Report Date: 10/21/2022
To: ematson@purdue.edu, ahsmith@purdue.edu, lee3450@purdue.edu
From: The Team Gangsture (Gesture Drone Control)

- Hyeongbin Park (The Leader / mtanger@kw.ac.kr)
- Seunghwan Kim (franz0602@stu.jejunu.ac.kr)
- Yujin Lee (yj5878@kw.ac.kr)
- Soeun Lee (thdms8477@jejunu.ac.kr)

Summary

The draft paper for abstract, introduction, and literature review was written. Each part of development (networking, drone, iOS, and AI) started to develop since all equipment was delivered. There were no issues with the drone, thus, the laptop and the drone had a successful connection. Additional test for cloud and UDP socket was conducted.

What "Gangsture" completed this week:

- **Completed to write the draft paper**
  - Abstract, introduction, and literature review section were written and reviewed by Purdue students.
  - For the literature review section, some papers were summarized by each member. Additional documents would be added to the literature review such as, research about hand recognition and image processing.
- **Trained and modified a deep learning model**
  - Deep learning model was built by two types of libraries, PyTorch and Tensorflow. Each library had its pros and cons [1] so the most compatible library would be selected in the future.
  - Process of preprocessing for image dataset was changed. It contained image resize (360x640 to 64x64), data augmentation (horizontal and vertical), and color channel (gray scale to RGB) [2], [3].
- **Connected the drone to the laptop and ran the code**
  - The drone camera could be checked on the laptop in real time by using Python. The connection used Wi-Fi and proceeded in a local state. The connection did not require a separate device but, there was a delay of about 1 second [4].

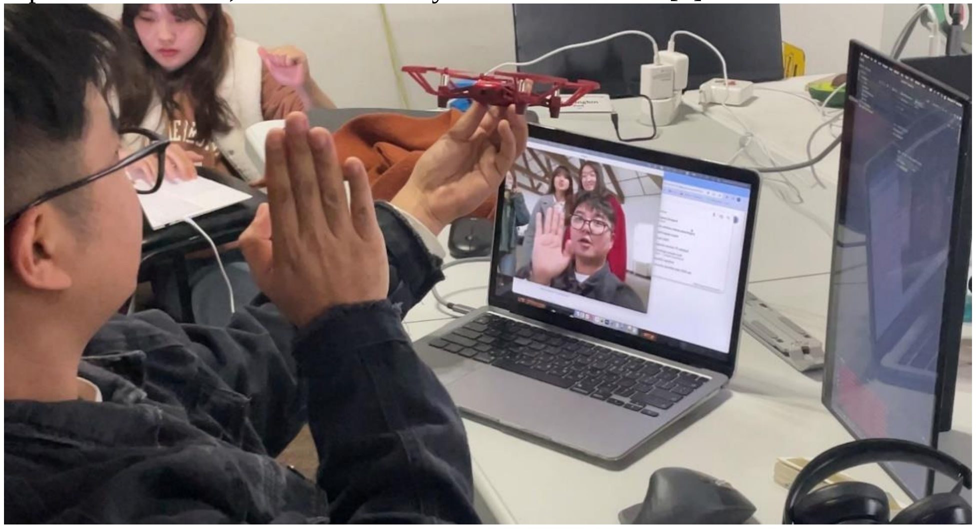
Fig. 1. Successful connection between the drone and the laptop

    - The laptop could immediately check the battery condition of the drone.
    - The drone could record videos and store them on the laptop [5].
- **Searched how to save the video on the cloud**
    - Cloud API was made. After download 'client_secrets.json', the code was executed and a test file was uploaded [6].
    - How to save the video on the cloud automatically was searched. It could be schedule automated backups or on-demand backups [7].
- **Tested the UDP socket using Python**
    - Python codes for server and client were developed and run. There was a delay problem for 3 seconds.
- **Studied and searched about CoreML**
    - Integrated the SqueezeNet model, a pre-trained model provided by Apple into the iOS application.
    - The model was used for tasks like image classification.

Things to do by next week

- Prepare for the mid presentation
- Solve the development environment conflicts
- Modify and look for a better deep learning model or algorithm of hand recognition
- Search about the delay problem of the UDP socket using Python
- Study how drone and iOS application communicate with socket

Problems or challenges:

- **Conflicts between development libraries of AI**
    - Since most AI libraries are developed and updated there are many conflicts problems.
    - To prevent collision between libraries, a new virtual environment that installs only desired libraries would be created each time with Anaconda.
    - PyTorch especially has conflicts with CUDA, GPU, and Nvidia driver version [8]. It needs another way to use PyTorch such as, using a cloud server or training only with CPU [9].
- **Upload images taken by the drone to Google drive using the network**
    - The drone cannot connect directly to the Internet so, the laptop receives video from the drone, stores them separately, and uploads them to the cloud.
- **"Failed to prepare device for development" error occurred when the iOS application is about to run**
    - The cause was an iOS version problem.
    - The latest iOS version has been updated so, Xcode and MacOS had to be updated and it took time.

References

[1] *Introduction to project code examples*. [Online]. Available: https://cs230.stanford.edu/blog/tips/#tensorflow-or-pytorch-. [Accessed: 21-Oct-2022].

[2] *Classifying images of hand signs*. [Online]. Available: https://cs230.stanford.edu/blog/handsigns/. [Accessed: 21-Oct-2022].

[3] F. Borba, “Tutorial: Using deep learning and CNNS to make a hand gesture recognition model,” *Medium*, 08-May-2019. [Online]. Available: https://towardsdatascience.com/tutorial-using-deep-learning-and-cnns-to-make-a-hand-gesture-recognition-model-371770b63a51. [Accessed: 21-Oct-2022].

[4] Damiafuentes, “DJITelloPy/record-video.py at master · damiafuentes/djitellopy,” *GitHub*. [Online]. Available: https://github.com/damiafuentes/DJITelloPy/blob/master/examples/record-video.py. [Accessed: 21-Oct-2022].

[5] Kinivi, “Kinivi/Tello-gesture-control: Control DJI Tello using hand gesture recognition on drone`s camera video-stream. feel free to contribute!,” *GitHub*. [Online]. Available: https://github.com/kinivi/tello-gesture-control. [Accessed: 21-Oct-2022].

[6] “Cloud apis google cloud,” *Google*. [Online]. Available: https://cloud.google.com/apis. [Accessed: 21-Oct-2022].

[7] “Create and manage on-demand and automatic backups | cloud SQL for mysql | oogle cloud,” *Google*. [Online]. Available: https://cloud.google.com/sql/docs/mysql/backup-recovery/backing-up. [Accessed: 21-Oct-2022].

[8] Pytorch, “IMPORTERROR: DLL load failed: The operating system cannot run %1. · issue #7579 · Pytorch/Pytorch,” *GitHub*. [Online]. Available: https://github.com/pytorch/pytorch/issues/7579. [Accessed: 21-Oct-2022].

[9] “Google colaboratory,” *Google Colab*. [Online]. Available: https://colab.research.google.com/drive/1QnC7lV7oVFk5OZCm75fqbLAfD9qBy9bw#scrollTo=Fj9YcAnsT4B_. [Accessed: 21-Oct-2022].