Single-Handed Gesture UAV Control and Video Feed AR Visualization for First Responders

Dimitrios Sainidis*
Visual Computing Lab (VCL),
Information Technologies Institute (ITI),
Centre for Research and Technology - Hellas (CERTH),
Thessaloniki, Greece †
dsainidis@iti.gr

Dimitrios Tsiakmakis
Visual Computing Lab (VCL),
Information Technologies Institute (ITI),
Centre for Research and Technology - Hellas (CERTH),
Thessaloniki, Greece

Konstantinos Konstantoudakis
Visual Computing Lab (VCL),
Information Technologies Institute (ITI),
Centre for Research and Technology - Hellas (CERTH),
Thessaloniki, Greece

Georgios Albanis
Visual Computing Lab (VCL),
Information Technologies Institute (ITI),
Centre for Research and Technology - Hellas (CERTH),
Thessaloniki, Greece

Anastasios Dimou
Visual Computing Lab (VCL),
Information Technologies Institute (ITI),
Centre for Research and Technology - Hellas (CERTH),
Thessaloniki, Greece

Petros Daras
Visual Computing Lab (VCL),
Information Technologies Institute (ITI),
Centre for Research and Technology - Hellas (CERTH),
Thessaloniki, Greece

ABSTRACT

Unmanned Aerial Vehicles (UAVs) are becoming increasingly widespread in recent years, with numerous applications spanning multiple sectors. UAVs can be of particular benefit to first responders, assisting in both hazard detection and search-and-rescue operations, increasing their situational awareness without endangering human personnel; However, conventional UAV control requires both hands on a remote controller and many hours of training to control efficiently. Furthermore, viewing the UAV video-feed on conventional devices (e.g. smartphones) require first responders to glance downwards to look at the screen, increasing the risk of accident. To this end, this work presents a unified system, incorporating single-hand gesture control for UAVs and an augmented reality (AR) visualization of their video feed, while also allowing for backup remote UAV control from any device and multiple-recipient video streaming. A modular architecture allows the upgrade or replacement of individual modules without affecting the whole. The presented system has been tested in the lab, and in field trials by first responders.

Keywords

First responders, UAV, gesture control, augmented reality.

*corresponding author
†http://vcl.iti.gr/

INTRODUCTION

Over the past decade, Unmanned Aerial Vehicles (UAVs or, more informally, drones) have become increasingly popular and affordable, assisting and cooperating with humans in various scenarios, such as inspection (Özaslan et al. 2017; Shihavuddin et al. 2019), mapping (Wu et al. 2019; Hill 2019), exploration (Joyce et al. 2019, human-drone interaction (Karjalainen et al. 2017), and search and rescue missions (Mishra et al. 2020; Burke et al. 2019). Specifically, in search and rescue scenarios First Responders (FRs), utilise UAVs for quickly identifying victims in large areas, investigating dangerous scenes with safety, or even inspecting inaccessible areas from a remote location by viewing their video-stream. However, controlling UAVs in such scenarios require skilled personnel and the use of cumbersome hand-held controllers that require both hands to operate, thus restricting FRs' motion and the performance of concurrent tasks. Additionally, viewing the video-stream on conventional devices such as mobile phones can distract FRs by requiring them to look away from their immediate surroundings, lowering situational awareness and increasing the risk of accident.

Towards this end, we present a system that combines a UAV with an augmented reality (AR) device and a hand gesture capturing sensor, allowing users to intuitively fly the drone with one hand and at the same time view the UAV video-stream on the AR device without taking their eyes from the scene. Our system is based on Microsoft's HoloLens[1], DJI's Mavic 2 Enterprise Dual UAV[2] and the LeapMotion controller[3]. However, due to its modular design, it is not bounded to these specific hardware components and can be adapted to any other UAV, AR device or motion sensor. The system has been tested by FRs and researchers both in the lab and in realistic response mission conditions.

This work is the extension and realization of previously published work in progress (Konstantoudakis et al. 2020), which had defined gestures for UAV control and validated them in a simulated environment. The main contributions of the present work include:

- The **extension** of the previously defined **gesture set**.
- The use of gestures to **control real UAVs**, with low latency and a full range of UAV motions.
- A system encompassing UAV **gesture control**, **visualization** of UAV's camera feed in AR, and optional backup control and video reception by a **Command & Control Center**.
- A **modular architecture** based on message broker exchanges that will allow the upgrade or replacement of individual modules independently from the whole system.

The presented work has been carried out in the context of EU-funded project FASTER[4], which aims at providing FRs with innovative tools and technologies. The project includes both technical and FR partners, which has allowed for co-design as well as validation of the current work.

The rest of this paper is organized as follows: Related work provides a review on current interfaces on UAV gesture control and visualization; Aims and requirements defines the scope of this work; System architecture describes the proposed solution's architecture and components; Gesture-based UAV control and AR visualization present the implementation and performance of the two main sub-systems. Lastly, Conclusions and future work provides a conclusion and discusses future work.

RELATED WORK

As drones are becoming increasingly popular, several organizations around the world use them to enhance their operational capabilities by developing innovative solutions (e.g. Giones and Brem 2017). For example, Volckaert 2018 describes a novel decision support system, in which drones fly above a certain area of interest, scanning for potentially dangerous situations and informing FRs on site. Search-and-rescue(S&R) missions are an integral part of an FR's job. To this end, Silvagni et al. 2017 presents a UAV-based system for S&R missions in mountainous terrains, allowing FRs to locate potential victims without the need to physically visit the site. Even though, UAVs can be employed for assisting FRs while operating on the field, controlling them with conventional hand-held devices can be challenging, and not ideal for these scenarios.

[1] https://docs.microsoft.com/en-us/hololens/hololens1-hardware

[2] https://www.dji.com/gr/mavic-2-enterprise

[3] https://developer.leapmotion.com/

[4] https://www.faster-project.eu/

On the other hand, gestures are an intuitive and integral part of human communication and are crucial at complementing verbal exchanges in daily communication, especially when discussing more complex topics (Pavlovic et al. 1997). Consequently, intuitive hand gestures can provide a natural alternative to conventional joystick-based control in human - UAV interaction, and significant research effort has been made towards the development of such interfaces. In our previous work (Konstantoudakis et al. 2020) we defined and evaluated two sets of single-hand gestures for UAV control, and similar to this work Sarkar et al. 2016 introduced a hand-gesture control interface. Authors in (Ge, Liang, et al. 2016, S. Yuan et al. 2018) gather data from depth sensors like those found on AR Devices (Hololens) and use it to estimate the 3D hand pose. Others (Zimmermann and Brox 2017, Spurr et al. 2018) use monocular RGB images to estimate the 3D location of key hand points like certain joints, which can accurately express the 3D shape of a hand. Authors in Walker et al. 2018 present a system that gathers data from sensors on the AR head-mounted display (HMD), detects the position of the head relative to its environment, identifies hand gestures, and combines all to control a UAV. Advances in deep learning led the authors of Molchanov et al. 2016 to propose a recurrent convolutional neural network (RNN) model leveraging 3D input form depth, color, and stereo-IR (infrared) sensors to overcome the ambiguity of how different people perform hand gestures. Tackling the same problem, Ge, Ren, et al. 2019 introduced a Graph Convolutional Neural Network (Graph CNN) method to reconstruct a 3D mesh of hand surface achieving greater detail in regards to hand shape and pose. A natural language interface based on gestures was proposed by Chandarana et al. 2017, whereby using twelve distinct and simple hand gestures they could define multiple trajectories. Nagi et al. 2014 cataloged unique spatial gestures through the use of colored gloves. A recent study by Chan et al. 2018 and an earlier one by Nagi et al. 2014 gathered data from three sensors, an RGB camera, an Inertial Measurement Unit (IMU), and flex sensors placed on the subject's fingers and managed to combine them to recognize hand pose and finger movement with a frequency of 100 Hz.

Another set of technologies that have been used for mitigating the challenges when designing robotic systems are AR and VR (Virtual Reality). For instance, Leutert et al. 2013 exploited spatial AR to display complex robot data in an understandable way while Qian et al. 2019 developed an AR-based platform to provide an "X-ray see-through vision" to a surgeon wearing an HMD. Similarly, Erat et al. 2018 developed a system allowing users to control a UAV from both an exocentric and egocentric view and at the same time by providing an X-ray see-through view, although his system relied on an external motion tracking system. In a following work, L. Yuan et al. 2019 proposed a non-invasive form of interaction where a UAV can be controlled by gaze, while Liu and Shen 2020 built a scaled 3D map of the environment rendered on an AR device where users could set points to navigate the drone in a more immersive and intuitive manner. Apart from these works, focusing on alleviating the challenges related to developing UAV systems, AR and VR have been applied to entertaining and educational activities as well. For instance, Thon et al. 2013 gamified the experience of visiting a museum with the use of a small UAV that can fly indoors. Users can control the drone and through its camera shoot augmented targets inside the museum which will reveal a short history lesson video. Mirk and Hlavacs 2014 explored the possibility of using a UAV and Virtual Reality hardware (e.g. Oculus Rift) to transform tourism from a physical-only activity to a virtual one as well. Although inspiring, most of the aforementioned works are not targeting the FRs' domain and the associated challenges when developing applications for such scenarios. In this work, we specifically focus on designing a modular UAV system targeting FRs' applications.

AIMS AND REQUIREMENTS

Project Aims

The main objective of this work is to provide the foundation for seamless and more intuitive integration of UAVs into first responder operations. This should extend to both **control** and **vision**, and take into account the conditions in which first responders operate, which are both physically and mentally challenging, and can often be dangerous.

Traditionally, UAVs are **controlled** via hand-held remote controllers. Although well-established, these are not ideal for such conditions: They require both hands to operate, effectively preventing the pilot from taking any other action while controlling the drone; they are not intuitive in their use, with one hand controlling pitch and roll and the other controlling yaw and throttle; and they add to the encumbrance of the first responder user, whose carrying capacity is already taxed with protective equipment, communications hardware, and other vital items.

Similarly, users usually **see** the drone's camera feed on a smartphone connected to the hand-held remote, which requires them to glance downwards to look at the screen. However, in the context of disaster response, taking their eyes off their surroundings can be dangerous, much like car drivers who are advised not to glance at their phones while driving.

With the above in mind, the present work aims to enhance UAV use in FR missions in the following ways:

- To make UAV control more intuitive and easier to learn.
- To reduce the fatigue associated with carrying and using a hand-held remote controller and allow UAV operators to use their off hand for other vital tasks.
- To provide improved UAV-assisted situational awareness.

The above objectives can be realized via two parallel aims:

1. To provide an intuitive and easy to learn gesture-based interface for controlling UAVs, obviating the need to carry and use a hand-held remote controller.
2. To integrate the UAV's camera feed into the user's field of view, allowing them to be constantly aware of it without taking their eyes off their immediate surroundings.

Requirements

The FASTER project, in whose context the present work was carried out, has identified a number of user requirements related to UAV control and visualization. These include single-handed control, intuitiveness, reliability and fast response times, and were drawn in collaboration with the FR partners of the project. This work presents the first completed version of the control-visualization system. Although it has not yet addressed all identified requirements, it serves to provide a platform both for the evaluation of the involved technologies and the updating of said requirements.

In our earlier work (Konstantoudakis et al. 2020), we had compiled a list of design requirements regarding UAV gesture control ensuring that the interface is intuitive and easy to both learn and use, in the context of a crisis response mission. Among others, some key points included:

- that control is achieved with a single hand, leaving the other hand free to perform tasks or carry vital equipment.
- that gestures are comfortable for the user.
- that similar but opposite gestures correspond to opposite UAV commands (e.g. "up" and "down").
- that gestures are mapped to UAV control values in a continuous manner, allowing smooth control.

The most important technical requirement in gesture control is latency. The UAV should respond to gestures almost immediately. A latency below 200 ms can be considered optimal, as this is close to the period at which many UAVs consume control data (see Gesture-based UAV control). A latency of 1 second or more would largely negate any usability or intuitiveness benefits from gesture control, as it would impose additional mental strain on the pilot, who must try to compensate for it.

Regarding the camera feed visualization, the primary requirement is the integration of the feed into the user's field of view in a manner that allows them to see both it and their surroundings at the same time. In addition, technical requirements include good video resolution, an adequate frame rate and low latency. While there are no specific thresholds for these, better playback qualities should both increase users' awareness and make for a more comfortable experience.

Modular Approach

This work touches on a number of different technologies: gesture recognition, UAVs, real-time communication, and visualization. Although all of them are mature enough for practical applications, all are also expanding fields, with new research and implementations surfacing to improve on the status quo or propose wholly different methodologies. This imposes an additional requirement on the present work, that of modularity: each part should be designed to be an independent module and communicate with the other parts in predefined formats. This approach can allow individual parts of the pipeline to be replaced or upgraded without compromising their inter-operation with the whole.

SYSTEM ARCHITECTURE

The system presented in this work aims to facilitate gesture control of a UAV and AR visualization of its camera feed. In this section, we present the overall system architecture, as well as hardware, software, and communication components utilized in its realization.

Overall Architecture

The system's two primary endpoints are the user and the UAV. A secondary, optional, endpoint can be the command and control center, whose personnel can receive the video feed and even take control of the aircraft.

Figure 1 shows an overview of all system components and their interrelation. In accordance with the Modular Approach requirement, the whole comprises of a number of largely independent modules, which can be exchanged or upgraded as needed without affecting the others. Modules communicate with each other via a Kafka broker (for UAV commands) and an RTMP (Real-Time Messaging Protocol) server (for streaming video), both preferably running on the same local network the user is connected to, in order to keep latency low. Optionally, the video stream flow can be routed through the broker instead, not making use of an RTMP server. In AR visualization the pros and cons of this approach are discussed.

Deployed in the user's immediate space, the **gesture capture** module consists of a computer with the LeapMotion peripheral. A Unity[5] application, utilizing the LeapMotion Software Development Kit (SDK), captures the user's hand motions and translates them into commands for the UAV. Worn on the user's head, the AR device hosts the **AR visualization** module, which receives the video stream from the drone, via the RTMP server, and displays it to the user.

The **UAV interface** connects the UAV to the rest of the system. It comprises of a custom Android application running on a smartphone connected to the drone's remote controller. The app constantly polls the broker for new commands and forwards them to the UAV, while also streaming its camera feed to the RTMP server. Note that the UAV interface module requires no user intervention beyond startup.

The **command and control center** (C&C) is an optional additional endpoint. This can be a local, on-site, coordination point, or a remote control center connected via the Internet. The drone's camera feed can be received by anyone able to connect to the RTMP server (depending on security and credentials required), hence multiple C&C personnel can view the video stream on their own computers, or receive streams from multiple UAVs in a larger-scale mission.

Due to the modular architecture, the command and control center can even take control of the UAV simply by posting commands to the appropriate broker topic. Such backup control can be realized using the keyboard, mouse, touchscreen, or gestures. Similarly, control of the UAV can be easily transferred to another user, as necessitated by the mission's development.

[5] https://unity.com/

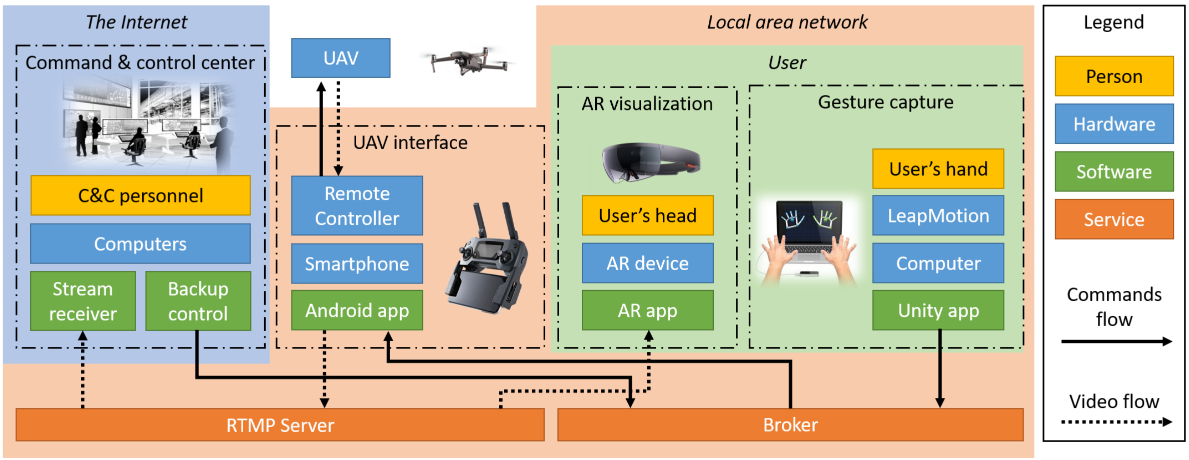


Figure 1. Architecture of the presented system, showing the three main modules (gesture capture, AR visualization and UAV interface) plus an optional command & control center connection. Two main types of information – UAV commands and video feed – are routed through the broker and the RTMP server, respectively.

Equipment

Users' hands are tracked by the LeapMotion controller (now a part of Ultraleap[6]), a USB peripheral device that uses infrared LEDs and cameras to capture the position of hands and fingers, including joints, with sub-millimeter accuracy and a high frame rate (Weichert et al. 2013). The controller is connected to a computer, which runs the gesture capturing and interpretation software.

The Microsoft HoloLens was used to provide an augmented (or mixed) reality display to the user. Early implementations used the first version of the HoloLens. The HoloLens 2[7], scheduled to be integrated into this system in the following months, is expected to bring improvements to the network connectivity, display resolution, field of view and camera. Software development for the HoloLens can be built as Universal Windows Platform (UWP) software.

The DJI Mavic 2 Enterprise Dual was used for developing and testing the architecture. At the time of ordering (Q1 2020), it was one of the few drones supported by the DJI Mobile SDK[8] platform, which was essential for development. Additional tests involved the DJI Mavic Mini[9], for which SDK support became available in July 2020. The integration of additional UAV models or AR devices requires only minor adaptations, due to the modular architecture.

The UAV interface hardware consists of the UAV's remote controller and a connected Android smartphone.

Figure 2 shows the complete user endpoint part of the system in action: the user holds his hand above the LeapMotion to control the UAV, while viewing its live video feed in HoloLens.

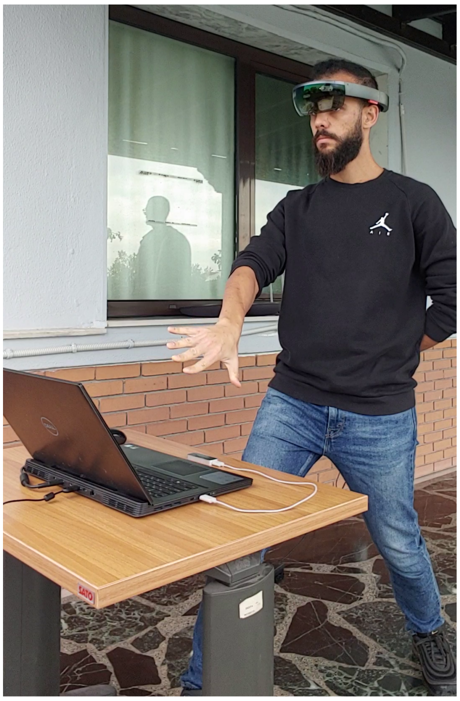


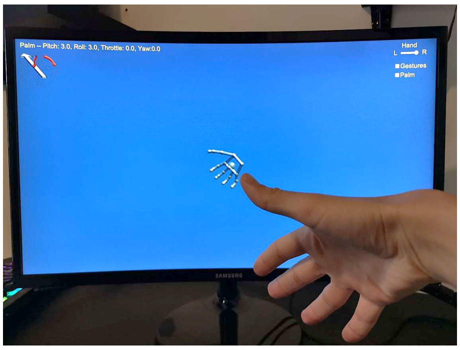
Figure 2. The complete user endpoint in action, consisting of the gesture capture module (LeapMotion controller connected to a computer) and the AR visualization module (HoloLens).

Software

Hand data is captured, interpreted, translated into UAV commands, and posted to the broker by a custom Unity application running on the computer to which the LeapMotion is connected. It uses relevant LeapMotion plugin and SDK functions and employs quaternion algebra to calculate 3D angles for the palm and individual fingers.

The app also includes a graphical user interface where a user can input the broker's address, select left or right hand, and enable or disable different modes of control. Textual and visual feedback informs the user of the detected gesture. Figure 3 shows the gesture capturing app in use, as a user performs a pitch forward/roll right gesture. Textual and graphical feedback can be seen in the upper left, while handedness and mode settings in the upper right.

The AR visualization module was created using Unity 3D and built for Universal Windows Platform (UWP), in order to be compatible with the HoloLens. The Mixed Reality Toolkit[10] (MRTK) provided a range of necessary components and features relevant for AR.



Palm -- Pitch: 3.0, Roll: 3.0, Throttle: 0.0, Yaw:0.0

Figure 3. The gesture capture and interpretation app. Here, the user commands the drone to pitch forward and roll to the right. Note the textual and visual feedback on the upper left and settings on the upper right.

[6] https://www.ultraleap.com/

[7] https://www.microsoft.com/en-us/hololens/hardware

[8] https://developer.dji.com/mobile-sdk/

[9] https://www.dji.com/gr/mavic-mini

[10] https://microsoft.github.io/MixedRealityToolkit-Unity/

DJI's Mobile SDK coupled with the Android Studio IDE[11] was used to develop an Android app to act as an interface between the UAV and the rest of the pipeline. The SDK acts as an interface between the remote controller and the smartphone attached to it. It allows for high-level flight control, aircraft telemetry recording, state information for both the aircraft and the remote controller, access to the aircraft's internal storage, full control of the camera and gimbal, and finally live video capture and streaming.

Communication

There are two main communication channels connecting the different modules: The RTMP server facilitates real-time video streaming, while the message broker handles all other communication, including UAV commands, connection details to the RTMP server, and any additional metadata desired (UAV GPS coordinates, heading, battery status, etc.). Although the message broker can disseminate any type of messages, including video frames, an RTMP server is optimized for this task and is often the preferred solution.

To achieve as close to real-time video transmission as possible, the RTM protocol was used which offers as low as 1 second delay between the server and the client and also offers high frame rate capability (maximum of 30 FPS) when the connection allows it. The lightweight communication server MonaServer[12] was used to simplify the process of live video streaming. MonaServer was easy to set up and required minimal system resources. Once the right port was forwarded, the server was ready to receive video feed from any source.

Generic message exchange in the presented system was facilitated by an Apache Kafka[13] broker, which supports both binary and JSON formats and incurs minimal latency. The broker's contents are organized into topics, to which messages are posted, and subsequently downloaded from. As there is no Kafka client implementation for either Android or UWP-compatible Unity builds, the UAV interface and AR modules must communicate with Kafka using REST (HTTP) operations, such as GET and POST. To that end, the Confluent Kafka[14] implementation was used, which includes a REST proxy and a corresponding API.

GESTURE-BASED UAV CONTROL

Gesture Capture and Interpretation

Background

In our previous work (Konstantoudakis et al. 2020) we had defined two UAV control gesture sets: finger-based control, in which gestures are differentiated according to which fingers are extended; and palm-based control, in which the drone mimics the movements and tilt of the user's palm. Having early implementations of both connected to a UAV flight simulator, a user-study had been conducted to identify user preference ans rate the comfort, usability and learning curve of each.

In the current work, that gesture capture interface has been improved and refined, and used as part of the system architecture to control a real UAV with hand gestures. As the previous user study concluded that palm-based control is more popular, especially among more experienced users, this mode is the main focus of the current development. However, finger-based control is still supported, and the two may be used alone or in tandem, as preferred by each user.

Overview

In palm-based control, users control the drone using their open hand with fingers splayed. Hand tracking data by the LeapMotion include finger state, normal vectors and the hand's distance from the controller itself. The controller is placed on a flat surface (e.g. a desk) in front of the user. The gesture capturing and interpretation module is packaged into a Unity application that uses the LeapMotion SDK to track the hand's state and orientation, interprets it and translates it to basic UAV commands (pitch, roll, yaw and throttle).

Palm-based control is active when all fingers are extended. The hand's motions correspond with analogous movements of the drone, with:

- hand tilt corresponding to pitch and roll
- horizontal hand rotation corresponding to yaw

[11] https://developer.android.com/studio

[12] https://sourceforge.net/projects/monaserver/

[13] https://kafka.apache.org

[14] https://www.confluent.io/

- lateral movement corresponding to altitude (throttle) change

When fingers are not extended this corresponds to no command, which causes the UAV to brake and remain stationary. Gestures are relative to a rest state, with calibration taking place automatically when first the all-fingers-extended state is detected. This sets the zero point for both altitude and yaw: moving the hand higher than the the zero point will cause the aircraft to ascend; rotating it so that it point to the left compared to the zero state will cause it to yaw to the left; and so on.

For more extensive information on the two modes of gesture control, please refer to Konstantoudakis et al. 2020.

Improvements

For the present work, the output of the gesture capturing and interpretation module has been routed to the broker, as outlined in System architecture. UAV commands are compiled into a JSON structure, which constitutes the body of the message posted to the broker.

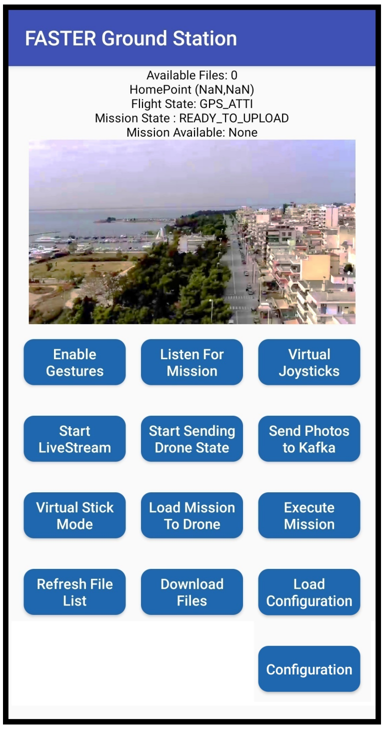
Moreover, two planned additions to the module have been realized: support for the **left hand**; and the definition of gestures to correspond to **take-off, landing, and abort landing** commands. In contrast to piloting gestures, which use as input the hand state without any memory or consideration of previous moments, the take-off and landing gestures are defined with a **time component**. The corresponding gestures are similar to the ascend/descend motion but use an upward-facing palm. In order to complete take-off or landing, the user must execute the respective gesture and hold it for a few seconds. This mirrors similar commands in the hand-held remote, where, for example, take-off is accomplished by holding both sticks down and inwards for 1 second. Landing abortion has been mapped to an "ascend" command during the landing procedure.

Interface Application with Real UAV

An Android application was developed to act as a communication interface with the drone. Designed to be installed on a smartphone physically connected to the UAV remote controller, it is responsible for connecting it to the rest of the system architecture (in particular the Kafka broker, and the RTMP server), and processing any data to and from the drone. The aircraft itself connects to the remote controller using a separate communication protocol (either 802.11g WiFi or DJI's proprietary OcuSync 2.0).

The main screen of the app is shown in Figure 4. The user can connect to a Kafka broker and an RTMP server by typing the corresponding IP addresses and ports on the Configuration page. There, one can also adjust the sensitivity of the four basic controls all quadcopters have: yaw, throttle, pitch and roll. This can allow different pilots to adapt gesture control to their own range of hand motions and level of expertise.

The app is responsible for downloading control messages from Kafka in real time, parsing the JSON structure, translating them into UAV commands and forwarding those commands to the drone. In addition, drone telemetry data are captured every second and include: GPS position (latitude, longitude and altitude), heading expressed in degrees from the earth's magnetic north, and the percentage of the remaining battery capacity of the aircraft. This information can also be posted live to Kafka and can be crucial to mission control, including a remote C&C. Lastly, the app gives the user access to the internal storage of the aircraft, where photographs or recorded videos are stored. Any media can be downloaded to the smartphone or be uploaded to a predefined Kafka topic. It also includes additional functionalities not directly relevant to the present work, such as virtual stick navigation and waypoint mission support.


FASTER Ground Station

Available Files: 0
HomePoint (NaN,NaN)
Flight State: GPS_ATTI
Mission State : READY_TO_UPLOAD
Mission Available: None

Figure 4. UAV interface application main screen, showing live video feed and buttons to available functionalities.

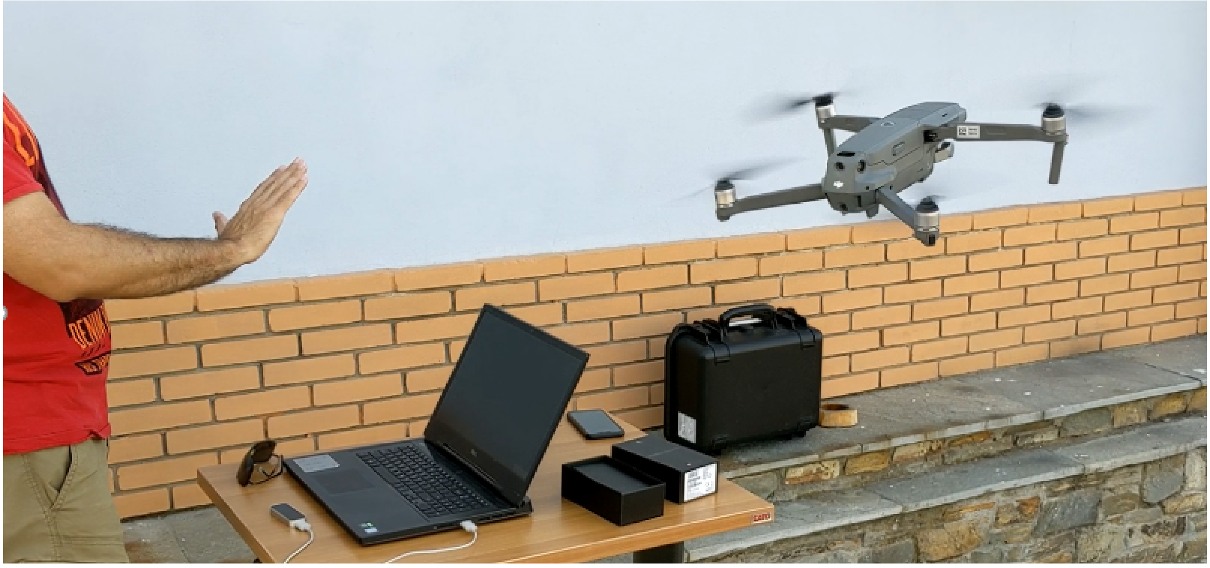


Figure 5. Gesture control integration with the DJI Mavic 2 Enterprise. Note how the drone tilts to follow the user's hand.

Performance

Regarding the performance of gesture-based control, the main goal was to achieve the same level of control and fluency of movement as a hand-held remote controller would allow. Specifically, the aircraft should be able to execute automated take-off and landing, as well as complex maneuvers by combining all four controllable degrees of freedom. The latency, which is the delay between a user issuing a command and the drone executing it, must be low enough to allow pilots to react effectively, and ideally should be unnoticeable even by a trained pilot. Finally, the movement should be fluent and continuous without any stutters.

During testing the drone responded almost instantaneously, following the tilt and orientation of the user's hand, as can bee seen in Figure 5. It could perform the same range of motions as the remote controller, resulting in smooth flight. Test pilots could efficiently guide the aircraft to any desired position and orientation.

To independently test the performance of the gesture-based UAV control system, several volunteers, who had previously helped define and evaluate gestures in a simulated environment in our previous work, were asked to control a real drone and assess its responsiveness. All reported that the gestures translated to drone movements intuitively and without any noticeable delay. Volunteer testers included two FRs who are also trained UAV pilots, at varying levels of experience (one intermediate and one expert); they also reported their satisfaction at the system's responsiveness. In addition, they expressed the opinion that, for a novice user, the gesture-based control would be more intuitive and easier to learn than the actual remote controller. The expert pilot affirmed that with minor improvements he would feel confident performing a real mission with this system.

To measure the latency of gesture control, a high-speed camera was set up to capture both user and drone at 240 frames per second. Afterward, the number of elapsed frames between gesture completion and drone reaction was counted. Latency was measured in three different scenarios, depending on the location of the broker: on the same local network as the user endpoint; in the same city; and in a different country altogether. Figure 6 shows both the minimum and the average latency measured in this experiment. Frame measurement at 240 frames per second allows for an error of about 4 ms, which is equivalent to less than 3%. That small amount of error has no impact on the conclusions of this experiment.

As mentioned before, a DJI UAV consumes commands from its remote controller every 40-200 ms, hence that degree of latency is inevitable. It may be noted that the latency in a local area network is minimal, at less than 200 ms, i.e. no worse than that of conventional control. Naturally, latency increases when a remote broker is used. However, at around 300 ms, latency is still within acceptable limits for UAV control even when the broker is in a different country.

Due to the infrared cameras of the LeapMotion involved in gesture capturing, robustness often suffered during testing in bright sunlight or on overcast days, as a result of high levels of infrared radiation scattered in the atmosphere. This could be moderated to a large extend by placing the LeapMotion controller in shade and avoiding open sky backgrounds.

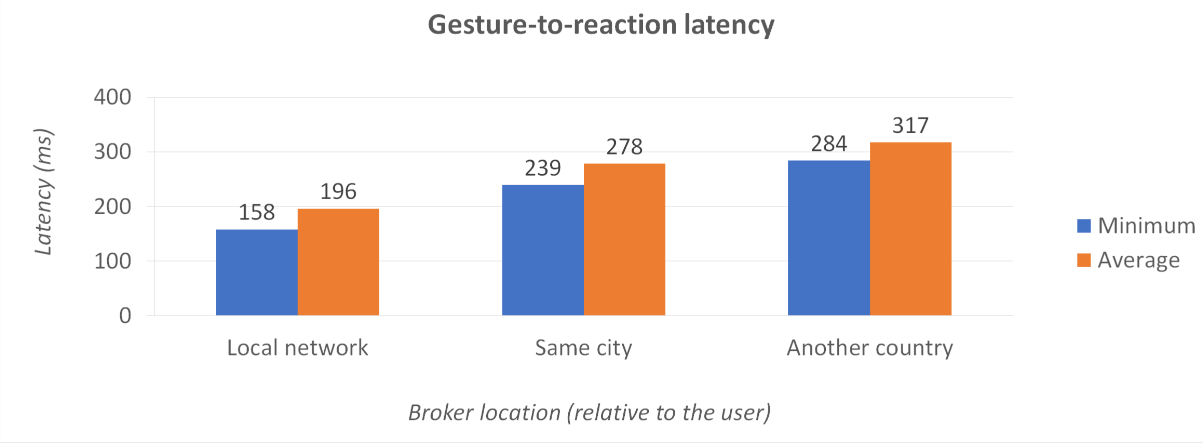


Figure 6. Latency measurements according to broker location relative to the user.

A more thorough validation of the presented system requires larger-scale tests, a wider base of volunteer participants, realistic scenarios, and a setup of objective and subjective performance measurements, similar to the ones performed in simulation in Konstantoudakis et al. 2020. However, due to the COVID-19 pandemic and restrictions associated with it, it was not possible to conduct such tests thus far. By necessity, these will be planned as future work in the following months, and will provide the opportunity to test both the present system as well as future improvements.

AR VISUALIZATION

AR visualization aims at integrating the UAV's camera feed in the user's field of view, allowing them to be aware of both the video feed and their natural surroundings simultaneously, thus maximizing their situational awareness. To this end, a HoloLens application was developed and different streaming methods and settings were tested to identify the optimal solution.

HoloLens Application

The AR visualization app for HoloLens was developed in a Unity 3D environment and built for UWP, a Windows 10 platform for multiple devices (IoT, mobile, desktop, XBox, etc). The main part of the interface consists of the live video player. A toggle button can select the video source as there are multiple options for video-streaming. Two streaming methods can be used for representation, Apache Kafka and RTM Protocol as outlined in System architecture. On the interface, parameters like the broker's IP, relevant topic names, and the URL of the RTMP server can be configured by the user via a virtual keyboard. In addition, two buttons provide play and stop video functionality. Figure 7 shows a first-person view captured by the HoloLens, with the app window in the center and the real-world surroundings, including the UAV, in the background. The app communicates with the server (Kafka or RTMP) over a local network or the Internet.

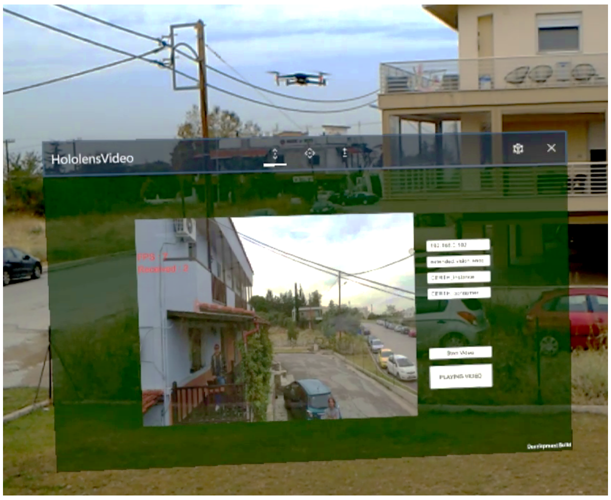


Figure 7. A first-person point of view, as captured by the HoloLens. As the drone is facing the pilot, he can be seen in the lower left in the AR video window.

Video Streaming

Two video streaming methods were implemented and evaluated: RTMP streaming and streaming via the Kafka broker. Streaming performance was evaluated on two criteria: frame rate and latency.

RTMP streaming is natively supported by the DJI SDK and is the most widespread solution for streaming in general. It supports frame-rates of up to 30 frames per second, depending on network quality. Latency is largely dependent on the network. Similarly to the gesture-reaction latency, RTMP latency was measured using a high-speed camera, and it was found to range from a minimum of about 3 seconds, when using a dedicated RTMP server on the same local network, to about 6 seconds over the Internet.

Streaming video via the Kafka broker is a non-standard solution that was pursued in an attempt to minimize latency, since message exchanges through Kafka is usually realized with millisecond-latency. This approach requires that each frame is posted to the broker by the UAV interface app as a separate message, while the HoloLens app continuously listens for new messages.

However, HoloLens connection to the Kafka broker has been inconsistent and largely problematic, as several milliseconds are routinely wasted to establish the connection for each polling request. This connection delay imposes a significantly low limit on frame-rate, at about 7 frames per second. Latency was significantly lower than the RTMP, at about 130 ms.

In comparison, in a generic use case, Kafka's lower latency cannot compensate for the largely reduced frame-rate and inconsistent playback. However, this issue is probably particular to the HoloLens, as the same application running on a desktop computer exhibited much better connectivity. The HoloLens 2, scheduled to be integrated into this project in the coming months, is expected to have improved connectivity performance, hence further evaluation will be performed on that latest version.

CONCLUSIONS AND FUTURE WORK

In this work, we have presented a system for the easy and intuitive control of UAVs using single-hand gestures and the visualization of the UAVs' camera feed in augmented reality. The work is aimed towards and inspired by first responders, who need both intuitive control and free hands in their exacting and often dangerous missions.

Three main subsystems have been implemented and presented: gesture capture, AR visualization, and UAV interface. A modular architecture has been defined, with modules communicating via a message broker and an RTMP server. Each module is largely independent as long as it complies with the defined input and output formats, hence each can be upgraded or replaced with future implementations. The complete system has been tested both in the lab and in actual conditions, including by first responders.

The present work is the continuation of a previously published work in progress. Although it has reached some maturity and been successfully deployed in real conditions, work will continue. Despite the fact that early implementations focused on the Android OS, more mature versions will also be ported to iOS in order to support a wider range of mobile devices. Regarding gesture control, future plans focus on improving portability and robustness. The present work captures motions with the LeapMotion controller, which, being a USB peripheral, requires a computer to function. Declared plans by LeapMotion developers to support Android have not yet come to fruition, and, due to the company's merger into UltraLeap, may never do. In addition, LeapMotion hand tracking, while very robust indoors, often suffers outdoors during the daytime. This is likely a consequence of its infrared-based detection, which is foiled by the background of a bright sky.

Clearly, in order to accommodate first responder requirements, a more robust and portable mode of hand tracking is needed. Towards that end, plans involve the exploration of three different modalities: vision-based hand tracking, such as Mueller et al. 2018; *Google AI Blog* 2020; tracking based on colored (R. Y. Wang and Popović 2009) or non-colored (Han et al. 2018 markers, especially appropriate for first responders as it can easily be integrated into protective gloves; tracking using the vision and depth sensors integrated into HoloLens (Puljiz et al. 2019; *Mixed Reality Toolkit* 2020); or a combination of the above, such as Ababsa et al. 2020.

Regarding the AR visualization, the current version displays the UAV's video feed in a floating window. This is a first step that does not capitalize on the full range of AR's capabilities. Future work in this regard will focus on employing co-registration between virtual objects (the video feed) and real-world objects (the UAV, but also buildings or other obstacles).

An initial step toward this end, will be to track the UAV in virtual space using live GPS coordinates and/or IMU sensor readings, such as from an accelerometer or gyroscope. This would create a virtual counterpart for the UAV that would track it even when there is no direct line of sight. When there is a line of sight, localization could be enhanced and made more robust with deep-learning-based automatic visual correction.

Further steps would utilize this UAV tracking to allow for a contextualized video display that can position, scale and orient the video to be consistent with the user's point of view. Monocular depth estimation (e.g. Godard et al. 2019) can be employed to refine the positions of objects captured by the UAV's camera and position them accordingly in the virtual space. The HoloLens's on-board spatial mapping can also register nearby walls and other objects. Ultimately, the goal is to create the illusion of vision behind obstacles, as inspired by Y. Wang et al. 2007.

As this planned co-registration research provides a wider range of options, respective user-studies must also be conducted to determine which visualization configurations is comfortable, intuitive, and appropriate for the users. This could depend on the circumstances: users might prefer to view nearby objects out of their direct line of sight in scale and context, but more distant views might be more useful when viewed in their full size. Thus, future work in UAV camera feed AR visualization will follow two parallel and interconnected paths: co-registration between the real and virtual worlds; and visualization user studies.

Finally, due to restrictions posed by the COVID-19 pandemic, our collaboration with FRs was limited to only a few domestic field trials. Nonetheless, more testing will be planned as future work to gather feedback from end users and improve the aforementioned systems so that they meet their special needs and requirements. Also, the involvement of FRs in the testing process is expected to result in future improvements.

ACKNOWLEDGMENTS

This research has been supported by the European Commission funded program FASTER, under H2020 Grant Agreement 833507. The authors would like to thank the Hellenic Rescue Team Attica for testing the proposed system and providing valuable feedback. Additional thanks are due to volunteer first responders and drone operators Michail Fotoglou and Eleni Antoniou for testing the gesture control system during its development and offering feedback from an experienced operator's point of view.

REFERENCES

Ababsa, F., He, J., and Chardonnet, J.-R. (2020). "Combining HoloLens and Leap-Motion for Free Hand-Based 3D Interaction in MR Environments". In: *International Conference on Augmented Reality, Virtual Reality and Computer Graphics*. Springer, pp. 315–327.

Burke, C., McWhirter, P. R., Veitch-Michaelis, J., McAree, O., Pointon, H. A., Wich, S., and Longmore, S. (2019). "Requirements and Limitations of Thermal Drones for Effective Search and Rescue in Marine and Coastal Areas". In: *Drones* 3.4, p. 78.

Chan, T. K., Yu, Y. K., Kam, H. C., and Wong, K. H. (2018). "Robust hand gesture input using computer vision, inertial measurement unit (IMU) and flex sensors". In: *2018 IEEE International Conference on Mechatronics, Robotics and Automation (ICMRA)*. IEEE, pp. 95–99.

Chandarana, M., Meszaros, E. L., Tmjillo, A., and Allen, B. D. (2017). "Analysis of a gesture-based interface for uav flight path generation". In: *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, pp. 36–45.

Erat, O., Isop, W. A., Kalkofen, D., and Schmalstieg, D. (2018). "Drone-augmented human vision: Exocentric control for drones exploring hidden areas". In: *IEEE transactions on visualization and computer graphics* 24.4, pp. 1437–1446.

Ge, L., Liang, H., Yuan, J., and Thalmann, D. (2016). "Robust 3d hand pose estimation in single depth images: from single-view cnn to multi-view cnns". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3593–3601.

Ge, L., Ren, Z., Li, Y., Xue, Z., Wang, Y., Cai, J., and Yuan, J. (2019). "3d hand shape and pose estimation from a single rgb image". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 10833–10842.

Giones, F. and Brem, A. (2017). "From toys to tools: The co-evolution of technological and entrepreneurial developments in the drone industry". In: *Business Horizons* 60.6, pp. 875–884.

Godard, C., Mac Aodha, O., Firman, M., and Brostow, G. J. (2019). "Digging into self-supervised monocular depth estimation". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3828–3838.

Google AI Blog (2020). URL: https://ai.googleblog.com/2019/08/on-device-real-time-hand-tracking-with.html (visited on 01/23/2020).

Han, S., Liu, B., Wang, R., Ye, Y., Twigg, C. D., and Kin, K. (2018). "Online optical marker-based hand tracking with deep labels". In: *ACM Transactions on Graphics (TOG)* 37.4, pp. 1–10.

Hill, A. C. (2019). "Economical drone mapping for archaeology: Comparisons of efficiency and accuracy". In: *Journal of Archaeological Science: Reports* 24, pp. 80–91.

Joyce, K., Duce, S., Leahy, S., Leon, J., and Maier, S. (2019). “Principles and practice of acquiring drone-based image data in marine environments”. In: *Marine and Freshwater Research* 70.7, pp. 952–963.

Karjalainen, K. D., Romell, A. E. S., Ratsamee, P., Yantac, A. E., Fjeld, M., and Obaid, M. (2017). “Social drone companion for the home environment: A user-centric exploration”. In: *Proceedings of the 5th International Conference on Human Agent Interaction*, pp. 89–96.

Konstantoudakis, K., Albanis, G., Christakis, E., Zioulis, N., Dimou, A., Zarpalas, D., and Daras, P. (2020). “Single-Handed Gesture UAV Control for First Responders – A Usability and Performance User Study”. In: *17th International Conference on Information Systems for Crisis Response and Management (ISCRAM 2020), Blacksburg, VA (USA), May 24-27, 2020*. Vol. 17. ISCRAM, pp. 937–951.

Leutert, F., Herrmann, C., and Schilling, K. (2013). “A spatial augmented reality system for intuitive display of robotic data”. In: *2013 8th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. IEEE, pp. 179–180.

Liu, C. and Shen, S. (2020). “An Augmented Reality Interaction Interface for Autonomous Drone”. In: *arXiv preprint arXiv:2008.02234*.

Mirk, D. and Hlavacs, H. (2014). “Using Drones for Virtual Tourism”. In: *Intelligent Technologies for Interactive Entertainment*. Ed. by D. Reidsma, I. Choi, and R. Bargar. Cham: Springer International Publishing, pp. 144–147.

Mishra, B., Garg, D., Narang, P., and Mishra, V. (2020). “Drone-surveillance for search and rescue in natural disaster”. In: *Computer Communications*.

Mixed Reality Toolkit (2020). URL: https://microsoft.github.io/MixedRealityToolkit-Unity/ (visited on 12/03/2020).

Molchanov, P., Yang, X., Gupta, S., Kim, K., Tyree, S., and Kautz, J. (2016). “Online detection and classification of dynamic hand gestures with recurrent 3d convolutional neural network”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4207–4215.

Mueller, F., Bernard, F., Sotnychenko, O., Mehta, D., Sridhar, S., Casas, D., and Theobalt, C. (2018). “Ganerated hands for real-time 3d hand tracking from monocular rgb”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 49–59.

Nagi, J., Giusti, A., Gambardella, L. M., and Di Caro, G. A. (2014). “Human-swarm interaction using spatial gestures”. In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, pp. 3834–3841.

Özaslan, T., Loianno, G., Keller, J., Taylor, C. J., Kumar, V., Wozencraft, J. M., and Hood, T. (2017). “Autonomous navigation and mapping for inspection of penstocks and tunnels with MAVs”. In: *IEEE Robotics and Automation Letters* 2.3, pp. 1740–1747.

Pavlovic, V. I., Sharma, R., and Huang, T. S. (1997). “Visual interpretation of hand gestures for human-computer interaction: A review”. In: *IEEE Transactions on pattern analysis and machine intelligence* 19.7, pp. 677–695.

Puljiz, D., Stöhr, E., Riesterer, K. S., Hein, B., and Kröger, T. (2019). “Sensorless hand guidance using microsoft hololens”. In: *2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. IEEE, pp. 632–633.

Qian, L., Zhang, X., Deguet, A., and Kazanzides, P. (2019). “Aramis: Augmented reality assistance for minimally invasive surgery using a head-mounted display”. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, pp. 74–82.

Sarkar, A., Patel, K. A., Ram, R. G., and Capoor, G. K. (2016). “Gesture control of drone using a motion controller”. In: *2016 International Conference on Industrial Informatics and Computer Systems (CIICS)*. IEEE, pp. 1–5.

Shihavuddin, A., Chen, X., Fedorov, V., Nymark Christensen, A., Andre Brogaard Riis, N., Branner, K., Bjorholm Dahl, A., and Reinhold Paulsen, R. (2019). “Wind turbine surface damage detection by deep learning aided drone inspection analysis”. In: *Energies* 12.4, p. 676.

Silvagni, M., Tonoli, A., Zenerino, E., and Chiaberge, M. (2017). “Multipurpose UAV for search and rescue operations in mountain avalanche events”. In: *Geomatics, Natural Hazards and Risk* 8.1, pp. 18–33. eprint: https://doi.org/10.1080/19475705.2016.1238852.

Spurr, A., Song, J., Park, S., and Hilliges, O. (2018). “Cross-modal deep variational hand pose estimation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 89–98.

Thon, S., Serena-Allier, D., Salvetat, C., and Lacotte, F. (2013). "Flying a drone in a museum: An augmented-reality cultural serious game in Provence". In: *2013 Digital Heritage International Congress (DigitalHeritage)*. Vol. 2, pp. 669–676.

Volckaert, B. (2018). "Aiding First Incident Responders Using a Decision Support System Based on Live Drone Feeds". In: *Knowledge and Systems Sciences: 19th International Symposium, KSS 2018, Tokyo, Japan, November 25-27, 2018, Proceedings*. Vol. 949. Springer, p. 87.

Walker, M., Hedayati, H., Lee, J., and Szafir, D. (2018). "Communicating robot motion intent with augmented reality". In: *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, pp. 316–324.

Wang, R. Y. and Popović, J. (2009). "Real-time hand-tracking with a color glove". In: *ACM transactions on graphics (TOG)* 28.3, pp. 1–8.

Wang, Y., Krum, D. M., Coelho, E. M., and Bowman, D. A. (2007). "Contextualized videos: Combining videos with environment models to support situational understanding". In: *IEEE Transactions on Visualization and Computer Graphics* 13.6, pp. 1568–1575.

Weichert, F., Bachmann, D., Rudak, B., and Fisseler, D. (2013). "Analysis of the accuracy and robustness of the leap motion controller". In: *Sensors* 13.5, pp. 6380–6393.

Wu, K., Rodriguez, G. A., Zajc, M., Jacquemin, E., Clément, M., De Coster, A., and Lambot, S. (2019). "A new drone-borne GPR for soil moisture mapping". In: *Remote Sensing of Environment* 235, p. 111456.

Yuan, L., Reardon, C., Warnell, G., and Loianno, G. (2019). "Human gaze-driven spatial tasking of an autonomous MAV". In: *IEEE Robotics and Automation Letters* 4.2, pp. 1343–1350.

Yuan, S., Garcia-Hernando, G., Stenger, B., Moon, G., Yong Chang, J., Mu Lee, K., Molchanov, P., Kautz, J., Honari, S., Ge, L., et al. (2018). "Depth-based 3d hand pose estimation: From current achievements to future goals". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2636–2645.

Zimmermann, C. and Brox, T. (2017). "Learning to estimate 3d hand pose from single rgb images". In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4903–4911.