Drone Motion

Piloting a Drone Using Hand Gesture Detection System

Hakan Erol Joji Asako Soheil Karimi Trivi Tran

Table of Content

1. Abstract:

The current methods of piloting drones are difficult and not easily accessible. The average person might have an interest in drones, but does not want to spend time going through the learning curve. This exposed a void which could be filled. We present a sensor-free, accurate, and intuitive system for hand motion controlled UAV piloting. Our method efficiently classifies hand gestures from simple video capture, translating them to drone motion. Our implementation takes advantage of the strength of deep convolutional neural networks (CNN) to achieve high accuracies. In order to eliminate discrepancies due to different people and backgrounds during classification, a simple calibration technique is used to maintain reliability. Furthermore, a Discrete Bayes Filter is used to sequentially process frames and estimate the state of the system. We hope our model will serve as a platform for people to easily and intuitively fly drones.

2. Introduction:

Drones are a recent new invention that are being used for a variety of applications by many people for such as capturing images at a high level above the ground, delivering or moving stuff in air, or even for fun. However, in order to fly a drone, the user have to go through a learning curve before they can fly a drone properly. The amount of time it takes a person to fully be able to control a drone would depend on the individual, and for some people that is not convenient or even possible to spend that much time. Especially, for the individuals who are pre-teens or seniors, they might not be able to fly a drone due to the difficulty of it. Therefore, our goal is to invent a system that would simplify this step and allow everyone at all ages to fly the drone easily. This system would primarily aim for beginner who has no prior experience flying a drone. Also we want to target other groups of people as well. We want people who are interested in virtual reality, video games or sci-fi movies enjoy using our new product. We want to invent a system that is one step closer to the future and is more interesting to use than traditional remote controllers.

A solution to the problem that was explained above is to use the hand movements of a person to control the drone instead of using a remote controller. The remote controller normally has many buttons which would easily confuse a beginner. On the contrary, it is easier and much more enjoyable to use the hand movement to control it. Therefore, we created a system that will convert the hand gestures of the user into the flight motions of the drone. This system would capture the hand gestures of a person who stand in front of the camera and use that hand gestures to command the flight of the drone. In order to detect and differentiate the hand gestures of the user, we used the state-of-the-art deep learning method (classifier) which will label the images of hand movement from the camera and use that label to command the drone.

In order to create the classifier that would accurate label the image with a correct movement, we had to gather a lot of images on different people and different movement and use these images (our dataset) to train the model. By looking at different pictures of different moves, the models starts classifying different pictures, and the way it does the classification is that it tries to look at similar pictures of a move and finds specific details that separates the move from other moves. By looking at specific details in each picture it recognizes if the picture is similar to the other pictures it has trained on and tries to match the given picture with a specific move. We also improved the accuracy of the classification by adding Discrete Bayes Filter on top of our classifier. The filter would eliminate any noise (misclassified movements) created from the classifier output. We use the output generated from model to command the drone using an API called PS-Drone. In this project, we have successfully implemented the following tasks:

- Create a classifier, using a Convolutional Neural Network, to classify different movements.
- Implement the Discrete Bayes Filter to improve the accuracy of the predicted move.

- Using the PS-Drone API to fly the drone via a laptop and ultimately receiving input from the model and sending move commands to the drone.

- Connect the classifier together with PS-Drone to pilot the drone with the hand movement.

- Optimizing the flight motion.

3. Group Management:

- TriVi Tran - t1tran@ucsd.edu
- Hakan Erol - herol@ucsd.edu
- Joji Asako - jasako@ucsd.edu
- Soheil Karimi - skarimik@ucsd.edu

In our group we don't necessarily assign exact roles to members, instead we all collaborate on each aspect of the project ensuring each and every one of us understands how everything works. Keeping the same philosophy, we mostly make decisions as a whole. There are cases however, where one member might be much more knowledgable on a certain topic and we will trust their judgment on certain decisions. Still though, the rest of the group would be consulted at first so everyone is on the same page. We meet every Monday, Wednesday, and Friday to spend a few hours looking at our progress and deciding how and what we should proceed to work on next. Outside of regular meeting hours, we stick to Slack for communication. Each week we assess our progress, taking a look at our specific goals and the big picture, ensuring we are on track. If we see that we are a little behind, we will meet on extra days of the week to get back on track.

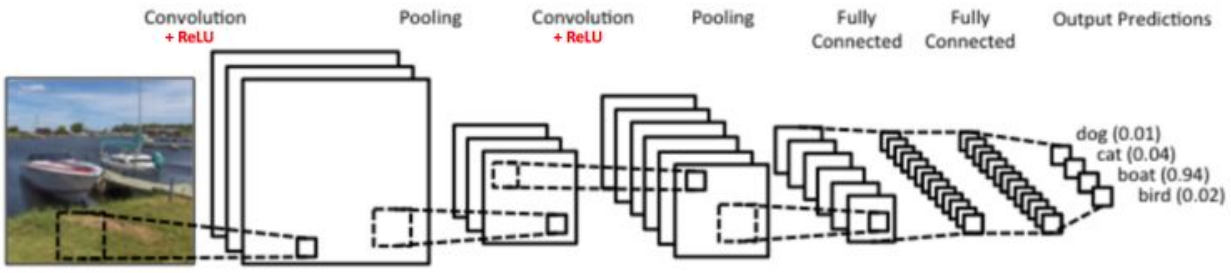
4. Technology:

- AR Drone 2.0 Elite Edition - A standard affordable product made by Parrot. We chose to use this drone because it was able to do all simple flight movements, which met our requirements for this project. The drone is inexpensive which would be appropriate for the beginner or anyone who have not fly a drone before. This would allow the user to experience how to fly the drone without worrying too much about damaging the drone.

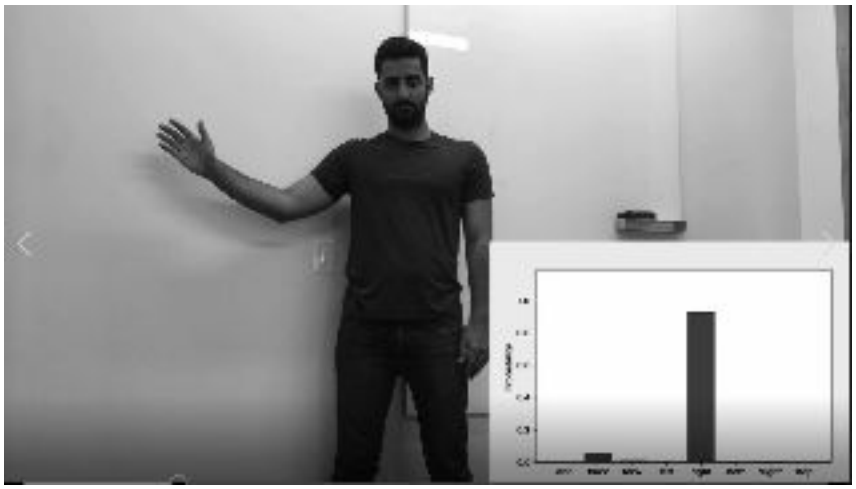
- PS Drone API - An API written in python, specifically for the AR Drone 2.0. It allows us to control the flight movement of the drone easily and to use it along with our machine learning classifier. The API is programmable and allow us to improve the flight movement by adjusting the speed of the drone with different parameter.

- Convolutional Neural Network (CNN) - A state-of-the-art deep learning method or a classifier that was used to predict the flight commands. CNN, in general, has been widely used for image and video recognition. It mimics the way of how the brain learning new things that we see with our eyes. The CNN has many layers in between the image and the output. Each layer will have a different representation of the image. As the image go through all the layers, the CNN will be able to learn and identify the most important features for each movement/output and will be able to label the image with the correct movement. The classifier requires a learning process before it can actually predicts a command. It first needs to process the images of each hand gesture, captured by the webcam, and update the belief of each movement. Then, it will try to guess the flight commands upon a new image that it receives. This allow us to use our hand gestures and translate them to drone flight commands. Our CNN has 18 layers and was pre-trained with millions of general images. We, then, train the model with our hand gesture images, which include several images of each movement (move right, move left, move forward, move backward,

rotate right, rotate left, stop, and landing). By training the model with these images, the model can now recognize each movement accurately when there is a new unlabeled image feeding into the model. This allow us to feed the images captures from the camera into the model and acquire the labeled movement/ output from the model to use to command the drone. The picture below is a simple diagram demonstrating how CNN work. Note that is diagram only show a few layers in order for reader to understand the workflow of CNN.



- Discrete Bayes Filter - a mathematical algorithm that updates the belief of the current movement using the belief of previous movements as a hint. This allows us to improve prediction accuracy before sending the actual command to the drone by taking into account the sequential nature of video frames. The algorithm also helps eliminate some noises from our classifier's predictions. For example, when we have multiple continous frames(images) labeled with the same movement but one image was suddenly misclassified by our model and was labeled with a different movement, we can consider this as a noise to our the classification. Discrete Bayes Filter update the belief of each movement as they receive an output from CNN model. The belief of one movement will increase as it receives the same movement again and again. When it receive a different movement suddenly, what it has believed on will be changed slightly and will not impact the result of the whole system. The picture below demonstrates how Discrete Bayes Filter work with our CNN model. We can see the the probability of right is very high due to the multiple continous frames that were labeled with ‘move right’. Any misclassified movement will not impact the performance of our system.



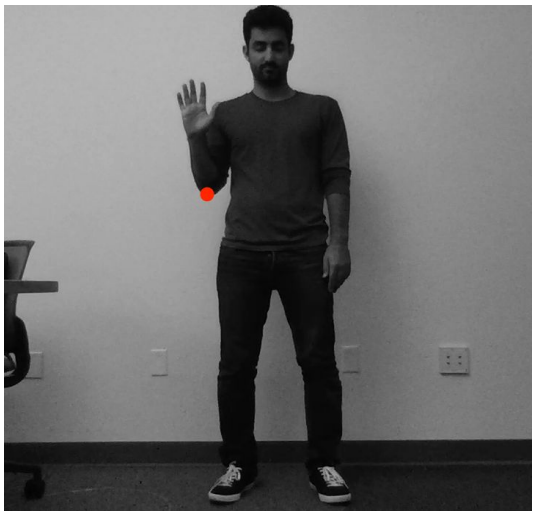
5. Milestones:

I. Hand Gesture Detection:

The most important element of our project is the hand gesture detection system. The goal of this system was to be able to detect and classify our hand gestures in real time. In order to do this, we started out with a small pretrained network model called Resnet 18. We used this in conjunction with a deep learning framework called PyTorch to implement our own network. The network we started with had previously been trained on millions of images, so it was fairly good at tackling image classification problems. However, we had to take extra steps in order to make the model work for our hand gestures. To train the model to recognize what we wanted, our team captured many video samples of the hand gestures to create a dataset. We first trained our model on these captured images in order for the network to learn what our hand gestures look like. When training at a specific location and on a specific person, test results were very good when testing was performed at the same spot with the same person. When we would move to a new location or switch the person testing it out in real time, our results would vary a great deal. This showed us that our model had not been able to generalize to only the movements, and still gave weight to things like the background and the person's clothing. Initially to overcome this issue, we introduced a calibration system. This had the user perform all of the different hand gestures at their current location, so the model could adapt to that exact environment. This didn't take too long, and greatly improved our testing accuracy. Although we were pleased with this result, we were not fully satisfied with it. We didn't want to force users to train every single time before using the system, but rather have the model be general enough to work in all locations and on all people. In order to do this we knew that we still had to gather lots of data in different environments in order to keep training our CNN and to increase its accuracy and generalization.
After training, real time testing can be performed easily by performing the hand gestures at the desired location. We made a testing environment in which the computer says out loud which gesture is being classified, so that we don't have to risk our drone during the testing process.

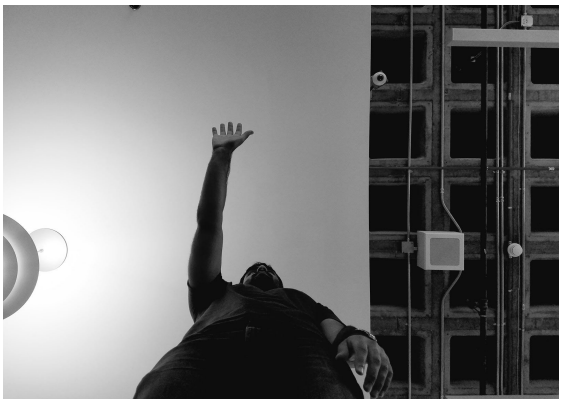
- **Final Adjustment**

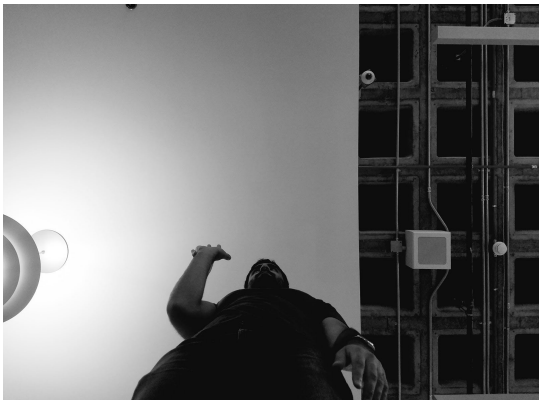
  There were two problems that we encountered when the camera is located in front of the user:
  1) Our CNN model could not differentiate the gesture of forward and backward.

  2) It requires calibration step before using the system. We overcome this challenge by changing the angle of the camera, which means the camera will be located on the ground and let it looks

up. This change will fix the problems that we had and it also make our model become general, which means a random user can walk up to the camera and control the drone without going through the calibration step. However, we would recommend the user to go through the calibration step because it improves the performance and giving our model an opportunity to become better by training on different person at different location.





- **Deliverables**

1. Sample Images of Hand Gestures with camera angle in front of a person - A set of sample images of different hand gestures for visualization. This gives the audience a sense of what the CNN is training on. These are the images of hand gestures before we change the angle of the camera. (show above)
2. Sample Images of Hand Gestures with camera angle from bottom-up: These are the images of hand gestures after making the final adjustment to this part of the project. The CNN perform really well with this new adjustment.(show above)
3. Dynamic Visualization - In real time, we will demo the classifier's probability distribution of our hand gestures with easy to understand graphs

II. Piloting UAV:

We used a fairly inexpensive drone for testing. The AR-Drone 2 can be controlled through a full featured API, written in and for Python, known as PS-Drone. We connected to the drone through a laptop and control it with this API. We tested all the drone movements we were planning to implement. Then we compared the piloting performance of the computer to the mobile app provided by the drone's manufacturer. Using the PS-Drone API, we tested and improved the piloting of our drone through commands sent by a computer to the drone. We wrote many scripts that send a variety of commands in order to understand how the drone reacts, allowing us to optimize how and when we send these commands. The goal was to achieve accurate flight with smooth transitions between motions.

- **Deliverables**

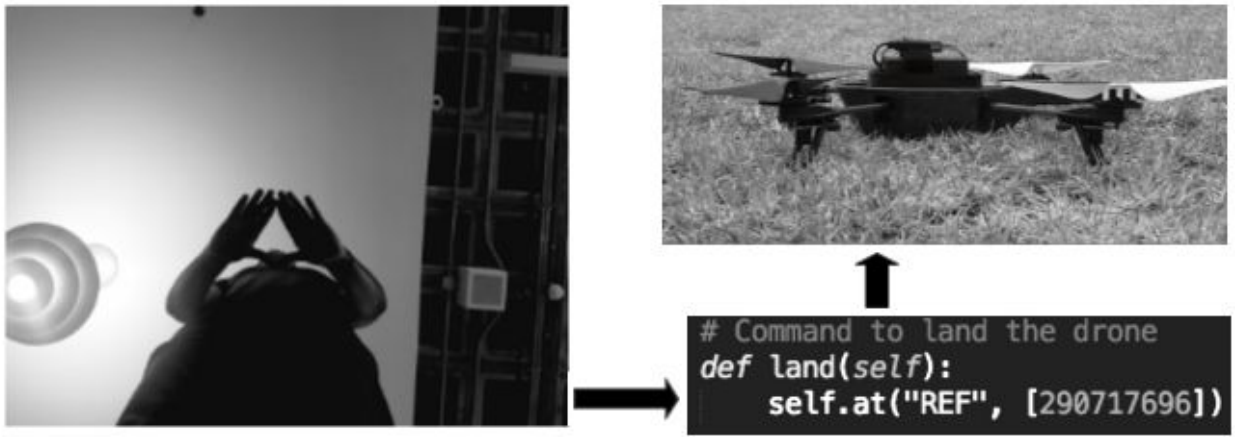
Observe how in the video displaying flight with the drone's native app, the movements are jerky and hard to control. With PS-Drone, although the drone crashed due to human error, the movements of the drone are clearly much smoother and flow together much better. The main advantage of using the PS-Drone API is the fact that the drone actually moves in the direction

you tell it to, where as with the phone it is quite out of control and has a mind of it's own. This along with the much smoother flight makes it a viable solution for our system.

III. Connecting the Systems:

In order to reach our end goal of actually controlling the drone with the movement of the hands, we need to connect the detection system along with the PS-Drone API. We will integrate the API calls into the detection system so that the system is as intuitive as we want it to be. Given a classified hand gesture from our CNN model, the appropriate command will be sent to the drone.

- **Deliverables:**
    - Connecting the systems in order to achieve the desirable outcome.



IV. Optimizing Flight:

To improve the fluidity and intuitiveness of piloting the drone, we spent time tweaking certain aspects of PS-Drone API calls. We changed certain parameters such as flight speed, transitions between motions, etc. in order to make the user feel like the drone is truly connected to their hand gestures, and to make the experience as enjoyable as possible.

- **Deliverables**

  Achieving fluid movement, Easy control and safe movement by restricting the given input from PS-Drone API to the Drone.

    - Fluid movement: Using efficient and fast algorithms to generate a move and send it to the drone for fast and fluid reaction.
    - Easy control: Filtering some of the moves in order to reduce the complexity of piloting the drone.

- Safe movement: Maintaining a moderate constant speed that would prevent the drone from crashing to people or objects with high velocity.

6. Conclusion:

We have successfully created a system that can accurately classify the hand movement and use them to fly the drone. We have successfully connected all the systems and made them all to work as a unit while they stay efficient and productive. This system allows the user to easily fly the drone and enjoy the experience as much as possible. The moves are very intuitive so the learning curve is eliminated for new users and they can start controlling a drone immediately. With this new technology instead of the traditional technology (remote controllers) a bigger range of people are finally able to pilot a drone without being trained on how to do so. In the future, we plan to continuously improve the model by training it on different people. We also plan to integrate this project into a phone platform and expand the program so it would work on most types of drones.

Currently our system has only been running on our laptops, using the webcam in order to capture hand gestures for classification. This means that for testing including at outdoor locations, we had to bring our laptop to run our system. This is obviously not the most convenient thing, and as an end product we would not want to release it this way. In order to make our system portable and an convenient as possible, we want to port it to mobile phones. This means that we would keep the same exact functionality, but all in the palm of your hand. Since our updated system works from the ground up, meaning that the camera is placed underneath you and pointing at the sky, this also makes it more convenient for the user. With our initial setup, we would have to find an elevated surface to place the laptop on in order to capture a person's body in the camera's frame. With the updated setup, we would place our laptop on the ground, with the webcam pointing to the sky. This is also not desirable of course. With a phone however, you can go anywhere and place the phone at your feet and begin piloting the drone with your hand gestures. There are deep learning frameworks that allow you to easily transfer existing models you have made on a computer to a mobile platform. In the coming months we will surely look into this and hope to make some good progress.