Obtaining Real-Time Sensor Data from Unmanned Aerial System

Project Mid-Progress Report

Vlad Dubrovenski and Han Zhou

GitHub Link:

https://github.com/vladi7/DroneVideoRecognition

Video Link:

https://umkc.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=2d51dfea-0dd9-46ab-af13-ac6a01201b04

1. **Introduction**
   Unmanned Aerial System, or simply drone, have become an important tool in various industries. For example, in agriculture, the drones can be utilized to collect the data about the crops and even use the drones to apply various chemicals to crops in order to improve the efficiency. Another big industry is public safety, where law enforcement, fire departments, and other teams are using drones for the past several years to collect the data and improve the safety of team members. The delivery is another area where the drones can be utilized, and Amazon is currently testing drones for delivery of small-sized packages.
2. **Motivation**

   Our motivation for this project is to explore various parameters that can be collected from DJI Tello drone. This is a simple a drone that, meanwhile, has many sensors which makes this drone very robust. For example, one can land it on a hand, because the drone has the photo sensor that can detect the palm of the hand. Also, this drone has a few stabilization module which makes it extremely hard to crash it even for a novice user.

   We believe this drone is ideal for testing as it is extremely robust and hard to destroy. Also, this drone has a nice programming API that we can use to both control it and retrieve the information from most of its sensors. And, of course, it is relatively inexpensive, so, we don't have to worry too much even if we crash it, which actually happens quite often since it is usually flown inside the apartment.

3. **Challenges**
   We encountered a few issues while designing the solution for our system. First one was about the connection to computer. It was pretty simple to do. One of the authors already connected the drone with computer over wifi prior, but he used Delphi, which is a low-level language written in C and Assembly (one of the only languages that are not written exceptionally in C). We found it to be very easy to connect the drone to PC with node.js using socket_io library. Another challenge was sending commands from front end to the drone. It was solved by issuing commands through socket to our back end Node.js server which transmitted those commands to the drone via another socket.
   The last challenge for this increment was how to get the data from the sensors. We found out from the documentation that the data can be obtained from another port of the drone, for which we used another socket, and transmitted it to the front end from Node.js.
4. **Architecture**
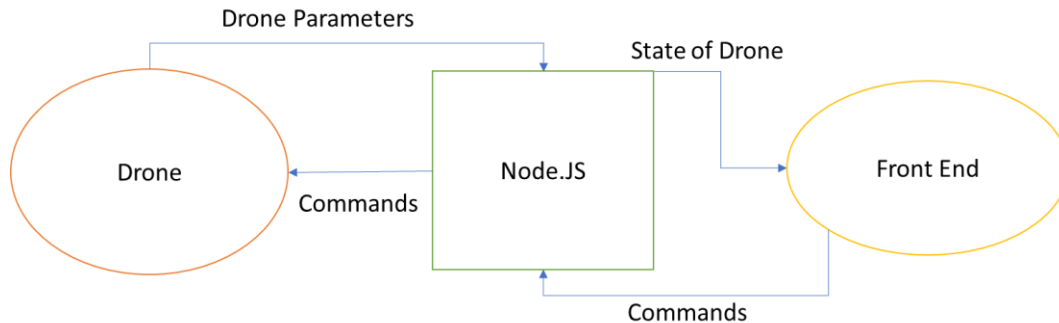
The above figure is the architecture of our project. This architecture will most likely change after the addition of video stream and robustness to improve the UDP reliability from the Application Layer. The scheme has three modules: Drone, Node.JS, and Front End.

- Drone
  Drone has the IP address of 192.168.10.1 and 3 different ports. Port 8889 is used for receiving the commands and replying with either "ok" or "error" with the error message included. There are various movement commands that can be issued. Also, command "streamon" and "streamoff" are used to turn on and turn off the video stream from the drone. The stream can be obtained from the port of 11111. This is our next task. The video stream will also need to be decoded. We also use the set commands mainly to set various speed of the drone (e.g. "speed 100") which is measured in cm/s. We do not use the read commands as all the sensors' parameters and battery level can be obtained from the port 8890. We cannot obtain any connectivity parameters (wifi etc.) from 8890 but they are irrelevant to this project. All of the ports use UDP protocol.
- Server (Node.js + Express.js)
  We created the Node.js server using Express.js for simplicity as we did not need to fine-tune anything in this server. For dealing with sockets, we used the socket_io library. For this project increment, we only use 3 sockets, one to connect to the front end, another one to connect to the drone to control it, and the last one to obtain the information about the drone. For the next increment, we plan to add another socket for video retrieval from the drone.
- Front End
  We decided to use Angular CLI for front end to have some practice with this great framework. We used Typescript instead of JavaScript to gain some experience with it. We connected to NodeJS with socket using "socket.io-client" library. We also designed the front end to be responsive, simple, attractive, and reliable.

5. **Experiment**

For the experiment, we retrieved and processed all the parameters available from the drone. Some of the parameters needed further calculations, like conversion of the Air Pressure from oz/in^2 to mmHg. Horizontal velocity and acceleration where giving in x and z directions, so we used Pythagorean theorem to calculate the magnitude of the velocity and acceleration. This experiment demonstrated that even with a simple and relatively cheap drone we could retrieve a lot of useful information, like temperature and air pressure. This can be used, for example, by firefighters to check the temperature inside the building before going in. Of course, they would have more complicated drones with a lot of sensors, but the goal of this project was to explore the efficient way of obtaining real-time sensor data from the drone.

6. **Conclusion/Future Work**

In this increment of our project, we obtained all possible useful parameters from the drone. We also can have the complete control of the drone from the web front end. We do skip a few irrelevant commands that we plan to add in the future increment. Another goal of the final increment will be to obtain the video from the drone and display it, process it, and display in our front end. Further goal is to make our back end robust and less error-prone. We have tried a few designs, but they all seem to have major drawbacks. We will discuss it further in our final report.