Controlling UAV camera using Leap Motion Sensor

YeSeul Kim
College of Engineering
Texas A&M University
College Station, USA
leslieyskim@tamu.edu

Raghav Hari Krishna
Dept. of Computer Science and Engineering
Texas A&M University
College Station, USA
vsraghavhk@tamu.edu

ABSTRACT

UAVs generally have a camera on them which is often controlled by a dedicated controller. Replacing the control system with a gesture-based control system provides more intuitive user interface and better human-robot interaction. In this project we attempt to implement this setup using a Leap Motion Controller and a Parrot drone in order to demonstrate the benefits of gesture-based drone control in human-robot interaction.

KEYWORDS

Human-Robot Interaction, Gestures, Leap Motion Controller, UAV

1 Introduction

Facilitated by the recent robotic development in both hardware and software, it expanded the potential applications of robots. In particular, Unmanned Aerial Vehicles (UAVs), or drones, have gained popularity in diverse applications such as search and rescue and constructions. However, the deployment of such robots in real world settings can be a challenging task because of the unstructured nature of the environment. As a result of the challenging environment, the human supervisor is involved in the operation in most applications by providing the drone directional commands. Although there have been various human-drone interface methods such as direct physical control, remote controllers, smartphones, tablets, these methods are not effective for novice users in controlling the drone. Thus, it is imperative to develop a Human-Robot Interaction (HRI) strategy that is effective and intuitive for novice users in order to increase the user adoption rate as well as to fully leverage the recent advancement of UAVs. To this end, this paper aims to explore hand-gesture based HRI methods to control UAVs, which is more natural and easier for non-expert users.

2 Background

Traditionally, hand-held remote controllers, smartphones, or tablets were commonly used to pilot UAVs. These interfaces to control UAVs often have remained unintuitive, mandating users to get extensive training. Also, it continuously occupies both hands of the users. Thus, a recent approach suggests more intuitive and natural human-drone interaction by means of gestures. Gesture-based interaction will allow users to perform other tasks instead of continuously controlling the drone by releasing their hands. Such drone control methods give basic commands such as “go up”, “go down”, “turn left”, “take off” by recognizing the user’ hands, arms, or the entire body via various sensor information [1]. For example, the work of [2] used Microsoft's Kinect™ version 2 sensor to extract the user's skeletal features for both static and dynamic gesture recognition. However, such depth-based methods using multiple body parts put limitations on user location with respect to the camera so that the body part is fully presented. This may not be effective for a dynamic environment like construction. Also, RGB images are commonly used to recognize gestures [3]. However, such RGB data are prone to issues such as variation in illumination. Thus, in this study, we explore leap motion sensors to recognize hand-gestures. Leap motion sensor is widely used for gesture recognition because it can track any motion made by fingers and hands within 1m hemispherical space and outputs the data robustly under any ambient lighting [4].

3 Proposed Human-Drone Interaction System

The proposed system is to have a gesture-based drone control system where the user can send commands to the drone via simple hand gestures using leap motion sensor to start and fly the drone, to turn the drone left or right (yaw), and to land the drone. In our project, we use the Leap Motion Controller [5] as our input device for the gestures and use the PyParrot API [6] to control a Parrot Anafi Drone [7], both of which are connected to the same computer.

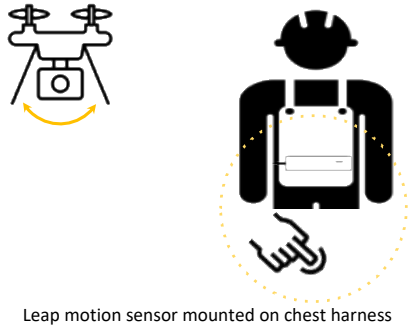

Figure 1: **Overview of the Proposed Human-Drone Interaction**

3.1 Leap Motion Controller

The Leap Motion Controller is an optical hand tracking module, which can be used to track and record detailed and accurate movements of the hand and fingers, at up to 290 frames a second with capable hardware. This data can be acquired and used using its API which has bindings to multiple languages including Python. In this project we use the API version 3.2 with binding to Python 2.7 [8].

The LMC sensor will be attached to the user's chest harness which is commonly used by drone controllers. This allows the user to make hand gestures right in front of them while monitoring the video feed which is also usually right in front of them.

3.2 Parrot Anafi Drone

The Parrot Anafi Thermal drone is a professional 4K HDR compact drone used commonly for aerial photography and scouting. It also has developer support through its Olympe SDK as well as 3rd party APIs such as PyParrot [6]. We chose the PyParrot API for its ease of use and support for Python in Windows and Mac machines as Olympe SDK only supports Linux machines. Using PyParrot API for Parrot Anafi required minor modifications in the API because the API did not support Parrot Anafi drone.

3.3 Hand-Gestures

In our project we implemented a set of basic gestures to take-off, turn, and land. The take-off or fly command is given by a “thumbs up” gesture held for 2 seconds in front of the Leap Motion Controller (LMC) sensor. Similarly, the “thumbs down” gesture will send the command to land the drone from its position.

We also implemented the “swipe” gesture to turn the drone left or right (yaw). Swiping across the LMC sensor from the right to left will turn the drone right by 30 degrees and swiping left to right will turn the drone left by 30 degrees. When there is a video feed on the monitor in front of the user, it is more intuitive for the user to swipe right to see more information on the left and vice versa. This is why the command is designed to turn the drone the opposite way of the direction of swipe.

Each of our gestures has a 2 second delay built into it along with clear messages to the user on what the sensor sees, the commands sent to the drone, and the status of the drone while commands are executed. There is also a 2 second timeout in the sensor after every command to give the entire system enough time to work around any network delays which may occur with the drone and the computer.

3.4 Setup

The setup we used for our project consisted of a Windows computer with the LMC sensor connected to it using the Leap Motion API V3.2 although PyParrot API supports Windows, Mac, and Linux. The Parrot drone was connected to the computer via WiFi using the PyParrot API modified to support Parrot Anafi. The code for data collection from the LMC sensor was written in Python 2.7 which actively communicated with the PyParrot API written in Python 3.7 to control the drone in real time.

4 Current Work

4.1 Gesture Capabilities

The system we created for this project currently supports 2 different gestures, the thumbs up/down gesture and the swipe gesture. Similar gestures can be added to the code easily by simply adding control statements for specific directions to these gestures.

4.1.1 Thumb Up/Down gesture. The LMC sensor identifies thumbs up/down by a series of checks as follows: a) All fingers except the thumb is closed completely, b) The thumb is fully extended, c) The direction of the thumb is towards the positive Z-axis (for thumbs up) a negative Z-axis (thumbs down).

These series of checks prevent accidentally engaging one of the commands, as well as making the gesture specific to the direction they are shown. Hence if a new gesture to move the drone right or left needs to be added, the developer only needs to add control for the positive and negative X axes, such that “Thumbs right” and “Thumbs left” correspond to move right and move left.

4.1.2 Swipe gesture. The swipe gesture is one of the pre-defined gestures that the LMC supports. Each frame the sensor records, it compares it with the previous frame to see the direction of movement of the hand. After a couple of frames, if the speed and direction of the movement is consistent, it is considered a swipe gesture and stores its related information (such as speed, direction, state, etc).

In our program we check if this swipe gesture takes place for more than 0.5 seconds consistently and then engage its related command (turn left/right) accordingly. The 0.5 second ‘confirmation’ time worked out as a long enough duration to prevent false positives and a short enough duration to identify the gesture for true positives without the user having to think too much about it.

4.2 Flight Capabilities

The Parrot Anafi drone through the PyParrot API allowed us to program take-off, landing, turns, and movements. Thus, the commands from the gestures are mapped to the appropriate commands and are executed after explicitly displaying the command to the user as an acknowledgement. The drone was able to take off, wait for commands, turn in either direction and land when the commands were given.

Although we tried to test our code indoors, Parrot Anafi tends to drift indoors because it cannot update its GPS location correctly and the indoor ambient environment.

5 Challenges

Our initial idea for the project was to have a system which can control the drone’s entire camera system through the LMC sensor while the flight was controlled by a dedicated pilot. We quickly realized that this and many other ideas we had could not be done due to the limitations of the hardware and its supporting APIs, as discussed below.

5.1 Leap Motion Controller

The Leap Motion Controller is a sophisticated tracking device which has a wide variety of applications it can be used for. But it does come with its drawbacks.

5.1.1 Tracking in very bright lighting conditions. The Leap Motion Controller works ideally in indoor lighting. In environments with very bright lighting or a lot of sunlight (Even if the sensor itself is not in direct sunlight), the accuracy of the sensor drops and the sensor goes into "*Robust Tracking mode*" [9] which has increased processing latency and loss of data from very fast motions, in exchange for being able to give some level of data in a wider range of IR lighting conditions.

We found that the effects of this robust tracking mode negatively affected our gesture recognition, especially the swipe gesture. We also noticed that the orientation of the sensor, which defines the directions of positive Z and X axis directions, is decided based on which directions the hands in the frame are facing [9]. Since tracking was affected by bright conditions, the sensor could not decide the correct orientations, which led to the sensor mis-identifying thumbs up and thumbs down.

Both of these issues can potentially be solved by forcing the sensor to never go into robust tracking mode and never change the orientation of the axes. It worked as predicted indoors, but we could not perform field testing on these settings to verify or validate it due to lack of time.

5.1.b Leap Motion API. Official support for the Leap Motion API and its bindings to high level languages to Python has been entirely stopped. The last supported Python version is 2.7 which breaks compatibility with Python 3 and above. Although there are some Python 3 wrappers and implementations from the community, these are not reliable in getting all data correctly and most do not work. Thus, if we require bindings to python 3, we need to write it ourselves.

This issue also presented another hurdle in our development. We could combine the sensor's API and the drone's API, neither could we efficiently transfer commands and data between the two APIs. Our solution was to store the commands in a file locally and read it from the other program. This adds unnecessary latency and possible points of failure to the system.

5.2 Parrot Anafi

5.2.1 WiFi connections issues with computer. The Parrot drones have the ability to connect with a computer using WiFi. But this connection is not always reliable. There are a few instances during our development where the drone will be connected to the PC and working fine, but would stop accepting commands and sending acknowledgements back to the computer. Resetting the WiFi connection, resetting the drone, and establishing a new connection seems to be the only way to get it back to a working condition.

5.1.2 API for the Parrot Anafi. A major challenge we faced is the limited support by Pyparrot API for the Anafi drone. The official SDK for Anafi drones from Parrot was only released in May 2020. Pyparrot API, being a 3rd party API does not officially support Anafi drones yet. Although basic drone command functions were applicable for Anafi drones, camera-related functions were not working.

From our testing, it was not possible to control the camera gimbal or receive video feed from it. As a work around, we had to turn or move the body of the drone in order to change the camera view instead of changing the gimbal. The video stream function will be implemented when it is supported by Pyparrot API. Another possible measure is to use the official SDK from Parrot although it is supported only on Ubuntu (Linux) and not on Windows or MacOS.

6 FUTURE WORK AND IMPROVEMENTS

6.1 Testing

In the time we had to work on this project, we conducted a limited number of field testing which brought up a lot of issues with the code as well as raised some questions on the capabilities of the hardware. More and frequent field testing of drone's flight, gestures in different lighting conditions, and handling of false positives and negatives in gesture recognition is necessary to identify the right parameters, settings, and environment to use this technology in the future, as well as to ensure a level of robustness to the system. In future, we can further test commands such as moving forward/backward or left/right as we planned to implement them along with some gestures mapped to it.

6.2 More and better gesture support

As testing showed, the LMC sensor is limited by the lighting condition but it also gives the options to work around it manually. The sensor's ability to identify gestures without requiring machine learning or high computation shows promise for future work to include more gestures with more complexity, although it may require conducive lighting conditions. Adding support for a "Key-tap" gesture to go to a certain location in space, pinch to zoom in/out on the video feed, and a "circle" gesture to engage a custom scouting routine are just a few of the possibilities with this gesture-based-control technology.

6.3 Directions to Drone via Hand-gestures

The proposed system in this project can only provide limited navigation commands but cannot provide spatial information to the drone. The hand-gesture command combined with the video feed can be further processed to give information of target location to the drone so that it can fly to the pointed designated location, or perform a task based on that location (such as a panoramic scout of a building). This may require higher computation, but will improve the user experience in the field.

7 Conclusion

Despite its drawbacks and limitations, the LMC is still a very sophisticated and reliable sensor for gestures under conducive environments. Since gestures are part of how we communicate, it makes sense to design this form of communication with robots as

well. This form of gesture-based-control system demonstrated here shows a lot of potential in improving the usage of UAVs and robots in the field and to improve the standards of human-robot interaction.

REFERENCES

[1] B. Gromov, J. Guzzi, L. M. Gambardella, and A. Giusti, "Intuitive 3D Control of a Quadrotor in User Proximity with Pointing Gestures," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, May 2020, pp. 5964–5971, doi: 10.1109/ICRA40945.2020.9196654.

[2] G. Canal, C. Angulo, and S. Escalera, "Gesture based human multi-robot interaction," in *2015 International Joint Conference on Neural Networks (IJCNN)*, Jul. 2015, pp. 1–8, doi: 10.1109/IJCNN.2015.7280540.

[3] A. Cosgun, A. J. B. Trevor, and H. I. Christensen, "Did you Mean this Object?: Detecting Ambiguity in Pointing Gesture Targets," p. 8.

[4] B. Hu and J. Wang, "Deep Learning Based Hand Gesture Recognition and UAV Flight Controls," *Int. J. Autom. Comput.*, vol. 17, no. 1, pp. 17–29, Feb. 2020, doi: 10.1007/s11633-019-1194-7.

[5] "Tracking | Leap Motion Controller | Ultraleap." https://www.ultraleap.com/product/leap-motion-controller/ (accessed Dec. 13, 2020).

[6] amymcgovern, *amymcgovern/pyparrot*. 2020.

[7] "Parrot ANAFI - The compact and resistant drone with a 4K HDR camera." https://www.parrot.com/us/drones/anafi (accessed Dec. 14, 2020).

[8] "Python SDK Documentation — Leap Motion Python SDK v3.2 Beta documentation." https://developer-archive.leapmotion.com/documentation/python/index.html (accessed Dec. 14, 2020).

[9] "Using the Leap Motion Control Panel — Leap Motion Unity SDK v2.3 documentation." https://developer-archive.leapmotion.com/documentation/v2/unity/supplements/Leap_Application.html (accessed Dec. 14, 2020).