optimization, and helped prepare the paper. J.S. provided insights into target pose filtering, helped with the experimental evaluation, and aided design decisions. L.C. conceived the project, provided supervision, guidance, funding, helped prepare the paper, and contributed to both the hardware design and software decisions.

Competing interests
The authors declare no competing interests.

Additional information
Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s44182-024-00012-1.

Correspondence and requests for materials should be addressed to Samuel Ubellacker.