PinpointFly: An Egocentric Position-control Drone Interface using Mobile AR

Linfeng Chen
clinfeng.jlu@gmail.com
Tohoku University, Japan

Kazuki Takashima
takashima@riec.tohoku.ac.jp
Tohoku University, Japan

Kazuyuki Fujita
k-fujita@riec.tohoku.ac.jp
Tohoku University, Japan

Yoshifumi Kitamura
kitamura@riec.tohoku.ac.jp
Tohoku University, Japan

ABSTRACT

Accurate drone positioning is challenging because pilots only have a limited position and direction perception of a flying drone from their perspective. This makes conventional joystick-based speed control inaccurate and more complicated and significantly degrades piloting performance. We propose PinpointFly, an egocentric drone interface that allows pilots to arbitrarily position and rotate a drone using position-control direct interactions on a see-through mobile AR where the drone position and direction are visualized with a virtual cast shadow (i.e., the drone's orthogonal projection onto the floor). Pilots can point to the next position or draw the drone's flight trajectory by manipulating the virtual cast shadow and the direction/height slider bar on the touchscreen. We design and implement a prototype of PinpointFly for indoor and visual line of sight scenarios, which are comprised of real-time and predefined motion-control techniques. We conduct two user studies with simple positioning and inspection tasks. Our results demonstrate that PinpointFly makes the drone positioning and inspection operations faster, more accurate, simpler and fewer workload than a conventional joystick interface with a speed-control method.

CCS CONCEPTS

• **Human-centered computing → Human computer interaction (HCI)**.

KEYWORDS

Visualization, Spatial Perception, Positioning, Videography

ACM Reference Format:
Linfeng Chen, Kazuki Takashima, Kazuyuki Fujita, and Yoshifumi Kitamura. 2021. PinpointFly: An Egocentric Position-control Drone Interface using Mobile AR. In *CHI Conference on Human Factors in Computing Systems (CHI '21), May 8–13, 2021, Yokohama, Japan.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3411764.3445110

CHI '21, May 8–13, 2021, Yokohama, Japan

ACM ISBN 978-1-4503-8096-6/21/05…$15.00
https://doi.org/10.1145/3411764.3445110

1 INTRODUCTION

Due to the increasing functionality of affordable drones, interactive indoor drone applications will be progressively deployed. Examples include videography, structural inspections, monitoring [18], assembling [44], warehouse inspections[14], museum exploration [68], short-range deliveries [14], painting [69], mid-air displays [72], and so on. For such applications, a drone's accurate and arbitrary positioning capability is significant to ensure adequate interactivity and safety in a physically limited indoor space. While many drone interfaces have been proposed for indoor setup (e.g., [38, 42, 43, 46, 58, 74]), their concepts mainly focus on intuitive command inputs to trigger fundamental drone movements or photographing objects based on an image captured by a drone camera. Novice pilots continue to struggle to arbitrarily specify their drone's destinations with the current joystick or prior first-person view (FPV)-based interfaces.

The following are the major reasons for this challenge: (1) Pilots have insufficient spatial perception [36] of the position and direction of flying drones from their own perspectives. (2) In most systems, drone-centric control, which refers to a drone controlled in its local coordinate system, frequently requires pilot effort to recognize and comprehend drone motions. For example, pilots might need to engage in mental rotation to correctly describe input commands. (3) The speed-control mechanism in conventional joystick controllers, where the pilot's input is mapped to the object's velocity, is ill-suited for accurate positioning tasks [75]. Unfortunately, these three issues often appear together, which complicates mapping a joystick's two-stick inputs to a drone's 3D movements in the air. For example, if pilots fail to estimate a drone's status and location, it is difficult to accurately control its angular and linear velocity. This difficulty in turn hinders accurate positioning of the drone at the exact target locations. These issues greatly decrease the piloting performance, cause anxiety, and explain why long-term training is required for novices.

Researchers have actively worked on new drone interfaces to address such issues. A solution for poor spatial awareness of a flying drone is the use of virtual overlay. FlyAR helps pilots understand the automated aerial flight process of drones by visualizing their paths using augmented reality (AR) overlays [78]. Such AR-based visual overlays improve viewer's estimation of the depth distance of floating objects (e.g., [13, 16]). Another solution for coordinate system and control mechanism issues is introducing a new control concept. Many studies have demonstrated that user-centric

(egocentric) control is more efficient and natural than the drone-centric control of conventional joystick controllers. For example, the benefits of such egocentric strategies as the headless mode [12], a plane-operation-based egocentric interface [74], and body gestures [15, 58, 59, 63, 65] have been verified. However, to the best of our knowledge, no interface has achieved accurate and arbitrary drone operations by simultaneously addressing the above three issues.

We propose PinpointFly, an egocentric position-control drone interface using mobile AR. Our goal is to achieve accurate and arbitrary drone control through direct manipulations that allow pilots to move drones to any position and direction in the given space as they wish (just like controlling visual elements in a graphical user interface). With PinpointFly, pilots are able to select the exact position of the flying drone which is visually highlighted on the mobile AR device. With this enhanced spatial perception, the pilots can arbitrarily design subsequent positions and motions of the flying drone by position-control interactions over the mobile AR's touchscreen. All interactions can be described in the user's coordinate system. In this paper, we focus on designing a PinpointFly interface for visual-line-of-sight (VLOS) [54] indoor operation scenarios and motion control techniques. In two user studies, we separately verified two of PinpointFly's interaction elements: drone positioning and videography.

Our previous work was demonstrated with a two-page abstract [10]. This present paper reports its significant updates with in-depth design considerations, user studies, and discussions. The following are the contributions of this paper:

(1) Proposed PinpointFly, a new drone interface design for indoor or short-range VLOS (visual-line-of-sight) operation, with an egocentric position-control concept using mobile AR and motion-tracking technologies.
(2) Presented detailed interaction designs for fundamental drone positioning and videography controls whose effectiveness was empirically confirmed by two formal user studies.
(3) Discussed its potential for various applications, current effectiveness, and future improvements.

2 RELATED WORK

2.1 Background of Drone Interface and Application

The largest application of the current drone market is videography. To assist videography, many drone automation algorithms and methods have been proposed to programmatically manage the entire flight process and trajectory (e.g., [3, 4, 11, 22, 25–27, 50, 71, 73]). However, their purpose is not to support interactive real-time drone manipulations, but to computationally generate feasible drone movements. With such automatic drone control technology, drones are now expected as autonomous delivery systems and in-situ navigation [40].

In the context of interactive drone control, more unique applications have been attempted. For example, drone display [21] has huge potential for indoor applications such as physical props [17, 33], 3D-content representation [5, 28], or mid-air displays [61, 72, 77]. Inspection[49] and delivery applications [35, 41] are increasingly attracting attention. These applications are also more challenging than simple FPV-based videography (e.g.,[37, 43]) because the pilots require stronger spatial awareness of the flying drone beyond the FPV from the drone camera, for example, where the drone is, where the next target is, and in which direction the drone now heads in relation to the entire inspection area. Such application examples clearly establish the need for a new control mechanism that allows pilots to freely position drones in any location and angle in the space to ensure safe and interactive drone operation capabilities.

2.2 Object-centric Drone Interfaces

Xpose is a touch interface that automatically generates drone movements for videography around an object specified by the user in FPV [43]. StarHopper is its extension that supports beyond visual line of sight (BVLOS) operation by adding a top view from the overhead camera[46]. This object-centric approach offers a powerful videography and inspection experience where pilots can focus on the target objects in the FPV without regard to the drone's actual position in the space. However, the drone's automatic movements (e.g. [55]) are limited within the predefined area or to identifiable targets, and additional conventional joystick inputs are required for further fine-tuning.

2.3 Egocentric Drone Interfaces

The egocentric concept is identical to the user-centric concept. Joystick based drone controllers normally use a drone-centric interface where the input depends on the current drone direction. The headless mode is included in the egocentric concept and can reduce the risk of crashes in VLOS operations [12]. Third-person piloting [66] provides an additional bird-view perspective using a second drone, supporting egocentric BVLOS drone manipulations . Many egocentric approaches have been achieved. Here, we focus on Augmented Reality (AR) and Natural User Interface (NUI) approaches for an indoor egocentric drone interface.

2.3.1 AR-based egocentric drone interface. Touchscreen interactions through AR views are very popular egocentric interfaces [19, 30, 57, 67] and have been increasingly applied to drone controllers. exTouch uses a spatially aware egocentric approach to manipulate moving physical objects (e.g., drones) through an AR-mediated mobile interface [38]. This enables spatial projection of the user's input onto the actuated target's motions. Yonezawa et al. used a virtual plane for defining the movements of a drone based on a position-control concept [74]. Its user first moves the plane's position and direction on a mobile AR view and draws a moving path on the plane to give the drone's moving path. While these work shares similar concepts to us, they primarily support fast input for abstract drone motions and not detailed drone positioning. FlyAR is a visualization technique that helps pilots understand drone's flying trajectory by AR overlays [78]. Yet FlyAR does not provide a drone's direction visualization and only works for pre-programmed drone control. ARPilot allows for drone flight plan by physically moving the AR mobile device around the target scene shown on it for BVLOS scenarios [11]. This does not support accurate real-time indoor VLOS operations because the approach that physically moves AR devices is not capable of pinpoint positioning accuracy. While many AR-based egocentric UI has been proposed, to our

best knowledge, there is no interface achieving an accurate and arbitrary drone control in short-range operation area.

2.3.2 NUI-based egocentric drone interface. NUI generally allows users to describe object movements in user coordinate systems. For example, Flying head [32] is a method that synchronizes the user's natural head and body movements with UAV motions. This offers pure egocentric drone control for improving the telepresence experience but it does not explore either position control or enhanced spatial perception. Gesture-based drone interfaces have been actively researched (e.g., [20, 52, 58]) to make drone control more intuitive and fun. Voice-based control has also been introduced [8, 9, 48]. These interfaces are intuitive and allow segments of the drone movement paths to be input. But this approach does not support detailed positioning instructions unless both target locations are known as well as pilot's spatial perception. DroneCTRL enables pilots to directly navigate drones by manipulating a laser pointer with two buttons to designate their target 3D positions [42]. This idea resembles our work because it includes both egocentric and position-control concepts. However, this approach with laser pointers would struggle to rapidly specify 3D locations in the air due to that pilot's spatial perception is not helped.

Another approach combined AR and NUI. Erat et al. proposed Drone-augmented human vision [18] where pilots can view a hidden area (e.g., a room behind a wall) through an AR headset and control the drone in that area by gestures. A user study demonstrated the benefits of its egocentric approach in such unique BVLOS scenarios. Huang et al. expanded the AR-based drone control interface using natural language [34]. Target locations are indicated and visualized by gestures in the AR view and voice commands are sent to a flying drone to manipulate its movements. The both works sufficiently integrate AR technology and a set of body gestures; yet their interaction response is slow due to the technical limitations of gesture and voice recognizers that cannot achieve quick and accurate drone movements.

Our work expands such prior work by offering enhanced spatial perception and touchscreen-based motion controls through mobile ARs for accurate and arbitrary 3D drone positioning.

2.4 Improving Spatial Perception using AR and 3DUI

We introduce several AR/VR researches that improve user's spatial perception of objects. AR visual overlays increase the user's understanding of the spatial relationships of virtual and physical information. For instance, such graphical hints as object-aligned virtual cutaways [79] or Magic Lenses [60] support depth perception. Wither suggested using virtual shadow planes and color-encoded markers for improving it [70]. Diaz et al. also concluded that the virtual cast shadows of floating objects are powerful depth cues [16]. Hedayati et al. showed that AR can improve user's ability to synthesize information collected by a robot (e.g., a camera stream) with the contextual knowledge of its movements in the environment [31]. Adding graphical hints has also been utilized in 3DUIs. For example, Shift-sliding uses the cast shadow representation of floating objects to help users understand their 3D positions as well as repositioning them by sliding interaction in 3D scenes [64]. From such a large body of research, virtual shadow overlays would significantly improve the viewer's understanding of the spatial relationships between objects and environments. We apply this idea to enhance a pilot's spatial perception for flying drones.

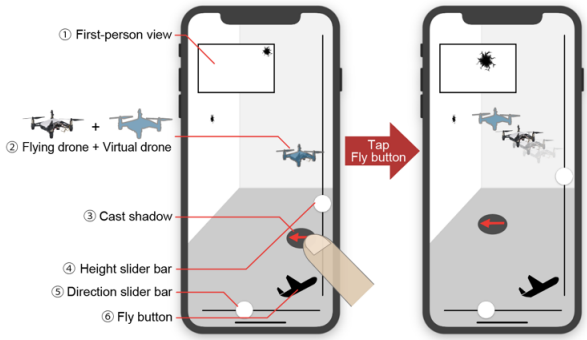


Figure 1: AR touch interface

3 PINPOINTFLY DESIGN

Since our survey indicates that no practical position-control drone interface has been achieved that enables pilots to directly and accurately position drones at any location and angle in an indoor space, we propose and design PinpointFly, a novel position-control drone interface. This section describe all of our interaction designs for it.

3.1 Concept

The followings are our three key design concepts derived from our motivation and survey:

1. Visually enhance spatial perception. Since a pilot's perception of a flying drone is limited, we improve it by visually augmenting its position and direction with a see-through mobile AR approach and a simple cast shadow metaphor (#3 in Fig. 1 left). This step clearly provides the drone's ground position and head direction.

2. Egocentric perspective. The conventional joystick-based controller uses a drone's coordinate system (drone-centric) where the drone's responses to the pilot input are inconsistent from the pilot's perspective depending on its head direction. We use an egocentric perspective where all the actions of a drone can be described in the user's coordinate system independently of the drone's head direction, which has been proven to be easier and safer [12].

3. Position control. Our primary concept is position control, which would considerably improve the performance, more than speed control, where the precise positioning is required, based on prior 3DUI (3D user interface) knowledge (e.g., [76]). Although speed control is more convenient in large spaces, our target is delicate drone movements in VLOS areas.

We acknowledge that our work stands solidly on the shoulders of a large body of HCI, XR, and 3DUI researches, and thus, the above individual concepts are not completely novel. However, our primary design challenge and contribution complementarily incorporate each concept from different domains into a unified interface with synergic effects.

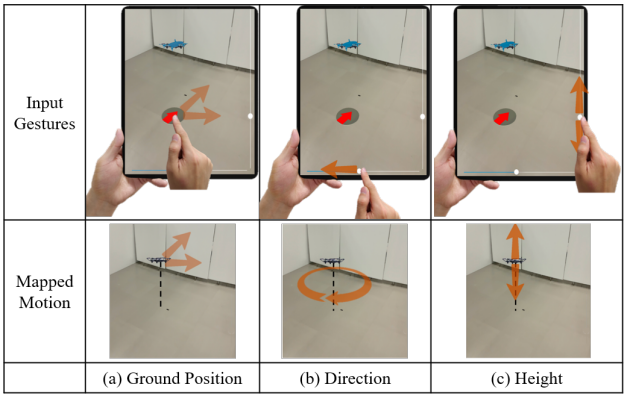


Figure 2: Drone's ground position, direction, and height control by touch interactions

3.2 Drone Positioning Interface

3.2.1 AR touch interface. Fig. 1 shows our AR interface using a mobile device (a smartphone or tablet). This acts as a viewfinder with a touch interface to manipulate drone movements. This also provides an FPV window (#1 in Fig. 1). Other interactions can be performed by manipulating the elements shown in #2 to #6. To simplify the graph, we omitted other detailed icons, such as landing. As shown in the figure, the AR interface captures the wall and floor surfaces of a room and the flying drone (#2 in Fig. 1), and the pilot manipulates the drone's 3D positions and angles through touch interactions. One challenge for manipulating the drone in the 3D space using the 2D touchscreen is how to convert 2D touch input to 3D positional information (e.g., the drone's target positions). We followed the Shift-Sliding [64] idea and used separate controls for the drone's ground position, direction, and height. By combining these, the pilot can indicate the drone's exact target positions and directions in the 3D space. These separate controls require multiple steps to make complex movements, but supports stabler and simpler interactions than mid-air interaction, such as the laser pointer approach [42]. The cast shadow can be dragged to change the ground position of the flying drone (#3 Fig. 1), and the height and direction slider bars (in #4 and #5 Fig. 1) are used to change the drone's height and direction.

3.2.2 Virtual drone and cast shadow visualization. To manage control status and improve a pilot's spatial perception, we introduced two key visualizations: a virtual drone (#2 in Fig. 1) and the cast shadow (#3 in Fig. 1). The virtual drone represents a goal (next state) of the drone, and the cast shadow with a red arrow represents the ground position and head direction of the virtual drone. The cast shadow has two roles: as an icon that helps the pilot's spatial perception of the drone and a ground-position manipulator that can be slid by drag interaction.

Fig. 1 left shows that the pilot touches the cast shadow icon to control the flying drone. Before the pilot begins manipulating it, the flying drone and virtual drone are fully overlapped in the AR interface, where the pilot can visualize the ground position of the current flying drone. The pilot can then indicate the next target's ground position by dragging the cast shadow icon. While dragging on the screen, the virtual drone also moves to the corresponding position indicated by the cast shadow icon. The new virtual drone's and cast shadow icon's positions let the pilot understand the exact ground position of the drone's next state (i.e., destination). Finally, after the pilot removes her finger off the screen, or after tapping the fly button (#6 in Fig. 1), depending on the motion-control technique, the flying drone immediately moves to the indicated target position (Fig. 1 right).

3.2.3 Ground position, height, and head direction control. As described above, to control the drone's ground position, the virtual drone's cast shadow is dragged to the desired target position on the touchscreen (Fig. 2(a)). The icon is then slid on the floor captured on the AR interface. The pilot can also move the AR device to frame an off-screen area.

To control the drone's head direction (yaw angle), the pilot adjusts the position of the knob on the horizontal slider bar at the bottom of the touchscreen (Fig. 2(b)). The slider bar's length corresponds to 360-degree rotation and the middle point of the slider bar corresponds to the facing direction of the user after initialization. For example, when the user slides the knob to the left, the virtual drone and its red arrow will rotate counterclockwise. To determine the drone's height, the pilot adjusts the vertical slider bar's knob (Fig. 2(c)). The maximum and minimum heights are currently set to the room's height and the ground respectively, which can be updated by recalibrating. We could have chosen simultaneous controls using multitouch gestures, e.g., pinch in/out and rotation on/around the cast shadow. Instead, considering the system reliability and applicability to other AR devices, we rely on a single finger interaction for accurate and stable inputs.

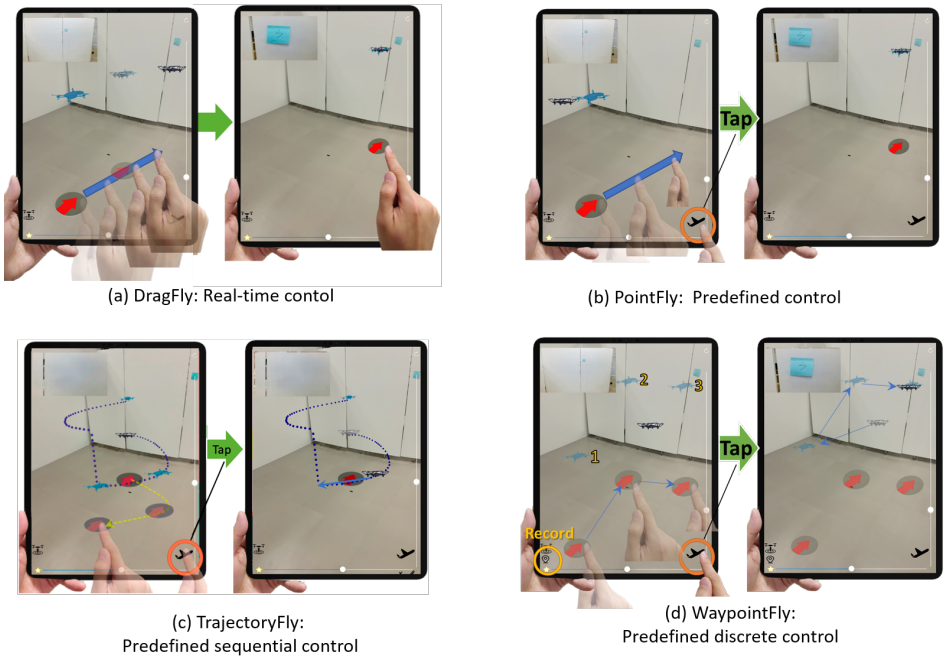
Although these designs can be further modified depending on the type of mobile device, we believe that the current design satisfactorily achieves our three concepts and is sufficient for evaluating the core idea of PinpointFly.

3.3 Motion-Control Techniques

With the drone's fundamental controls, we designed four motion-control techniques for PinpointFly system, DragFly, PointFly, TrajectoryFly, and WaypointFly, summarized in Fig. 3, to cover major drone operations: real-time, predefined discrete, and predefined sequential. Each technique can create drone's fundamental or segmented operations, and each can be easily switched in the current interface.

3.3.1 DragFly. Pilots can change the position and direction of the virtual drone on the screen by dragging the cast shadow (Fig. 3(a) left) and/or adjusting the height and direction using slider bars. The virtual drone's designated state is immediately input to the flying drone so that it follows the virtual drone in real time (Fig. 3(a) right). The pilots can also move around the operation area and/or move the mobile device to capture the flying drone and/or to drop the virtual drone at its intended area. This technique is appropriate for various on-line operations, such as videography, inspections, and wall painting.

3.3.2 PointFly . Similar to DragFly, pilots can designate the virtual drone's position and direction by simple drag and tap gestures on



(a) DragFly: Real-time contol

(b) PointFly: Predefined control

(c) TrajectoryFly: Predefined sequential control

(d) WaypointFly: Predefined discrete control

Figure 3: Four motion-control techniques: (a) DragFly, (b) PointFly, (c) TrajectoryFly, (d) WaypointFly

the screen and/or slider bars. However, the flying drone will move only after the pilot taps the fly button on the screen (Fig. 3(b)). After tapping, it moves to the indicated position (Fig. 3(b) right) through the nearest path using the PID feedback control algorithm [39]. This two-step method helps pilots carefully plan drone operations for point checks, selfies, and delivery.

3.3.3 TrajectoryFly. The pilots can record the movement paths of the virtual drone by drawing a path on the screen by dragging the cast shadow and/or the slider bars. The cast shadow and slider bar manipulations are easily switchable and the input path are sequentially recorded. After the path is designed and the fly button is tapped (Fig. 3(c) left), the flying drone automatically moves along the designed path (Fig. 3(c) right). Since the direction control cannot be recorded in this TrajectoryFly, pilots can manually regulate the drone's direction by adjusting the horizontal slider bar during this automatic drone movement. Such combined two-step and online controls are powerful for special perspective videos (e.g., nose-in circle) and drone-path planning, which are difficult for beginners with a joystick.

3.3.4 WaypointFly. Fig. 3(d): Similar to TrajectoryFly, pilots can designate waypoints (position and direction) anywhere by moving the cast shadow or the slider bar. Unlike PointFly, pilots can designate multiple waypoints to define complex drone operations. Each waypoint can be indicated by tapping a record button, and each virtual drone is numbered sequentially on the screen. After tapping the fly button, the drone moves to the recorded waypoints. This two-step option is useful when pilots are aware of observation points, for example, inspecting damage in architecture or infrastructure. Although the current implementation is limited to motion control, more detailed parameter adjustments, such as waiting times and visiting order, can be added.

4 PINPOINTFLY IMPLEMENTATION

4.1 Components and Workflow

We implemented a proof-of-concept prototype. Fig. 4 shows our workflow designed for an indoor scenario. The main components are a drone, a tracking system, a server, and a mobile AR device, which all communicate using User Datagram Protocol (UDP) .

4.1.1 Drone. We only have two requirements for the drone: motion programmable and FPV available over WiFi. We choose DJI TELLO ($98mm \times 93mm \times 41mm$) in this implementation since it is small enough for VLOS operation scenarios. Other small-size drone products like DJI Mavic or Spark are also available.

4.1.2 Tracking system. We expect on-drone SLAM-based tracking system would be available in the near future that enable us to deploy the system anywhere. In our current prototype, we have two tracking system options: aruco markers [24] and an optical motion capture system like OptiTrack [51]. Fig. 4 shows an example using aruco markers on the floor. A bottom-facing camera is mounted on the drone, and its captured video is sent to a video receiver connected to the server by 5.8-GHz wireless transmission (solid red line in Fig. 4). The Ubuntu server running our program written in Python calculates the drone's position and direction based on the received video images with captured markers using the OpenCV ArUco Library [23, 56]. Another tracking system is a motion capture system. For factory, classroom, or sport studio scenarios with fixed-capture areas, this option might be more suitable, since such optical

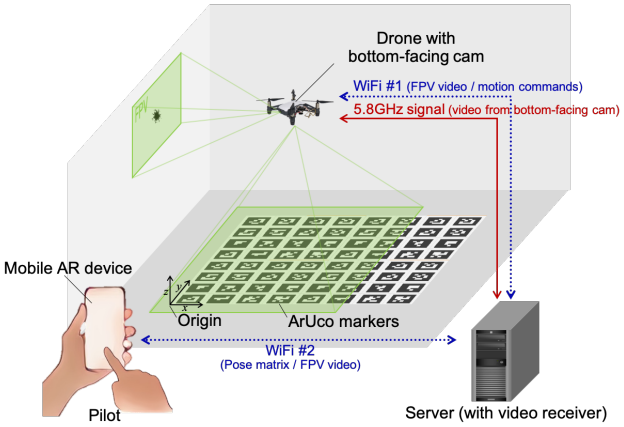


Figure 4: PinpointFly system workflow

markers are little enough to be fixed on a small indoor drone and offer a larger tracking area without visual noise like aruco markers.

4.1.3 Communication. A WiFi connection (#1 in Fig. 4) is used between the server and the flying drone to exchange motion commands using an open source library called TelloPy [29] and an FPV video stream. Another WiFi connection (#2 in Fig. 4) is created between the server and the mobile AR device held by the pilot. The mobile AR device (iPhone X or iPad Pro 2018 11 inch) runs our see-through mobile AR software to obtain the virtual drone's 3D pose matrix through Apple's ARKit library. The calculated virtual drone's pose matrix is sent to the server through the same WiFi (WiFi #2). The server also sends the drone's FPV video to the mobile device using the same WiFi network.

4.2 Calibration and Drone Control

Before using the system, we required a one-time calibration where the pilot captures several markers around the left bottom corner of the marker matrix on the floor (for the origin of the markers, see Fig. 4) by the mobile's video camera. This process makes coordinates for the mobile match markers and defines the shared coordinate system and the origin.

When the virtual drone position is decided on the mobile AR touchscreen, the output of PID feedback control algorithm [39] moves the flying drone to the 3D position indicated by the virtual drone. For safety and simplicity, the target speed in the PID feedback control algorithm was set to 150 cm/sec.

4.3 System Performance

We briefly validated the PinpointFly's positioning accuracy by calculating the distance error between the positions of the flying and virtual drones. The mobile device sends the pose matrix of the virtual drone to the server, which calculates the pose matrix of the flying drone in 60 FPS. Then the server can calculate the position/angle errors using these two pose matrices. Our test for OptiTrack version showed that the average errors of the distance and direction were $3cm$ (standard deviation (SD) = 0.8) and around 5 degrees (SD = 1.2). These values slightly varied depending on the area within the tracking volume. We also tested the version of aruco markers ($2m \times 2m \times 1.7m$) in a normal lighting environment. Since the accuracy of this version heavily depends on the drone's altitude, we tested the drone control $1.4m$ above the ground, which is relatively challenging for the current setup with the prepared aruco markers and a drone-mounted camera. The absolute distance error and direction error were $3.6cm$ (SD = 1.5) and 5.4 degrees (SD = 2.6). In both cases, PinpointFly's accuracy was very close to its tracking system accuracy. Using a better tracking system (e.g., higher resolution IR cameras, bigger markers) might increase the positioning accuracy.

We also measured the system's latency from giving a command until the drone begins to fly. Since multiple steps are required to complete the command, we manually assessed the total delay time by counting the number of elapsed frames through video coding. We found an average latency of 0.32 sec. (SD= 0.14), hinting at PinpointFly's adequate responsiveness for interactive drone-positioning scenarios.

5 USER STUDY 1: POSITIONING PERFORMANCE

We conducted two separate user studies to understand PinpointFly's two major interactions: its fundamental drone-positioning performance and practical videography performance with four motion techniques.

The first study examined how accurately and quickly pilots position their drones using DragFly interface in simplified VLOS operations. We only tested DragFly because it supports real-time control, which is appropriate for discussing both accuracy and time effects. We set a baseline for a joystick controller for the following reasons: (1) since joysticks have the largest familiarity in the drone community, we expect that active drone pilots can infer PinpointFly's performances based on their own daily experiences. 2) Although related egocentric drone interfaces exists (e.g., [38, 74]), we were unable to formally compare them because their systems and positioning accuracy data were unavailable. 3) Unlike 3DUI researches, a fundamental understanding of two control mechanisms (position-control and speed-control) remains unestablished in the drone community.

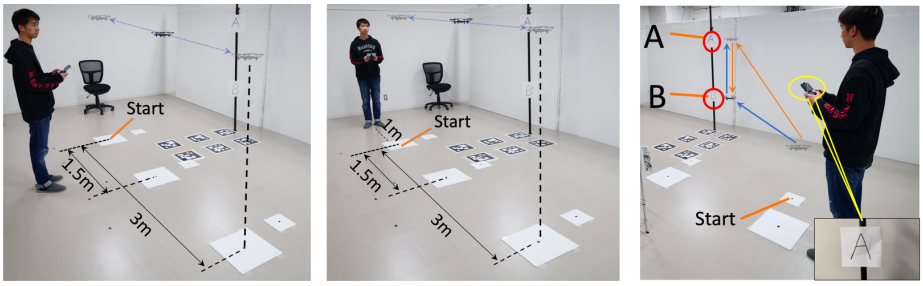
5.1 Participants

We recruited 12 participants (2 females/10 males) from a local university whose ages ranged from 21 to 24 (mean 22.7, SD = 1.2). Two were categorized as experienced drone participants (over 10 hours of drone operation) and others have no pilot experience. Before the experiment, they practiced with the joystick for five minutes because a pre-experiment concluded that five minutes was sufficient to familiarize them with its basic manipulation .

5.2 Setup

Fig. 6 shows the experimental space. We used the OptiTrack motion capture system with a 60-Hz sampling rate [51] to ensure a sufficient capture volume of $3m \times 3m \times 2.7m$. We used an iPhone X with a 5.8-inch touchscreen as the mobile AR device. The participants held a left-hand-throttle-mode joystick with an iPhone X to act as a FPV (Fig. 5). White squared boards on the floor (Fig. 6 (a)(b)) or on the pole (Fig. 6 (c)) indicated the target or the starting volume for the positioning tasks.

Figure 5: Drone with optical markers (top) and joystick (bottom)



(a) Sideways (b) Depth (c) Height

Figure 6: Experimental setup and tasks

5.3 Design

The basic strategy of this study was to investigate how DragFly works for different motion directions (tasks) and positioning difficulties. Thus, we designed three types of tasks: sideways, depth, and height movement. Our main interest was to test sideways and depth movement as they involve spatial perception difficulties in maneuvering. For height movement, which is much simpler, so we used a different setup using the FPV of a Tello drone camera and asked participants to position the drone to capture the target clearly (Fig. 6 (c)). The positioning difficulties were controlled in a common way: by target distances and sizes [47].

Sideways and depth-movement tasks. We set four independent variables: controller (PinpointFly and joystick), movement direction (depth and sideways), target distance (1.5m and 3m), and target size (20cm × 20cm and 40cm × 40cm). Even though we had only two choices for both distance and size due to the given tracking capture volume, these variations allowed us to obtain initial insights on how performance is affected by positioning difficulty and how PinpointFly is effective in an indoor scenario. The two types of movement directions, sideways and depth, clearly assess how the participants' distance perception affects drone positioning. The dependent variables were task completion time, distance error, System Usability Scale (SUS), and NASA Task Load Index (NASA-TLX). Time and error were measured every trial, and SUS and NASA-TLX were measured after finishing the trials of each controller. Each trial was repeated three times, so each participant completed 48 trials: 2 controllers × 2 directions × 2 distances × 2 sizes × 3 repetitions.

Height-movement task. Due to the different spatial perception of the above tasks and the limited room height, we modified our study design accordingly. We provided the FPV from the drone camera to the participants and asked them to change the drone height to clearly capture the target within the FPV. This is still a positioning task, but requires more accuracy. The independent variables were the controller (PinpointFly and joystick) and movement path (A to B and B to A; the distance between A and B is 1.4m). The dependent variables were identical to the previous tasks. Each participant performed eight trials: 2 controllers × 2 motion paths × 2 repetitions.

The presentation orders of the two controllers and the two tasks were counterbalanced among the participants. The orders of the other conditions were randomly presented. The approximate experiment time was about two hours, including many short breaks (e.g., battery changes every 5 min).

5.4 Procedure

Sideways and depth-movement tasks. Figs. 6(a) and (b) illustrate the overviews of the positioning task for the sideways and depth conditions. Two sets of three different-sized white panels were placed on the floor. We chose a simple 40cm × 40cm or 20cm × 20cm set to understand how the target volume affects the positioning. These sizes were decided by the drone size, where positioning a TELLO drone within the smaller panel volume required pinpoint control accuracy and a larger panel would be easier. One white panel was indicated as the start, and the others were used as the target volumes for short and long distance conditions (Figs. 6(a) and (b)). Participants were instructed to move the drone to the target volume (above the target panel) from the starting volume as accurately and quickly as possible. Here they were asked to initially set the drone's height to their eye position and maintained this height during all the trials. This situation was challenging, although it allowed a clear examination of how their depth estimates affected the drone positioning. Their tasks began immediately after the experimenter said "start". When they felt that the drone was positioned within the target volume, they said "OK" and the timer stopped. Here, aural feedback informed them whether the drone was inside the volume.

Height-movement task. The participants were asked to check the characters on the two panels with the drone's camera (Fig. 6(c)). They were asked to control the drone as accurately and rapidly as possible to capture the characters within the FPV to position the drone closer to the target panel. We recorded the time that the drone flew from one place to another until they informed us that the characters had been fully captured by the FPV.

5.5 Results

When we observed crashes in the joystick condition (average 3.2 times/person), the task were repeated. Just one crash occurred in PinpointFly (due to misoperation). The analysis separately used ANOVA and Wilkerson's signed rank test depending on the data normality.

5.5.1 Sideways and depth movement positioning. A four-way ANOVA on the task completion times revealed a main effect of controller ($F_{(1,11)} = 33.4, p < .001, \eta^2 = .752$) and target distance ($F_{(1,11)} = 44.2, p < .001, \eta^2 = .801$). Surprisingly, we did not see any interaction or main effects for the target sizes and movement

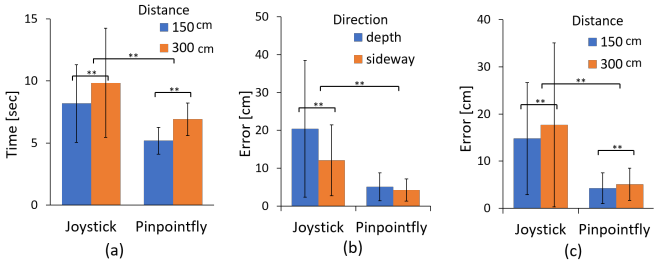


Figure 7: Positioning task results: (a) task completion time for controller vs target distance; (b) distance error for controller vs target direction; (c) distance error for controller vs target direction. Error bars represent standard deviation.

directions ($p > .05$). We expected that smaller targets and depth directions would require more time, but such effects mainly appeared in the error metric (described below). These two main effects are shown in Fig. 7(a) where PinpointFly significantly reduced the positioning times by approximately 37% and 30% compared to the joystick controller for 150 and 300 distances, respectively.

Our four-way ANOVA on the absolute distance error identified main effects for the controller ($F_{(1,11)} = 43.3, p < .001, \eta^2 = .797$), target direction ($F_{(1,11)} = 18.3, p < .001, \eta^2 = .624$), and target distance ($F_{(1,11)} = 10.8, p < .005, \eta^2 = .496$). We also found an interaction between the controller and direction ($F_{(1,11)} = 9.99, p < .005, \eta^2 = 0.476$). Fig. 7(b) shows this interaction. Generally, PinpointFly had less distance error than Joystick, which significantly lowered the positioning accuracy of the depth movement compared to the sideways movement. Fig. 7(c) shows the main effect of the target distance, where a shorter movement ($1.5m$) was more accurate than a longer one ($3m$) for both controllers. Our participants overall underestimated the target distance especially for the depth movements. On average, PinpointFly had $-0.48cm$ (SD = 5.7) distance error, mostly within the target volume, while Joystick made $-11.0cm$ (SD = 19.1) underestimation error, which means the depth perception issue was quite large.

The average SUS scores for PinpointFly and Joystick were 81.3 (acceptable [6]) and 51.8 (low marginal [6]). A Wilcoxon test showed a significant difference between the controllers ($z = 2.94, p < .01, r = .85$). A NASA-TLX test showed that PinpointFly had lower scores than Joystick for all the questions: mental, physical, temporal, performance, effort, and frustration. The averages were 28.7 for PinpointFly, which was a significantly lighter workload than Joystick's 55.4 ($z = 2.74, p < .01, r = .79$).

5.5.2 Height-movement videography. Since this videography task required only drone height control, the joystick also had high performance and no significant differences between controllers. The reason was clear; when the drone flew at a low altitude ($0 - 2m$ in this experiment), no spatial perception issue (e.g., depth estimation) occurred from the participant' perspectives (Fig. 6(c)) and the joystick's action was also quite simple. These trends might change for higher positioning as the perception issue becomes more dominant.

Regarding SUS, PinpointFly and Joystick had 67.1 (high marginal) and 53.5 (low marginal) scores. While there was no significant difference, PinpointFly still had more positive user responses. For the NASA-TLX workload, we identified the same tendency as in the positioning task, and the average score for PinpointFly was 38.4, which is a significantly lower workload than Joystick's 58.3 ($z = 2.82, p < .001, r = .81$).

5.6 Discussion of Study 1

Study 1 showed that PinpointFly increased the users' drone positioning performances in terms of time, error, usability, and workload. The advantages peaked for the depth movements. Due to the long experimental time, we skipped formal post-interviews, but all the subjective data revealed positive comments about PinpointFly. The target size did not affect the task completion time or the control accuracy for either PinpointFly or Joystick. For Joystick, the reason might be its unstable control and the participants' insufficient depth perception. For PinpointFly, the size of the cast shadow representation might not perfectly fit the smaller target area on the AR touchscreen. Our participants included two experienced pilots. Rather than their piloting skill, their spatial perception ability was more dominant.

6 STUDY 2: VIDEOGRAPHY PERFORMANCE

To investigate PinpointFly's total performance for videography tasks, Study 2 used two tasks using FPV: nose-in circle videography and point inspection. We made these choices for the following reasons: (1) both are quite representative motions in inspection contexts; (2) each requires sequential and discrete drone motions; (3) both require accurate drone positioning and smooth path-planning as well as continuous attention to FPV. Thus, a set of the two tasks allowed us to examine the practical effectiveness of each technique: DragFly, PointFly, TrajectoryFly and WaypointFly.

6.1 Participants

We recruited 12 participants (5 females/7 males) from a local university whose ages ranged from 22 to 25 (mean 23.6, SD = 1.2). One was an experienced user and two attended Study 1 and others have no pilot experience. Similar to Study 1, before the experiment, they practiced the joystick manipulation for five minutes.

6.2 Setup

Fig. 8 shows the drone, the joystick controller, and the entire experimental space for Study 2. We used aruco markers placed on the ground as the tracking system whose tracking range was $W : 2m \times D : 2m \times H : 1.7m$. Although this size might be inadequate, it remains sufficient to evaluate precise drone control for indoor scenarios. Drone tracking was achieved at 60 Hz. The entire space was covered by a net for safety and to simulate the walls as obstacles where touching the net equaled a crash. We used an iPad Pro as the mobile AR device because its large screen helped participants focus on both virtual drone manipulation and FPV.

6.3 Design and Procedure

6.3.1 Nose-in circle task: sequential monitoring. Participants performed a simple nose-in circle task with a given radius. The drone is aimed inward at a target around which it pivots with the nose

continually facing inwards as accurately and rapidly as possible using the given controllers. The target was a toy ($20cm \times 40cm \times 20cm$) above two cubes ($20cm \times 20cm \times 20cm$) (Fig. 8(a)). #1 in the figure is the current direction faced by the drone, and #2 denotes its correct direction facing the target. A red circle (radius: $50cm$) on the ground identified the ideal trajectory (Fig. 8(a)). They were instructed to move the drone along this red circle as much as possible. This process helped us measure the drone's position and angle errors. We compared TrajectoryFly and WaypointFly of PinpointFly with the Joystick baseline because they are effective sequential targets for the needs of the tasks. The participants stood at an initial position, but they could freely move, stand, or sit to complete the task as long as the mobile AR device captured the floor markers.

The type of controller was an independent variable (Joystick, TrajectoryFly, WaypointFly), and the dependent variables were the average absolute angle error, the average absolute position error, the task completion time, NASA-TLX and self-made subjective evaluations. The subjective evaluation inquired about intuitiveness, complexity, easiness, integration level, and preference using conventional 5 Likert scales based on SUS questions and our research interests. Each trial was repeated four times (two clockwise and two counterclockwise), and each participant completed 12 trials (3 control methods × 4 repetitions). Similar to Study 1, NASA-TLX and subjective evaluations were performed after the trials of each controller. Unlike Study 1, we used self-made questions instead of SUS because we felt that SUS's complete questions are unsuitable for comparing our two PinpointFly techniques that resemble each other.

6.3.2 Point inspection task: discrete checkpoints. We designed a simple point inspection task as a discrete positioning scenario where six target points were placed differently behind three walls (net) (Fig. 8(b)). We compared DragFly and PointFly with the Joystick baseline because they are considered suitable for such discrete targets tasks. The participants flew the drone to the target points instructed by the experimenter. When they thought the target figure was clearly captured in the FPV, they said "OK", which completed one trial. Then the next target was explained. The order was randomly presented (counter-balanced). The controller was the independent variable (Joystick, DragFly, PointFly). The dependent variables were identical to those in the nose-in-circle task. We recorded 60 data points per participant: 3 controller × 5 target trials × 4 repetitions.

6.4 Results

6.4.1 Nose-in circle task. A one-way ANOVA found a main effect of the controller on the absolute position error ($F_{(2,22)} = 262.16$, $p < .001$, $\eta^2 = .960$). A post-hoc test with Bonferroni correction found that TrajectoryFly and WaypointFly had significantly less error than Joystick ($p < .001$), and we found no significant difference between TrajectoryFly and WaypointFly (Fig. 9(a) left). A similar tendency was obtained for the absolute angle error (Fig. 9(a), middle). We also found a main effect of the controller on the task completion time ($F_{(2,22)} = 42.13$, $p < .001$, $\eta^2 = .793$). A post-hoc test showed that TrajectoryFly and WaypointFly were significantly faster than Joystick ($p < .001$). Unlike the error metrics, TrajectoryFly was significantly faster than WaypointFly ($p < .001$) (Fig. 9(a) right). All

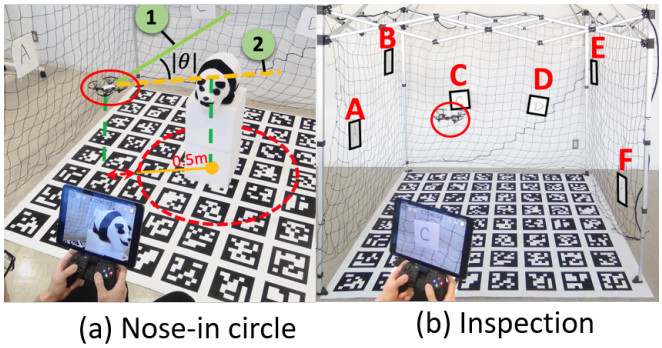


Figure 8: Setup for Study 2

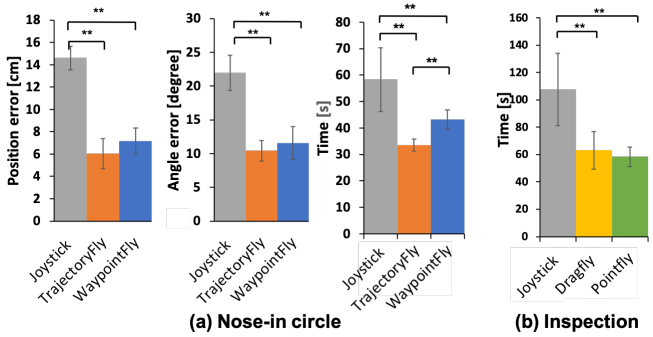


Figure 9: Results of Study 2: (a) absolute average position/angle error and task completion time for nose-in circle task; (b) task completion time for inspection task. Error bars represent standard deviation.

the data clearly suggest that more accurate and faster nose-in circle was achieved by PinpointFly.

We analyzed the NASA-TLX tests using the Freedman test. Our result identified a main effect ($\chi^2 = 22.1, p < .001$), and a post-hoc test showed that the WaypointFly and TrajectoryFly scores were 17.8 and 33.8, which are significantly lower than Joystick's score of 70.7 ($z = 2.98$, $p < .001$, $r = .86$, and $z = 3.17$, $p < .001$, $r = .92$, respectively). The results of our self-made questions that assessed intuitiveness, complexity, easiness, integration, and preferences were quite clear. TrajectoryFly and WaypointFly had considerably higher scores than Joystick because they corresponded to the quantitative time and error metrics. All data can be found in our supplemental materials.

6.4.2 Point inspection task. Our ANOVA found a main effect of controller for task completion time: $F_{(2,22)} = 66.56$, $p < .001$, $\eta^2 = 0.858$. A post-hoc test showed that DragFly and PointFly were significantly faster than Joystick ($p < .001$). However, we did not find any significant difference between DragFly and PointFly (Fig. 9 (b)).

We analyzed the NASA-TLX test using the Freedman test and found a main effect ($\chi^2 = 20.4, p < .001$). Its post-hoc test also

showed that the DragFly and PointFly scores were 33.9 for DragFly and 25.4 which were significantly lower than joystick score of 68.8 ($z = 3.04$, $p < .001$, $r = .88$, and $z = 3.17$, $p < .001$, $r = .92$, respectively). Our questionnaire showed participants' more positive evaluations than Joystick (the detailed data can be found in the material), which also supported the PinpointFly results (Fig. 9(b)).

6.5 Discussion of Study 2

Study 2 showed that PinpointFly's four motion-control techniques were more effective for sequential and discrete tasks than the Joystick baseline in terms of time, error, subjective response, and workload. As shown in Fig. 9(a), TrajectoryFly was faster and more accurate than WaypointFly. Based on the feedback of participants from the subjective questions and NASA-TLX they preferred WaypointFly because it does not require real-time manipulation. From Fig. 9(b), we also found that DragFly was faster than PointFly, even though participants preferred PointFly over DragFly. Based on their feedback, non-expert participants are more comfortable with the two-step procedure for careful control.

7 APPLICATION SCENARIOS

Inspection and troubleshooting are typical discrete targeting scenarios in which the drone's flying path is insignificant. Instead, pilots need to focus on the target's positions and directions. With WaypointFly or PointFly, pilots can quickly and accurately position the drone to its target positions with the desired directions (Fig. 10(a)). If the checkpoints are continuously arranged, TrajectoryFly may also be suited to design the drone's movement path.

Videography is one of the most popular drone applications [62]. PinpointFly supports most fundamental video tasks like selfies or dynamic object videos (e.g., pet's behaviors, etc.) with its DragFly mode. Pilots can also interactively customize the waypoints or paths freely to complete more sensitive video-taking tasks. Fig. 10(c) shows an example where TrajectoryFly is making nose-in circle camera motions around a 3D model. As described in Study 2, WaypointFly is also efficient and sometimes more stable for such videography.

Short-range delivery and assembly are becoming more common [1, 2]. However, non-automated deliveries or aerial constructions are usually dangerous or inaccurate. With PinpointFly, operators can easily deliver small items to workers anywhere as long as their mobile AR and tracking systems are working. Such application might be promising for factories or offices. Fig. 10(c) shows a particular example where PinpointFly allows workers to share tools.

Instruction and guidance Drones with visual display [53, 61, 72] have created a new spectrum of human-drone interaction research. To achieve high-definition and safe interactions, accurate positioning and lower mental workloads are critical. PinpointFly with additional display systems will considerably help operators manage instant advertisements, guide passersby, and instruct busy workers (Fig. 10(d)).

8 GENERAL DISCUSSION

8.1 Finding and Insights

System. Our system worked well and is replicable with free libraries (OpenCV, ARKit). Integrating multiple computation and transmission parts into a running system was challenging. More details are available at the GitHub repository[1].

Findings. Both of our studies show that PinpointFly successfully offers accurate and arbitrary drone control by reducing positioning errors by approximately 50% as well as task completion times by 33% to 45% compared to a baseline joystick. Although the position-control mechanism generally offers more accurate control than a joystick, these strong numerical advantages, which are rather surprising, strongly support our interaction design contributions.

Our participants also mentioned its potential safety, which is additional benefit of a position-control mechanism. Our work and findings, which are generally compatible with prior work (e.g., [32]), considerably expand such prior work by giving practical instances and empirical evidence through two studies. Our three concepts (AR overlay, egocentric, and position control) successfully addressed poor spatial perception, drone-centric, and speed control. However, we only understand the total effects and do not know individual effects and which concept is significant. Follow-up studies are critical.

Usage recommendation. The usage recommendations of the four motion-control techniques were generally validated by our studies. We also found additional recommendations. For example, both DragFly and PointFly can be recommended for discrete targets task; the former is faster, although it causes more fatigue. For sequential target tasks, both TrajectoryFly and WaypointFly are strong options, but WaypointFly is easier to use and TrajectoryFly offers more precise control. In fact, a combination of the four motion-control techniques can cover most of the applications. For spraying pesticides, pilots might use WaypointFly to cover most of the area and then use PointFly for the unfinished area. Another example is drone racing, where DragFly and TrajectoryFly might be combined to make more aggressive drone movements.

Target Users. PinpointFly is mainly designed for novices. We incorporated the participation of experienced users to investigate whether the interface is also appropriate for them. Our results showed a very similar trend for both cohorts. Interviews with expert users revealed that if they did not precisely know the spatial relations of the target and the drone, their piloting skills had problems or made position mistakes, suggesting that in some cases, more experienced users can also benefit from PinpointFly.

8.2 Limitations and Improvements

Operation area expansion. PinpointFly workspace is currently limited because we assumed VLOS operations. An interesting solution is to use the hybrid position and a rate-control interface (e.g., [7]). For example, the edges of touchscreen/slider bars are assigned to make fast rate control while their central area remains for accurate position control, where pilots can smoothly switch between position and speed controls without special commands.

Comparison with other egocentric approaches. We did not compare our work with other egocentric approaches because most of

[1] https://github.com/find1dream/PinpointFly

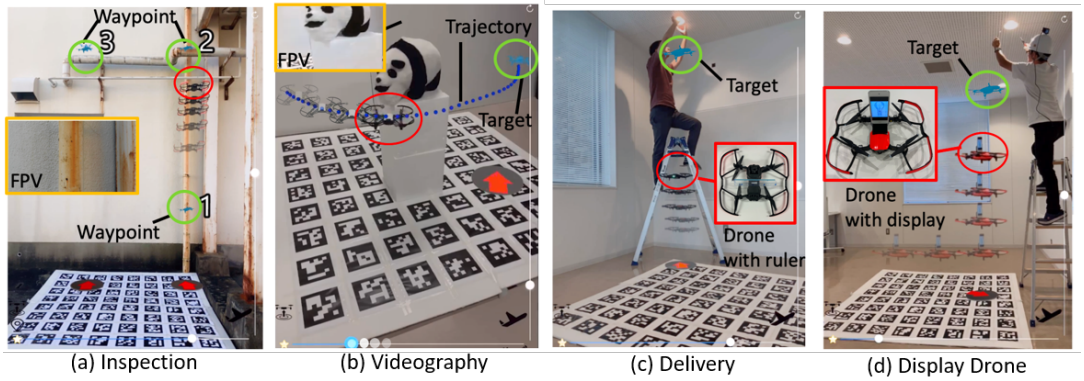


Figure 10: Application examples: (a) troubleshooting with WaypointFly, (b) videography with TrajectoryFly, (c) drone with ruler, (d) drone with display

those systems were designed for different purposes and are not commercially available. For example, prior FPV-based interfaces or trajectory generators are inconsistent with our assumptions, setup and tasks. Rather than performing incomplete validation through comparison with their replicas, our strategy was to provide broader and understandable performance metrics of PinpontFly by comparing it with the most common joystick controller. Nonetheless, we acknowledge that the comparison with joystick is not comprehensive without taking experience level into account as the joystick performance changes drastically with experience level. A follow-up test with pilots of different flight hours would be necessary. We also note that other mobile AR-based egocentric interfaces using gestures (e.g. [38, 74]) support different kinds of motion inputs, which are complementary with ours. It would be beneficial to explore how these NUIs can be integrated with our approach to achieve a better egocentric drone interface.

Cast shadow and slider bar: Although most participants performed positively with the cast shadow manipulations, they sometimes had difficulty when moving long distances (e.g., seven meters) as the cast shadow's side decreases. The size of the cast shadow should be dynamically adjustable to maintain visibility. While most participants understood the slider bar interface and its rationale, a tablet may better support multi-finger gestures in larger visual and motor spaces.

Object occlusion: Our interface does not currently address the occlusion problem caused when a cast shadow moves behind objects. However, a temporary solution includes additional visual guidance for pilots to move around the target to reduce visually occluded areas. We could also employ an approach using both FPV and TPV to construct 3D mappings of the environment with AR and computer vision technologies.

Mobile AR camera: Our camera system sometimes lost floor tracking due to light reflection from the floor. Solutions include adding optical filters to the camera. While a flat floor was used in this study, when a room has physical obstacles, the drone's visibility issue could be challenging.

Tracking system: Similar to many interactive drone researches [5, 18, 28, 42, 45, 74, 77, 78], the necessity of a tracking system might be a limitation. We are optimistic about this because, in the near future, inside-out tracking by on-drone sensors will address this issue. PinpointFly itself is designed independently from particular tracking systems to further support various tracking systems, allowing users to select the best one based on the given situations.

9 CONCLUSION

We proposed PinpointFly, an egocentric drone interface that allows pilots to accurately and arbitrarily position and rotate a flying drone by position-control interactions on a see-through mobile AR where the drone position and direction are visualized with a virtual cast shadow. We designed four fundamental motion-control techniques for PinpointFly and implemented its proof-of-concept prototype using off-the-shelf devices, including a motion tracker, see-through AR technology, and a small, programmable drone. From two user studies with indoor positioning and videography scenarios, we found that PinpointFly was more efficient, accurate, and induced less workload than a conventional joystick speed-control method. Future work will improve the interface's design (cast shadow size, shape, etc.) and explore new synergy with other gestural approach and tracking systems.

REFERENCES

[1] Frederico Augugliaro, Sergei Lupashin, Michael Hamer, Cason Male, Markus Hehn, Mark W. Mueller, Jan S. Willmann, Fabio Gramazio, Matthias Kohler, and Raffaello D'Andrea. 2014. The Flight Assembled Architecture Installation: Cooperative Construction with Flying Machines. *IEEE Control Systems Magazine* 34, 4 (Aug 2014), 46–64. https://doi.org/10.1109/MCS.2014.2320359

[2] Federico Augugliaro, Ammar Mirjan, Fabio Gramazio, Matthias Kohler, and Raffaello D'Andrea. 2013. Building Tensile Structures with Flying Machines. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*. 3487–3492. https://doi.org/10.1109/IROS.2013.6696853

[3] Mehmet Aydin Baytas, Damla Çay, Yuchong Zhang, Mohammad Obaid, Asim Evren Yantaç, and Morten Fjeld. 2019. The Design of Social Drones: A Review of Studies on Autonomous Flyers in Inhabited Environments. In *Proceedings of CHI Conference on Human Factors in Computing Systems*. Paper No. 250, 13pages. https://doi.org/10.1145/3290605.3300480

[4] Rogerio Bonatti, Yanfu Zhang, Sanjiban Choudhury, Wenshan Wang, and Sebastian Scherer. 2018. Autonomous drone cinematographer: Using artistic principles to create smooth, safe, occlusion-free trajectories for aerial filming. *arXiv* (aug 2018). arXiv:1808.09563 https://arxiv.org/abs/1808.09563

[5] Sean Braley, Calvin Rubens, Timothy Merritt, and Roel Vertegaal. 2018. GridDrones: A Self-Levitating Physical Voxel Lattice for Interactive 3D Surface Deformations. In *Proceedings of ACM Symposium on User Interface Software and Technology*. 87–98. https://doi.org/10.1145/3242587.3242658

[6] John Brooke. 1996. SUS: A Quick and Dirty Usability Scale. *Usability Evaluation in Industry* (Nov 1996), 189–194.

[7] Géry Casiez, Daniel Vogel, Qing Pan, and Christophe Chaillou. 2007. RubberEdge: Reducing Clutching by Combining Position and Rate Control with Elastic Feedback. In *Proceedings of ACM Symposium on User Interface Software and Technology*. 129–138. https://doi.org/10.1145/1294211.1294234

[8] Jessica R. Cauchard, Jane L. E, Kevin Y. Zhai, and James A. Landay. 2015. Drone & Me: An Exploration Into Natural Human-Drone Interaction. In *Proceedings of ACM International Joint Conference on Pervasive and Ubiquitous Computing*. 361–365. https://doi.org/10.1145/2750858.2805823

[9] Meghan Chandarana, Erica L. Meszaros, Anna Trujillo, and Bonnie Danette Allen. 2017. "Fly Like This": Natural Language Interfaces for UAV Mission Planning. In *Proceedings of International Conference on Advances in Computer-Human Interactions*. https://ntrs.nasa.gov/search.jsp?R=20170002593

[10] Linfeng Chen, Kazuki Takashima, Kazuyuki Fujita, and Yoshifumi Kitamura. 2019. PinpointFly: An Egocentric Position-Pointing Drone Interface Using Mobile AR. In *Proceedings of SIGGRAPH Asia 2019 Emerging Technologies*. 34–35. https://doi.org/10.1145/3355049.3360534

[11] Yu-An Chen, Mike Y. Chen, Te-Yen Wu, Tim Chang, Jun You Liu, Yuan-Chang Hsieh, Leon Yulun Hsu, Ming-Wei Hsu, Paul Taele, and Neng-Hao Yu. 2018. ARPilot: Designing and Investigating AR Shooting Interfaces on Mobile Devices for Drone Videography. In *Proceedings of International Conference on Human-Computer Interaction with Mobile Devices and Services*. 42:1–42:8. https://doi.org/10.1145/3229434.3229475

[12] Kwangsu Cho, Minhee Cho, and Jongwoo Jeon. 2016. Fly a Drone Safely: Evaluation of an Embodied Egocentric Drone Controller Interface. *Interacting with Computers* 29, 3 (Sep 2016), 345–354. https://doi.org/10.1093/iwc/iww027

[13] Zeynep Cipiloglu, Abdullah Bulbul, and Tolga Capin. 2010. A Framework for Enhancing Depth Perception in Computer Graphics. In *Proceedings of Symposium on Applied Perception in Graphics and Visualization*. 141–148. https://doi.org/10.1145/1836248.1836276

[14] Edward Companik, Michael Gravier, and Martin Farris. 2018. Feasibility of Warehouse Drone Adoption and Implementation. *Journal of Transportation Management* 28, 2 (May 2018), 33–50. https://doi.org/10.22237/jotm/1541030640

[15] Tiago Ogioni Costalonga, Lucas Mendes Avila, Luis Muniz, and Alexandre Santos Brandao. 2014. Gesture-Based Controllers to Guide a Quadrotor Using Kinect Sensor. In *Proceedings of Joint Conference on Robotics: SBR-LARS Robotics Symposium and Robocontrol*. 109–112. https://doi.org/10.1109/SBR.LARS.Robocontrol.2014.28

[16] Catherine Diaz, Michael Walker, Danielle Albers Szafir, and Daniel Szafir. 2017. Designing for Depth Perceptions in Augmented Reality. In *Proceedings of IEEE International Symposium on Mixed and Augmented Reality*. 111–122. https://doi.org/10.1109/ISMAR.2017.28

[17] Tinglin Duan, Parinya Punpongsanon, Daisuke Iwai, and Kosuke Sato. 2018. FlyingHand: Extending the Range of Haptic Feedback on Virtual Hand Using Drone-based Object Recognition. In *Proceedings of the SIGGRAPH Asia 2018 Technical Briefs*. Article 28, 4 pages. https://doi.org/10.1145/3283254.3283258

[18] Okan Erat, Werner Alexander Isop, Denis Kalkofen, and Dieter Schmalstieg. 2018. Drone-Augmented Human Vision: Exocentric Control for Drones Exploring Hidden Areas. *IEEE Transactions on Visualization and Computer Graphics* 24, 4 (Apr 2018), 1437–1446. https://doi.org/10.1109/TVCG.2018.2794058

[19] Huang. C. Fang, Soh. K. Ong, and Andrew. Y. C. Nee. 2014. Novel AR-based Interface for Human-Robot Interaction and Visualization. *Advances in Manufacturing* 2, 4 (Dec 2014), 275–288. https://doi.org/10.1007/s40436-014-0087-9

[20] Ramon A.Suarez Fernandez, Jose Luis Sanchez-Lopez, Carlos Sampedro, Hriday Bavle, Martin Molina, and Pascual Campoy. 2016. Natural User Interfaces for Human-drone Multi-modal Interaction. *International Conference on Unmanned Aircraft Systems* (2016), 1013–1022. https://doi.org/10.1109/ICUAS.2016.7502665

[21] Markus Funk. 2018. Human-Drone Interaction: Let's Get Ready for Flying User Interfaces! *Interactions* 25, 3 (April 2018), 78–81. https://doi.org/10.1145/3194317

[22] Quentin Galvane, Christophe Lino, Marc Christie, Julien Fleureau, Fabien Servant, Fran¸ois-louis Tariolle, and Philippe Guillotel. 2018. Directing Cinematographic Drones. *ACM Transactions on Graphics* 37, 3 (Jul 2018), 1–18. https://doi.org/10.1145/3181975

[23] Sergio Garrido-Jurado, Rafael Muñoz-Salinas, Fra cisco J. Madrid-Cuevas, and Rafael Medina-Carnicer. 2016. Generation of Fiducial Marker Dictionaries Using Mixed Integer Linear Programming. *Pattern Recognition* 51 (Mar 2016), 481–491. https://doi.org/10.1016/J.PATCOG.2015.09.023

[24] S. Garrido-Jurado, R. Muñoz-Salinas, F.J. Madrid-Cuevas, and M.J. Marín-Jiménez. 2014. Automatic Generation and Detection of Highly Reliable Fiducial Markers under Occlusion. *Pattern Recognition* 47 (June 2014), 2280 – 2292. https://doi.org/10.1016/j.patcog.2014.01.005

[25] Christoph Gebhardt, Benjamin Hepp, Tobias Nägeli, Stefan Stevšić, and Otmar Hilliges. 2016. Airways: Optimization-Based Planning of Quadrotor Trajectories according to High-Level User Goals. *Proceedings of CHI Conference on Human Factors in Computing Systems* (2016), 2508–2519. https://doi.org/10.1145/2858036.2858353

[26] Christoph Gebhardt and Otmar Hilliges. 2018. WYFIWYG: Investigating Effective User Support in Aerial Videography. arXiv:1801.05972 [cs.HC]

[27] Christoph Gebhardt, Stefan Stevšić, and Otmar Hilliges. 2018. Optimizing for aesthetically pleasing qadrotor camera motion. *ACM Transactions on Graphics* 37, 4 (jul 2018), 1–11. https://doi.org/10.1145/3197517.3201390

[28] Antonio Gomes, Calvin Rubens, Sean Braley, and Roel Vertegaal. 2016. BitDrones: Towards Using 3D Nanocopter Displays As Interactive Self-Levitating Programmable Matter. In *Proceedings of CHI Conference on Human Factors in Computing Systems*. 770–780. https://doi.org/10.1145/2858036.2858519

[29] Yazou Han. 2019. TelloPy. Retrieved September 17, 2020 from https://github.com/hanyazou/TelloPy

[30] Sunao Hashimoto, Masahiko Ishida, Akihiko Inami, and Takeo Igarashi. 2011. TouchMe : An Augmented Reality Based Remote Robot Manipulation. In *Proceedngis of International Conference on Artificial Reality and Telexistence*. 28–30. https://ci.nii.ac.jp/naid/10030783278/

[31] Hooman Hedayati, Michael Walker, and Daniel Szafir. 2018. Improving Collocated Robot Teleoperation with Augmented Reality. In *Proceedings of ACM/IEEE International Conference on Human-Robot Interaction*. 78–86. https://doi.org/10.1145/3171221.3171251

[32] Keita Higuchi, Katsuya Fujii, and Jun Rekimoto. 2013. Flying Head: A Head-synchronization Mechanism for Flying Telepresence. In *Proceedings of International Conference on Artificial Reality and Telexistence*. 28–34. https://doi.org/10.1109/ICAT.2013.6728902

[33] Matthias Hoppe, Pascal Knierim, Thomas Kosch, Markus Funk, Lauren Futami, Stefan Schneegass, Niels Henze, Albrecht Schmidt, and Tonja Machulla. 2018. VRHapticDrones: Providing Haptics in Virtual Reality Through Quadcopters. In *Proceedings of International Conference on Mobile and Ubiquitous Multimedia*. 7–18. https://doi.org/10.1145/3282894.3282898

[34] Baichuan Huang, Deniz Bayazit, Daniel Ullman, Nakul Gopalan, and Stefanie Tellex. 2019. Flight, Camera, Action! Using Natural Language and Mixed Reality to Control a Drone. In *Proceedings of IEEE International Conference on Robotics and Automatio*. 6949–6956. https://doi.org/10.1109/ICRA.2019.8794200

[35] Javier Irizarry, Masoud Gheisari, and Bruce N. Walker. 2012. Usability Assessment of Drone Technology as Safety Inspection Tools. *Journal of Information Technology in Construction* (Sep 2012).

[36] Matthew R. Johnson. 2011. Spatial Cognition, Spatial Perception. *The Yale Journal of Biology and Medicine* 84, 1 (Mar 2011), 63–63. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3064250/

[37] Niels Joubert, Mike Roberts, Anh Truong, Floraine Berthouzoz, and Pat Hanrahan. 2015. An Interactive Tool for Designing Quadrotor Camera Shots. *ACM Transactions on Graphics* 34, 6 (Oct 2015), 238:1–238:11. https://doi.org/10.1145/2816795.2818106

[38] Shunichi Kasahara, Ryuma Niiyama, Valentin Heun, and Hiroshi Ishii. 2013. exTouch: Spatially-Aware Embodied Manipulation of Actuated Objects Mediated by Augmented Reality. In *Proceedings of International Conference on Tangible, Embedded and Embodied Interaction*. 223–226. https://doi.org/10.1145/2460625.2460661

[39] Kiam Heong Ang, G. Chong, and Yun Li. 2005. PID Control System Analysis, Design, and Technology. *IEEE Transactions on Control Systems Technology* 13, 4 (Jul 2005), 559–576. https://doi.org/10.1109/TCST.2005.847331

[40] Pascal Knierim, Steffen Maurer, Katrin Wolf, and Markus Funk. 2018. Quadcopter-Projected In-Situ Navigation Cues for Improved Location Awareness. In *Proceedings of CHI Conference on Human Factors in Computing Systems*. Paper No: 433, 6 pages. https://doi.org/10.1145/3173574.3174007

[41] Przemyslaw M. Kornatowski, Anand Bhaskaran, Gregoire M. Heitz, Stefano Mintchev, and Dario Floreano. 2018. Last-Centimeter Personal Drone Delivery: Field Deployment and User Interaction. *IEEE Robotics and Automation Letters* 3, 4 (Oct 2018), 3813–3820. https://doi.org/10.1109/LRA.2018.2856282

[42] Thomas Kosch, Markus Funk, Daniel Vietz, Marc Weise, Tamara Müller, and Albrecht Schmidt. 2018. DroneCTRL: A Tangible Remote Input Control for Quadcopters. In *Adjunct Proceedings of ACM Symposium on User Interface Software and Technology*. 120–122. https://doi.org/10.1145/3266037.3266121

[43] Ziquan Lan, Mohit Shridhar, David Hsu, and Shengdong Zhao. 2017. XPose: Reinventing User Interaction with Flying Cameras. In *Proceedings of Robotics: Science and Systems XIII*. https://doi.org/10.15607/RSS.2017.XIII.006

[44] Pierre Latteur, Sebastien Goessens, Jean-Sébastien Breton, Justin Leplat, Zhao Ma, and Caitlin Mueller. 2015. Drone-Based Additive Manufacturing of Architectural Structures. In *Proceedings of International Association for Shell and Spatial Structures Symposium*. 1–12.

[45] Sang-won Leigh, Harshit Agrawal, and Pattie Maes. 2016. A Flying Pantograph. *Proceedings of International Conference on Tangible, Embedded, and Embodied Interaction* (2016), 653–657. https://doi.org/10.1145/2839462.2856347

[46] Jiannan Li, Ravin Balakrishnan, and Tovi Grossman. 2020. StarHopper: A Touch Interface for Remote Object-Centric Drone Navigation. In *Proceedings of Graphics Interface*. 317 – 326. https://doi.org/10.20380/GI2020.32

[47] I. Scott MacKenzie. 1992. Fitts' Law as a Research and Design Tool in Human-Computer Interaction. *Human-Computer Interaction* 7, 1 (1992), 91–139. https://doi.org/10.1207/s15327051hci0701_3

[48] Ahmed Mashood, Hassan Noura, Imad Jawhar, and Nader Mohamed. 2015. A Gesture Based Kinect for Quadrotor Control. In *Proceedings of International Conference on Information and Communication Technology Research*. 298–301. https://doi.org/10.1109/ICTRC.2015.7156481

[49] Nathan Michael, Shaojie Shen, Kartik Mohta, Yash Mulgaonkar, Vijay Kumar, Keiji Nagatani, Yoshito Okada, Seiga Kiribayashi, Kazuki Otake, Kazuya Yoshida, Kazunori Ohno, Eijiro Takeuchi, and Satoshi Tadokoro. 2012. Collaborative Mapping of an Earthquake-Damaged Building via Ground and Aerial Robots. *Jprnal of Field Robotics* 29, 5 (Sept. 2012), 832–841. https://doi.org/10.1002/rob.21436

[50] Tobias Nägeli, Lukas Meier, Alexander Domahidi, Javier Alonso-Mora, and Otmar Hilliges. 2017. Real-time planning for automated multi-view drone cinematography. *ACM Transactions on Graphics* 36, 4 (Jul 2017), 1–10. https://doi.org/10.1145/3072959.3073712

[51] NaturalPoint. Accessed: 2019. Motion Capture Systems - OptiTrack Webpage. http://precog.iiitd.edu.in/people/anupama.

[52] Wai. S. Ng and Ehud. Sharlin. 2011. Collocated interaction with flying robots. In *Proceedings of IEEE International Symposium on Robot and Human Interactive Communication*. 143–149. https://doi.org/10.1109/ROMAN.2011.6005280

[53] Hiroki Nozaki. 2014. Flying Display: A Movable Display Pairing Projector and Screen in the Air. In *Extended Abstracts on Human Factors in Computing Systems*. 909–914. https://doi.org/10.1145/2559206.2579410

[54] Thomas Prevot, Joseph Rios, Parimal Kopardekar, John Robinson III, Marcus Johnson, and Jaewoo Jung. 2016. UAS Traffic Management (UTM) Concept of Operations to Safely Enable Low Altitude Flight Operations. In *AIAA Aviation Technology, Integration, and Operations Conference*.

[55] Mike Roberts and Pat Hanrahan. 2016. Generating Dynamically Feasible Trajectories for Quadrotor Cameras. *ACM Transactions on Graphics* 35, 4 (Jul 2016), 61:1–61:11. https://doi.org/10.1145/2897824.2925980

[56] Francisco J. Romero-Ramirez, Rafael Muñoz-Salinas, and Rafael Medina-Carnicer. 2018. Speeded Up Detection of Squared Fiducial Markers. *Image and Vision Computing* 76 (aug 2018), 38–47. https://doi.org/10.1016/J.IMAVIS.2018.05.004

[57] Daisuke Sakamoto, Koichiro Honda, Masahiko Inami, and Takeo Igarashi. 2009. Sketch and Run: A Stroke-based Interface for Home Robots. In *Proceedings of CHI Conference on Human Factors in Computing Systems*. 197–200. https://doi.org/10.1145/1518701.1518733

[58] Andrea Sanna, Fabrizio Lamberti, Gianluca Paravati, and Federico Manuri. 2013. A Kinect-based natural interface for quadrotor control. *Entertainment Computing* 4, 3 (Aug 2013), 179–186. https://doi.org/10.1016/J.ENTCOM.2013.01.001

[59] Ayanava Sarkar, Ketul Arvindbhai Patel, R. K. Ganesh Ram, and Geet Krishna Capoor. 2016. Gesture Control of Drone Using a Motion Controller. In *Proceedings of International Conference on Industrial Informatics and Computer Systems*. 1–5. https://doi.org/10.1109/ICCSII.2016.7462401

[60] Gerhard Schall, Sebastian Junghanns, and Dieter Schmalstieg. 2010. Vidente-3D Visualization of Underground Infrastructure using Handheld Augmented Reality. In *Geohydroinformatics : Integrating GIS and Water Engineering*.

[61] Jürgen Scheible and Markus Funk. 2016. In-situ-displaydrone: Facilitating Colocated Interactive Experiences via a Flying Screen. In *Proceedings of ACM International Symposium on Pervasive Displays*. 251–252. https://doi.org/10.1145/2914920.2940334

[62] Hazim Shakhatreh, Ahmad H. Sawalmeh, Ala Al-Fuqaha, Zuochao Dou, Eyad Almaita, Issa Khalil, Noor Shamsiah Othman, Abdallah Khreishah, and Mohsen Guizani. 2019. Unmanned Aerial Vehicles (UAVs): A Survey on Civil Applications and Key Research Challenges. *IEEE Access* 7 (2019), 48572–48634. https://doi.org/10.1109/access.2019.2909530

[63] Sang-Yun Shin, Yong-Won Kang, and Yong-Guk Kim. 2019. Hand Gesture-based Wearable Human-Drone Interface for Intuitive Movement Control. In *Proceedings of IEEE International Conference on Consumer Electronics*. 1–6. https://doi.org/10.1109/ICCE.2019.8662106

[64] Junwei Sun, Wolfgang Stuerzlinger, and Dmitri Shuralyov. 2016. Shift-Sliding and Depth-Pop for 3D Positioning. In *Proceedings of Symposium on Spatial User Interaction*. 165–165. https://doi.org/10.1145/2983310.2991067

[65] Ting Sun, Shengyi Nie, Dit-Yan Yeung, and Shaojie Shen. 2017. Gesture-based Piloting of an Aerial Robot using Monocular Vision. In *Proceedings of IEEE International Conference on Robotics and Automation*. 5913–5920. https://doi.org/10.1109/ICRA.2017.7989696

[66] Ryotaro Temma, Kazuki Takashima, Kazuyuki Fujita, Koh Sueda, and Yoshifumi Kitamura. 2019. Third-Person Piloting: Increasing Situational Awareness Using a Spatially Coupled Second Drone. In *Proceedings of ACM Symposium on User Interface Software and Technology*. 507–519. https://doi.org/10.1145/3332165.3347953

[67] Thomas Seifried, Michael Haller, Stacey D. Scott, Florian Perteneder, Christian Rendl, Daisuke Sakamoto, and Masahiko Inami. 2009. CRISTAL: A Collaborative Home Media and Device Controller Based on a Multi-touch Display. In *Proceedings of ACM International Conference on Interactive Tabletops and Surfaces*. 33–40.

[68] Sébastien Thon, Dominique Serena-Allier, Celine Salvetat, and Francoise Lacotte. 2013. Flying a Drone in a Museum: An Augmented-reality Cultural Serious Game in Provence. In *Proceedings of Digital Heritage International Congress*. 669–676. https://doi.org/10.1109/DigitalHeritage.2013.6744834

[69] Anastasia Uryasheva, Mikhail Kulbeda, Nikita Rodichenko, and Dzmitry Tsetserukou. 2019. DroneGraffiti: Autonomous multi-UAV Spray Painting. In *Proceedings of ACM SIGGRAPH 2019 Studio*. Article 4, 2 pages. https://doi.org/10.1145/3306306.3328000

[70] Jason. Wither and Tobias. Hollerer. 2005. Pictorial Depth Cues for Outdoor Augmented Reality. In *Proceedings of IEEE International Symposium on Wearable Computers*. 92–99. https://doi.org/10.1109/ISWC.2005.41

[71] Ke Xie, Hao Yang, Shengqiu Huang, Dani Lischinski, Marc Christie, Kai Xu, Minglun Gong, Daniel Cohen-Or, and Hui Huang. 2018. Creating and chaining camera moves for qadrotor videography. *ACM Transactions on Graphics* 37, 4 (Jul 2018), 1–13. https://doi.org/10.1145/3197517.3201284

[72] Wataru Yamada, Kazuhiro Yamada, Hiroyuki Manabe, and Daizo Ikeda. 2017. iSphere: Self-Luminous Spherical Drone Display. In *Proceedings of ACM Symposium on User Interface Software and Technology*. 635–643. https://doi.org/10.1145/3126594.3126631

[73] Hao Yang, Ke Xie, Shengqiu Huang, and Hui Huang. 2018. Uncut Aerial Video via a Single Sketch. *Computer Graphics Forum* 37, 7 (oct 2018), 191–199. https://doi.org/10.1111/cgf.13559

[74] Kazuya Yonezawa and Takefumi Ogawa. 2015. Flying Robot Manipulation System Using a Virtual Plane. In *Proceedings of IEEE Virtual Reality Conference*. 313–314. https://doi.org/10.1109/VR.2015.7223421

[75] Shumin Zhai. 1995. *Human Performance in Six Degree of Freedom Input Control*. Ph.D. Dissertation. University of Toronto.

[76] Shumin Zhai. 1998. User Performance in Relation to 3D Input Device Design. *ACM SIGGRAPH Computer Graphics* 32 (Nov 1998). https://doi.org/10.1145/307710.307728

[77] Xujing Zhang, Sean Braley, Calvin Rubens, Timothy Merritt, and Roel Vertegaal. 2019. LightBee: A Self-Levitating Light Field Display for Hologrammatic Telepresence. In *Proceedings of CHI Conference on Human Factors in Computing Systems*. Article 12, 10 pages. https://doi.org/10.1145/3290605.3300242

[78] Stefanie Zollmann, Christof Hoppe, Tobias Langlotz, and Gerhard Reitmayr. 2014. FlyAR: Augmented Reality Supported Micro Aerial Vehicle Navigation. *IEEE Transactions on Visualization and Computer Graphics* 20, 4 (Apr 2014), 560–568. https://doi.org/10.1109/TVCG.2014.24

[79] Stefanie Zollmann, Gerhard Schall, Sebastian Junghanns, and Gerhard Reitmayr. 2012. Comprehensible and Interactive Visualizations of GIS Data in Augmented Reality. In *Advances in Visual Computing*. 675–685. https://doi.org/10.1007/978-3-642-33179-4_64