

EE/CprE/SE 492 Weekly Report

2/27 - 3/12, 2026

sdmay26-20

AI Integrated Drone for Smart Crop Health and Growth Monitoring

Faculty Advisor/Client: Md Maruf Ahamed

Team Members:

- Jongmin Lee, jongminl@iastate.edu - Frontend/AI
- Z Harvey, zdharvey@iastate.edu - Frontend/Backend
- Andres Lopez, andresl4@iastate.edu - Hardware/Firmware
- Alex Pallan, apallan@iastate.edu - Cloud/Security
- Camden Senneff, csenneff@iastate.edu - Frontend/Drone SW
- Caleb Zea, calebzea@iastate.edu - Frontend/Backend
- Srdan Kopunovic, srdank@iastate.edu - Backend/Cloud

Summary of Work:

- Frontend
 - Finished frontend -> backend basic API integration
 - Fixed user sessions persisting across app launches
 - Made first draft of drone -> frontend -> backend image transfer
 - Finished field saving logic and API connection with backend
 - Created page for user to see and select their saved fields and option to create new one
 - Expanded the existing grid logic to find intersection points within the grid and save those latitude and longitude values for the drone to takes pictures.
- Backend
 - Completed and tested all API functios to handle GET, POST, and DELETE requests
 - Tested the transfer in both local and server environments
 - Added CRUDL operations to Postman collection for manual testing
 - Removed backend program system process configurstion
 - Added docket process to run backend program
- Cloud
 - Awaiting completed backend framework to be dockerized and deployed on AWS EC2 service. Until then, a terraform framework and deployment pipeline is ready
- Drone
 - Installed a new, smaller lens to significantly reduce the weight of the drone.

- Implemented a mock UART over TCP controller in the code which allows us to run simulated flights against INAV.
- Updated Raspberry Pi images to include the necessary dependencies and turn off unnecessary hardware for lower power draw. We now have a functional Raspberry Pi image that can be flashed to an SD card and immediately runs our software on boot, and is formatted as required for the frontend to read data from it.
- Images are now on the SD card ready for the rest of the system
- Started registration process with FAA in order to fly legally
- AI
 - Developed SLURM pipeline on the NOVA cluster to automate downloading and processing the dataset.
 - Upgraded the *auto_labeler.py* script with regex-based date parsing to handle varying filename formats across 2017, 2018, and 2019 data.
 - Added a biological "dirt filter" using OpenCV to eliminate false positives, ensuring tire tracks and bare soil are not mislabeled as mature corn.
 - Processed and labeled over 30 flights from 2017 and 2018, resulting in 9GB of training data.

Plans for Coming Week + Action Items:

- Jongmin Lee - The primary focus for next week is to finish processing the 2019 dataset and begin training the initial YOLOv8 model. Now that the labeling logic is robust, I will run the remaining 2019 links in small batches to stay within storage limits. Once the full dataset is unified on my local machine, I will organize the directory structure into the required YOLO format (train/val/test) and write the training script to run on a GPU-enabled node on the cluster.
 - Action Item: Verify the 2019 output quality to ensure the larger file sizes do not require further adjustments to the tiling logic.
- Z Harvey - Rigorously bug-test the new page and field functionality, and see if there are any ways for the user to get info from the backend they shouldn't be able to see. Then see what next steps are needed for frontend development.
- Andres Lopez - Look over and work on MSP for communication between Pi and FC
- Alex Pallan - Ensure the legality of operating the drone in local airspace, from registration to potentially getting permission from ISU/the FAA/any other authorities

- Camden Senneff - Make and solder on power and UART connectors to the Pi so we can power the Pi from the drone and make sure the communication link works. Over break, I will get the INAV flight simulation feature linked up to a flight simulator game so we can visually see what our drone will do.
- Caleb Zea - make sure image transfer works between drone -> frontend -> backend
- Srdan Kopunovic - Setup up fully functioning Continuous Deployment for .NET system so newest version is ran on remoyr server after completing GitLab pipeline. Additionally, need to created a test database that will be used for an integration test stage that will be added to pipeline setup. This will be used to test a real PostgreSQL database, as well as ensure all the API responses are behaving correctly.

Pending Issues:

- Jongmin Lee
 - The main bottleneck remains the strict 10GB disk quota, which requires manual intervention to clear space between flights. Additionally, the logic error in "Label 5" (Senescence) requires further attention; while the code has been patched to filter out non-biological noise such as tire tracks, the previously processed 2017 and 2018 datasets must be rerun or manually cleaned to purge the remaining incorrectly labeled images. This re-processing is necessary to ensure the final training set is biologically accurate and balanced.
- Z Harvey
 - Sometimes the API calls don't work and return an error, I'm not sure if this is due to something in the backend or frontend.
- Andres Lopez
 - How does MSP work
- Alex Pallan
 - Drone is unregistered and must broadcast position and identification information to ensure legality with federal authorities, and potentially not violate the ISU office of risk management's policies.
- Camden Senneff
 - Will need to calibrate the simulator to closely match our actual drone's specs for it to be effective, and it is only useful to an extent (it can test our drone controller software, but we can't test our real INAV configuration until we fly it).
 - Full drone simulation is blocked by the completion of our MSP command structs and the main flight loop's completion.

- Caleb Zea
 - Unsure if implementation for image transfer works quite yet, waiting for testing capabilities
- Srdan Kopunovic
 - Figure out if there is any available source of data that we have not looked through that we could use for our AI service to detect growth stages and anomalies in crop health.

Individual Contributions:

Team Member	Contribution	Week Hours	Total Hours
Jongmin Lee	Troubleshooting the pipeline crashes on the Nova cluster and rewriting the SLURM logic to handle files that exceeded our storage limits. Improving the data integrity of our training set by patching the auto-labeler's logic bugs Improved the accuracy of our growth-stage classification.	10	42
Z Harvey	Finished field saving logic and API integration with the backend. Fields can now be saved on a per-user basis. Created a new page for the user to see their saved fields, create a new field, and select which field they want the drone to fly. Expanded the existing grid logic to find intersection points within the grid and save those latitude and longitude values for the drone to take pictures.	8	33
Andres Lopez	There is an image, of myself, from our camera, on the SD card. 🕶️	10	34
Alex Pallan	Identified pipeline framework for creating a container in ECR to be deployed with ECS for a backend that can interact with a database hosted in AWS. Additionally started the process of registering the drone with the FAA under my Part 107 license.	8	28
Camden Senneff	Installed and focused new lens Implemented mock controllers for UART over TCP in drone software and verified connection to INAV simulator Created an implementation of libJPEG-turbo in our drone software for Andres to use when writing images Debugged camera issues and fixed several issues with the Raspberry Pi image build	19	56
Caleb Zea	Finishing basic flight API, cleaning up code for merge requests, fixing user session persistence, initial drone -> frontend -> backend image transfer	6	25
Srdan Kopunovic	Finalized API for image transfer, allowing client side interaction for sending requests for both individual and arrays of images, having a GET, POST, and DELETE operations to manage data received from the drone after flight. Additionally, mreplaced backend program running on server to be a docket process, as opposed to the temporary system process configuration in order to allow proper continuous deployment setup and versioning.	5	28