Performance of a Networked Human-Drone Team: Command Response and Interaction Effects

Alexander Freistetter
JKU Linz
Austria
k01455602@students.jku.at

Manuela Pollak
JKU Linz
Austria
manuela.pollak@jku.at

Karin Anna Hummel
JKU Linz
Austria
karin_anna.hummel@jku.at

ABSTRACT

Drones are flying machines which can be employed for sensing in different altitudes. While past research has made remarkable progress in providing autonomous drone navigation, the integration of humans and drones to collaboratively solve a task poses new challenges. Studies are needed to understand the crucial features of systems that integrate autonomous drone behavior and human-drone interactions. In this work, we study the performance of such a networked system for human-drone teams. We make use of a prototype consisting of an off-the-shelf drone with an on-board camera, autopilot, and Wi-Fi connectivity, and a custom smartphone app. The app supports the drone and performs vision-based autonomous navigation control and human gesture recognition, and provides a graphical user interface for the human. To study the benefit of collaboration, a concrete indoor use case is selected, namely bookshelf inventory. The drone scans the books in a shelf, while the human acts as a supervisor who may temporarily take over control to reposition the drone. In a measurement study, we investigate the command response delays of the networked system, effects of human intervention, and the performance of gesture-based interaction.

CCS CONCEPTS

• **Networks**; • **Computer systems organization** → **Distributed architectures**; **Robotics**;

KEYWORDS

Human-drone teaming; unmanned aerial vehicles; distributed architecture; gesture-based interaction

ACM Reference Format:
Alexander Freistetter, Manuela Pollak, and Karin Anna Hummel. 2020. Performance of a Networked Human-Drone Team: Command Response and Interaction Effects. In *The 6th ACM Workshop on Micro Aerial Vehicle Networks, Systems, and Applications (DroNet'20), June 19, 2020, Toronto, ON, Canada.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3396864.3399703

DroNet'20, June 19, 2020, Toronto, ON, Canada

ACM ISBN 978-1-4503-8010-2/20/06…$15.00
https://doi.org/10.1145/3396864.3399703

1 INTRODUCTION

Drones, aka unmanned aerial vehicles (UAVs), can be employed in various civilian application fields such as 3D cartography, surveillance, and entertainment. Mainly the ability to take pictures and videos from the air by on-board cameras is utilized, but also carrying small parcels and scanning other features by on-board sensors is feasible. In particular video streaming requires sufficient data rates, which can be provided by 4G/5G or Wi-Fi networks [1].

Autonomous drones are flying robots that can navigate and operate without human intervention. In particular in indoor scenarios, vision-based navigation is often chosen for its general applicability [10]. Besides autonomous operation, real deployment of drones will likely require assistance for teams of humans and drones. Supporting technologies need to address not only control, but also communication, and human-drone interaction. Among the technical challenges are the general timing behavior of the networked control system and the performance effects caused when integrating human-drone interaction with traditional autonomous drone operation. When the technical challenges are sufficiently solved, applications of human-drone teams can be envisioned in various use cases, for example, guiding through a public office building, cooperating during construction work, and jointly controlling traffic or large events.

In this paper, we contribute to the understanding of human-drone systems by a study of major timing characteristics. Hereby, we focus on the particularities of the system, that are its networked nature and the integration of autonomous drone operation and human-drone interaction. We use our custom implementation of a distributed human-drone system that consists of a drone equipped with an on-board first person view (FPV) camera and a smartphone app, connected through Wi-Fi. While the drone's embedded computer executes basic piloting and provides a video stream, the smartphone app implements the more complicated recognition and navigation control logic and provides user interfaces. To quantify the performance of the human-drone team, we select a use case where the drone operates most of the time autonomously. The human takes the role of a supervisor and supporter of the drone.

1.1 Use Case Bookshelf Inventory

The selected use case is bookshelf inventory, which is a common repetitive task in libraries and warehouses. The drone's task is to scan a bookshelf by counting the number of books in the shelf and capturing book titles autonomously using an on-board camera. This task requires image and text recognition and vision-based navigation. The navigation algorithm follows a simple simultaneous localization and mapping (SLAM) based approach, where the bookshelf is the area of interest. The books need to be scanned,

while at the same time, the drone's position relative to the books is estimated and used for navigation. First, the drone counts the number of books in the whole shelf. Then, the drone starts to scan the shelf layer by layer. After the last layer has been scanned, the task is finished and a list of found books is stored. During the autonomous flight, the human teammate can follow the status of the task presented on a smartphone screen and intervene at any point in time to take over control and to navigate the drone, e.g., in case it lost the bookshelf.

1.2 Contribution

The focus of this paper rests on studying performance characteristics of the networked human-drone team. We conduct experiments in a small testbed consisting of a micro drone, the Parrot Mambo, and a smartphone app which has been presented in [5, 6] and is extended for this purpose. In detail, we make the following contributions:

- From a functional perspective, we extend previous work by adding hand gesture recognition to interact naturally with the drone integrated in a distributed system architecture. Related approaches to support human-drone teams are summarized in Section 2, our system and hand gesture recognition are described in Section 3.
- We present insights about the performance of the human-drone system by studying: (i) the command response time, (ii) the flight-time and scan performance of autonomous drone operation compared to semi-autonomous behavior with the human in-the-loop, and (iii) the flight-time measured for gesture-based interaction compared to traditional steering with smartphone buttons. The experiments and results are described in Section 4.

2 RELATED WORK

Human-drone teaming systems require control modules that realize autonomous drone behavior, sensors, communication technologies, and means to coordinate human and drone activities. Intuitive and effective user interfaces for the human are further key prerequisites.

A major design decision relates to whether control is implemented locally on the drone or remotely on a more powerful computer [3, 6]. While on-board control is faster, distributed control allows to move computationally expensive algorithms to a more powerful remote computer or cloud, which may also be economically beneficial [7]. Such a distributed architecture can be implemented by making use of existing technologies, including middleware and communication support for robots and machines, such as the Robotic Operating System (ROS),[1] the Message Queuing Telemetry Transport (MQTT) messaging protocol,[2] or light-weight TCP/UDP socket communication. However, studies are needed to investigate whether a distributed architecture meets the real-time requirements of the control logic and which parts of the computation may be distributed at all.

A distributed system is also beneficial for future integration of multiple drones or swarms. The need to stay in control of such drone swarms is discussed in [9]. Decentralized formation control of a group of multi-rotor unmanned aerial vehicles is one of the traditional research fields in aerial swarm robotics. Algorithms typically aim at a stable and balanced formation of drones such as shapes known from bird flocking behavior, etc. Yet, for a human operator it is difficult to cooperate with multiple drones at the same time as this situation exceeds the human's sensory capabilities. A first step to characterize the human's capabilities when cooperating with (multiple) drones is to monitor physiological conditions and performance of the human operator when exposed to various control situations. Eye movement and brain activity are key features to be observed in order to estimate the attention and awareness of the human operator, as described in [11]. These insights impact the design of future human-drone interfaces.

Such human-drone interfaces have been investigated intensively in recent years [4]. The state-of-the-art includes traditional input/output devices such as gamepads and game controls or smartphones, speech recognition, visual markers, and hand gestures and visual body interaction. In [2], the intuitiveness and flexibility of these interfaces are discussed. To accept drones as teammates, drone autonomy and (artificial) intelligence need to be considered when designing user interfaces. In our work, we integrate natural user interfaces by hand gesture detection with autonomous navigation and study the performance of this interaction mode.

In previous work [5, 6], we implemented the networked architecture of a supporting system for human-drone teams and introduced the basic vision-based functions for a bookshelf inventory task. First measurements showed promising low latencies which make a distributed architecture feasible. An improved text recognition algorithm and optimization of flight stability demonstrate that the scanning accuracy of such a system is sufficiently high. In this paper, we extend our work by introducing gesture-based interaction and a performance study of the human-drone team.

3 SYSTEM IMPLEMENTATION

The system design is targeted to a general human-drone scanning task where the on-board camera of the drone is leveraged to provide video and image data. Yet, the configurations of the system and vision-based algorithms are shaped to the use case bookshelf inventory.

3.1 Distributed Architecture

Figure 1 visualizes the distributed architecture of the camera-based human-drone scanning system. The drone captures and sends video data using its on-board FPV camera, performs flight maneuvers, and sends inertial status information back to the smartphone app, including current directional speed. The smartphone app receives image data from the drone, performs image recognition, and sends back flight commands. Further, the app provides several interfaces for user interaction, including touch-based steering through buttons and hand gestures. Human intervention always overrules the autonomous navigation logic. The smartphone and the drone are connected through Wi-Fi IEEE 802.11 and standard TCP and UDP messages are used. The FPV drone provides the Wi-Fi network and acts as an access point and DHCP server, while the smartphone connects to the Wi-Fi network as a client device.

[1] http://www.ros.org/
[2] http://mqtt.org/

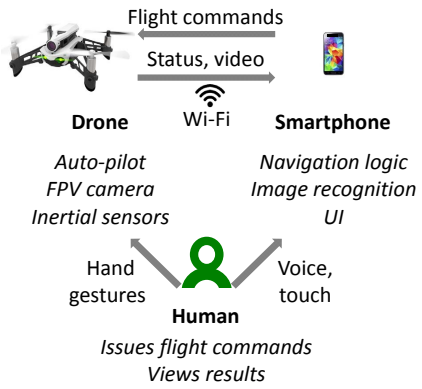

Figure 1: Distributed human-drone system architecture.

Table 1: Parrot FPV Mambo quadcopter hardware.

Frame, Battery	CPU	Sensors
180 × 180 × 40 mm, 70 g (with battery), 660 mAh (10 min)	800 MHz ARM proc.	3-axis accel., pressure, gyro, 720p camera

The drone employed in our testbed is the Parrot Mambo. Table 1 summarizes the major features of the drone. This light-weight quadcopter provides an autopilot and a flight command interface. The smartphone app is a custom Java-based Android app implementing the user interface of the bookshelf scanning task (visualization of detected books, voice and touch-based manual navigation), image and text recognition, and navigation control. The app uses Parrot's developers' classes for messaging,[3] OpenCV 4.1.1 for book detection (rectangle detection) and hand gesture detection,[4] and Google ML Toolkit for book title recognition (text recognition).[5]

The app was successfully run on four different Android smartphone platforms, among them a OnePlus 5T phone (Octa-core: 4 × 2.45 GHz, 4 × 1.9 GHz, GPU, 8 GB RAM). Due to the high processing load induced by image recognition, smartphones with a smaller computational footprint are not recommended.

3.2 Autonomous Scanning and Navigation

The system includes a simple SLAM-based algorithm, where the bookshelf is the area of interest in which books are detected. At the same time the book images are leveraged to navigate. Image recognition is continuously performed by the smartphone app. Obstacles that have the contour of a rectangle are identified as books (i.e., the angles have to be about 90 degrees and four edges need to be detected). Further, the ratio between the book's width and height is evaluated and a minimum number of books needs to be detected to recognize a bookshelf. The app also scans the book titles of books that are in the center of the view, i.e., where no distortion effects impair text recognition. The scan output contains the number of books and an estimate of book titles. Figure 2 shows the smartphone screen during one drone flight in front of a poster mimicking a bookshelf. Detected books are visualized by surrounding red rectangles.

[3]https://github.com/Parrot-Developers/Samples/blob/master/Android/
[4]http://opencv.org/
[5]https://developers.google.com/ml-kit/

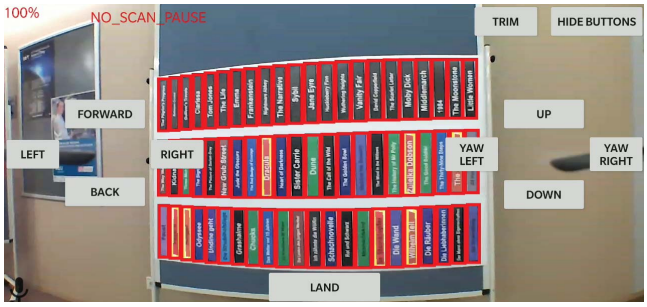

Figure 2: Smartphone GUI with camera feed of the drone, outcome of book detection, and UI buttons.

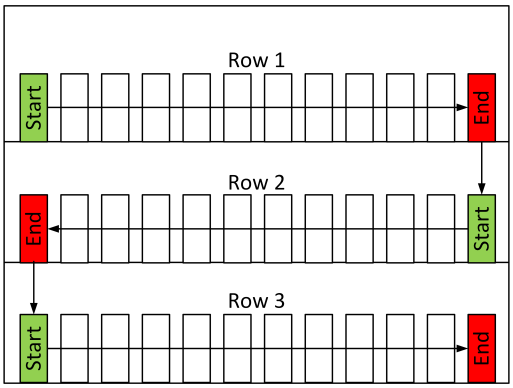

Figure 3: Zig-zag scan pattern used to scan the bookshelf; drone starts at the upper left corner.

Vision is also leveraged to autonomously navigate the drone. Figure 3 depicts the drone's scanning pattern. First, the drone moves to the upper left corner of the shelf, i.e., it flies up left until there are no more books spotted to the left and no upper row of books is detected. When this position is reached, the drone starts to perform a zig-zag movement pattern, scanning one row after the other until there are no more rows left. The current position of the drone is derived in relation to the bookshelf by analyzing the size and the position of the books in the view. To perform the zig-zag movement, 3D flight commands are issued to reposition the drone. Possible commands are moving up/down, forward/backward, and left/right. These high-level flight commands are based on Parrot's messaging API and packed into a TCP message. Each command can be represented by a four-tuple consisting of the vertical speed (gaz) and the three angles of movement along the aircraft axes, that are, pitch, yaw, and roll. The values are expressed as percentages of the maximum possible parameter value.

3.3 User Interaction

The smartphone app supports touch-based user interaction through traditional graphical user interface (GUI) buttons (see Figure 2)

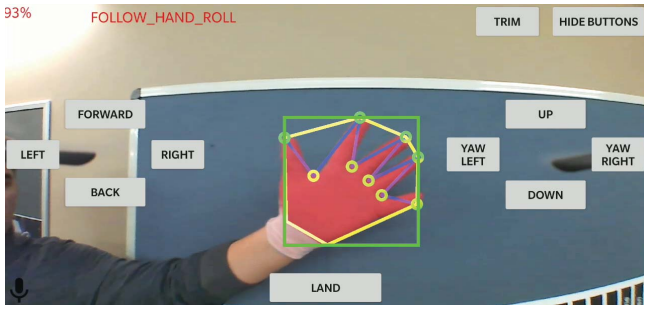

Figure 4: Hand detection visualized by the smartphone app: yellow color depicts the convex hull, circles correspond to convexity defect points and points on the hull, enclosing green rectangle is used for vision-based drone navigation.

but also gesture recognition.[6] Gesture-based interaction relies on vision-based detection of a hand and interpretation of the gesture.

To detect a hand, first color filtering is employed. We experimented with different colors including skin-like color and finally selected a red color region in the Hue Saturation Value (HSV) color space, which allows to differentiate a hand wearing a red glove from background colors usual in our use case, such as wood, white walls, etc. In a second step, the contour of the hand is detected by creating the convex hull of the region and detecting convexity defect points of the contour, which help to mark the fingers. A hand requires that the angles between neighboring fingers are in a range fitting to a typical hand frame (angle of 10 to 100 degrees to the thumb, and 10 to 60 degrees between all other neighboring fingers). All objects not matching these criteria are discarded. Figure 4 shows a detected hand visualized by the smartphone app. The polygon representing the hand is surrounded by the smallest possible rectangle, which is used for vision-based navigation.

It has to be noted that the hand detection algorithm requires the hand to be in the view of the drone, fingers need to be detectable, and the hand is best positioned perpendicular to the drone. With increased distance between the hand and the drone, detection accuracy decreases. In our experiments, distances up to 3 m did not result in detection failures, which is sufficient for close collaboration such as envisioned for bookshelf scanning.

Two gestures are currently implemented, *follow the hand* and *land*. When the drone detects one hand (i.e., the rectangle enclosing the hand) it follows the hand, i.e., it aims to keep the rectangle enclosing the hand in the center of the camera view and to keep a constant distance to the rectangle by continuously analyzing its relative size. Two hands are used to force the drone to land.

4 MEASUREMENT STUDY

In order to study the performance of the human-drone team, we measure (i) the drone command response time and investigate effects of increasing the command frequency, (ii) the flight-time and the scan performance of autonomous and semi-autonomous scanning, and (iii) the capabilities and limits of gesture-based interaction. We make use of the following metrics:

[6] Additionally, voice recognition is implemented, yet due to ambient noise of the drone, this interaction mode is erroneous and has not been followed further.

Figure 5: Experiment setup: poster-bookshelf and drone; floor markers are used for improved drone flight stability when used on plain colored floors.

Command response time. The command response time is measured on the smartphone as the time between the point in time a command is sent to the drone and the point in time the effect of the command is reported in the drone's inertial sensor response.

Flight reliability and flight-time. The reliability of a flight is measured as the ratio between successfully completed flights and overall performed flights. For successfully completed flights, the flight-time is measured between take-off and landing of the drone.

Book count and book detection accuracy. A task specific metric is the book count, measured as the ratio between the number of successfully counted books and the ground truth of available books. The accuracy of book detection is the ratio between book titles detected correctly and the number of different books available. Note that due to limitations in text recognition, a perfect match is rare, but still a human can interpret a closely matching title correctly. To express this, we define a book title to be correctly detected when a book exists in the shelf that has a Levenshtein distance of at most 20% to the book title estimate. The Levenshtein distance is a common string metric to describe the difference of strings [8].

4.1 Experiment Setup

All experiments are performed indoors with a poster mimicking the bookshelf. The start position of the Parrot Mambo drone is at the center of the bookshelf at a distance of about 120 cm to the poster wall, headed towards the poster. An Android smartphone OnePlus 5T is used for processing and provides the user interface. The distance between the drone and the smartphone varies between two and four meters. At the start of each experiment, the battery of the drone is at least 80%. Environmental conditions are similar in all experiments (daylight, no wind). Figure 5 visualizes the setup.

Table 2: Command response time.

Mean	Median	Std	Min	Max
394 ms	407 ms	154 ms	117 ms	754 ms

Table 3: Detected cycles with varying command time (CT).

CT (in ms)	1000	500	300	200	100	50	40
Nr. cycles	5	5	5	4	2	2	2

4.2 Command Response

The time between a command is issued and the drone moves accordingly is an important factor in vision-based navigation and has to be limited to assure real-time autonomous piloting. To study the command response time, the drone is placed at the start position headed towards the poster-shelf (see Figure 5), takes off, and performs the following flight operations: move up with a speed of 10%, hover, move down with a speed of 10%, and hover (the speed has been chosen consistently to the speed used for the scanning experiments presented later). Up and down movements are executed for a duration of 2 s, the hovering time is chosen uniformly and randomly in the interval of [2, 2.5] s. Randomization is used to avoid synchronization effects due to the fixed sending frequency of inertial sensor data from the drone to the smartphone (2 Hz). This cycle is repeated ten times to complete a flight. During the whole flight, the drone's vision algorithm is running to generate realistic system load. Overall, ten flights are performed resulting in 200 up/down commands.

To measure the command response time, the logs on the smartphone are analyzed. Let T_0 be the time the command is issued and T_1 be the time the corresponding inertial sensor response reporting the movement is received. The command response time is calculated as $T = T_1 - T_0$. While this measurement method avoids issues of clock synchronization between the drone and the smartphone, it adds delay introduced by sending the inertial sensor values to the smartphone. For finding T_1, the earliest point in time is selected, when the respective movement is detected. In detail, vertical movement is reflected by the speed value v_z of the inertial sensor response. If $v_z \leq Z_{up}$, an up movement is detected, and if $v_z \geq Z_{down}$, a down movement is detected. (In our study, $Z_{up} = -0.02$ and $Z_{down} = 0.06$, which has been empirically derived for the specific testbed drone.)

Table 2 summarizes the command response time results. The measured mean command response time is 394 ms. It has to be noted that the command response time includes transmission delays, yet, the small packet sizes and low latencies (mean round trip time of 10 ms [6]) of the direct short-distance Wi-Fi link between the drone and the smartphone impact the response time less than the fixed sending intervals of inertial values. The minimum command response time of 117 ms corresponds to a response sent without large sending delay of inertial values and is an indicator of the actual physical response of the drone.

To complement our experiment, we study the effect when increasing the command frequency, i.e., decreasing the command time (CT). Hovering times are scaled down with the command time. Five up and down cycles with hovering pauses as before are performed under varying command time settings, ranging from 1000 ms to 40 ms. Table 3 summarizes the number of detected cycles.

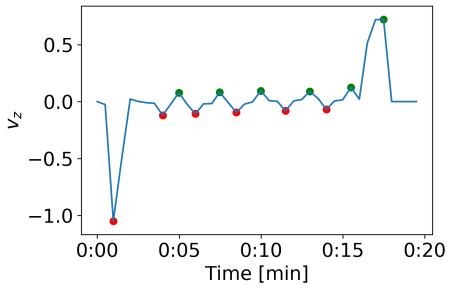

Figure 6: Vertical speed v_z, $CT = 500$ ms.

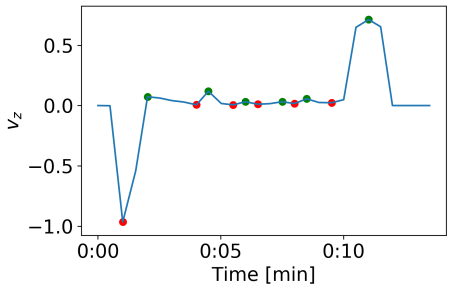

Figure 7: Vertical speed v_z, $CT = 200$ ms.

With decreased command time, the number of detected cycles decreases as the drone cannot report or perform all movements any more. Figures 6 and 7 visualize this effect. While Figure 6 shows a clear picture of the five up and down cycles expressed by the local extreme values of vertical speed v_z, the curve flattens in Figure 7. Here, only four cycles can be detected. Based on these results and confirmed by visual evaluation of the drone's actual flight, there is a minimal command time that should be adhered to, which is 300 ms in our setup.

4.3 Human Intervention

To see whether human intervention improves the results of the scanning task, the reliability of flight, flight-time, and the scan performance are evaluated for autonomous and semi-autonomous flights. Each flight starts at the starting position where the drone takes-off and performs the scanning task autonomously. Yet, during semi-autonomous flight the human intervenes and navigates the drone using smartphone buttons in cases the drone needs guidance, e.g., when the bookshelf is lost. Overall ten flights are performed for each flight mode.

A reliability of 0.8 is achieved for autonomous flight, and no failure is encountered for semi-autonomous flights. The median flight-time is 114 s and 108 s in autonomous and semi-autonomous mode, respectively, though the spread of values is high. Both modes can achieve a high book count ratio of above 0.98 and a very good mean accuracy of respectively 0.9 and 0.93. Figure 8 visualizes the

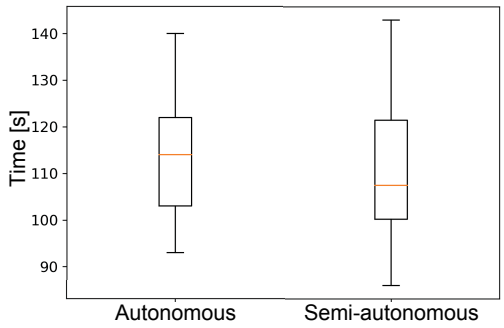

Figure 8: Flight-time when scanning the bookshelf in autonomous and semi-autonomous mode; boxplots: showing the median (middle line) and other quartiles (bottom and top line of the box).

Table 4: Book scanning results.

	Book count ratio	Accuracy
	Mean	Min, Mean, Max
Auto	0.998	0.77, 0.90, 0.99
Semi-auto	0.989	0.80, 0.93, 1

Table 5: Flight-time by interaction mode.

	GUI buttons	Hand gestures
	Mean, Median, Std	Mean, Median, Std
Turn	10 s, 10 s, 0.36 s	18.7 s, 17.9 s, 2.3 s
Recovery	3.0 s, 2.9 s, 0.53 s	6.6 s, 6.7 s, 0.8 s

flight-time statistics and Table 4 summarizes the scanning results. Whereas there is a clear improvement of 20% in terms of reliability when involving the human, remarkable differences in flight-time and scanning performance cannot be observed. As shown in the figure, the spread of the flight-time is higher when the human is involved due to manual steering.

4.4 Hand Gesture Interaction

Two different experiments are performed. First, the drone is placed at the start position but rotated by 180 degrees. After starting the drone, the task of the human is to manually *turn* the drone until it heads towards the bookshelf. The second experiment starts with the drone scanning the shelf when it is artificially forced to move up and, as a consequence, to loose sight of the shelf. The human's task is to guide the drone back to a *recovery* position. In both scenarios, ten flights are performed for each hand gesture interaction and traditional GUI button-based interaction.

Table 5 summarizes the flight-time needed in both scenarios. Using hand gestures results in a mean flight-time that is 1.87 times larger than the mean flight-time needed when using steering buttons in the turn scenario and 2.2 times larger in the recovery scenario. These huge differences are caused by the vision algorithm and the pause times the algorithm needs to navigate safely.

5 CONCLUSIONS

We studied the performance of a networked system supporting human-drone teams in solving a bookshelf inventory task. The system consists of a drone equipped with an on-board camera and basic piloting capabilities, and a smartphone app running vision-based image recognition and drone navigation control. In experiments, we showed that the mean command response time measured between the drone and the external control logic running on a smartphone is below 400 ms which renders high-level remote control feasible, yet, hard real-time guarantees are not possible through a common Wi-Fi link. The interventions of the human mainly improves the reliability of the scan flights from 80% (autonomous mode) to 100% (semi-autonomous mode), the flight-time needed to solve the task and the mean scanning performance cannot be improved. These results indicate that for a task comprising autonomous drone behavior to a large extent, human-drone collaboration may be leveraged to mitigate situations where drones cannot recover from a failure. When comparing intuitive hand gesture-based interaction with interaction by traditional smartphone app buttons, we find that gesture-based interaction is about two times slower. This insight may guide design choices on interaction modes in future human-drone collaborations.

REFERENCES

[1] M. Asadpour, B. Van den Bergh, D. Giustiniano, K. A. Hummel, S. Pollin, and B. Plattner. 2014. Micro Aerial Vehicle Networks: An Experimental Analysis of Challenges and Opportunities. *IEEE Communications Magazine* 52, 7 (July 2014), 141–149.

[2] S. Bonaiuto, A. Cannavò, G. Piumatti, G. Paravati, and F. Lamberti. 2017. Teleoperation of Robot Teams: A Comparison of Gamepad-, Mobile Device and Hand Tracking-Based User Interfaces. In *COMPSAC 2017: IEEE 41st Annual Computer Software and Applications Conference*, Vol. 2. IEEE, 555–560.

[3] S. Chaumette and F. Guinand. 2017. Control of a Remote Swarm of Drones/Robots Through a Local (Possibly Model) Swarm: Qualitative and Quantitative Issues. In *PE-WASUN '17: 14th ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks*. ACM, 41–45.

[4] R. A. S. Fernández, J. L. Sanchez-Lopez, C. Sampedro, H. Bavle, M. Molina, and P. Campoy. 2016. Natural User Interfaces for Human-drone Multi-modal Interaction. In *ICUAS 2016: International Conference on Unmanned Aircraft Systems*. IEEE, 1013–1022.

[5] A. Freistetter and K.A. Hummel. 2019. Human-Drone Teaming: Use Case Bookshelf Inventory. In *IoT 2019: 9th International Conference on the Internet of Things*. ACM, 1–4.

[6] K.A. Hummel, M. Pollak, and J. Krahofer. 2019. A Distributed Architecture for Human-Drone Teaming: Timing Challenges and Interaction Opportunities. *Sensors* 19, 6 (2019), 13.

[7] J. Krüger J. Guhl, J. Hügle. 2018. Enabling Human-Robot Interaction via Virtual and Augmented Reality in Distributed Control Systems. *Procedia CIRP* 76 (2018), 167–170.

[8] V. I. Levenshtein. 1966. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady* 10, 8 (1966), 707–710.

[9] Y. Liu, J. M. Montenbruck, D. Zelazo, M. Odelga, S. Rajappa, H. H. Bülthoff, F. Allgöwer, and A. Zell. 2018. A Distributed Control Approach to Formation Balancing and Maneuvering of Multiple Multirotor UAVs. *IEEE Transactions on Robotics* 34, 4 (Aug 2018), 870–882.

[10] K. McGuire, M. Coppola, Ch. de Wagter, and G. de Croon. 2017. Towards Autonomous Navigation of Multiple Pocket-drones in Real-world Environments. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (Vancouver, BC, Canada). IEEE, 244–249.

[11] A. H. Memar and E. T. Esfahani. 2018. Physiological Measures for Human Performance Analysis in Human-Robot Teamwork: Case of Tele-Exploration. *IEEE Access* 6 (2018), 3694–3705.