Allsky Camera Network for Detecting Bolides
Milestone 2

Members
Tyler Turner, tturner2021@my.fit.edu
Vincent Quintero, vquintero2021@my.fit.edu
Jean-Pierre Derbes, jderbes2021@my.fit.edu
Charles Derbes, cderbes2021@my.fit.edu
Faculty Advisor/Client
Csaba Palotai, APSS, cpalotai@fit.edu

Progress Matrix for Milestone 2:
If completion is not 100%, relative workload was preserved. For example, if the task was 50% complete by the end of the milestone each person's contribution to that 50% is 50% of what their initial contribution would have been.

Task	Completion	Tyler	Vincent	Jean-Pierre	Charles
Continuously fix endless stream of issues	90%	20%	20%	40%	20%
Add logs for easier diagnosis of issues	100%	0%	90%	0%	10%
Replace current C++ camera code	70%	30%	50%	20%	0%
Implement Server API	90%	70%	5%	25%	0%
Implement Client API	80%	10%	0%	50%	40%
Begin writing CLI	0%	0%	0%	0%	0%
IoT style setup	90%	20%	0%	30%	50%

Discussion of each accomplished task for the current Milestone:

- Task 1: Addressed most of the issues that arose throughout the second milestone. Examples include an issue with the notification system that was fixed (poor relative pathing) and a time zone issue where local time was in UTC. We will continue to address these issues as they appear but have omitted this "task" from the third milestone simply because fixing bugs is part of "general" software development.
- Task 2: Added logging and made debugging much easier. Previously the logging system was very primitive and consisted solely of unorganized prints to the standard output. Now the logging system is much cleaner and saves logs in chunks to separate files. This will help us fix bugs in the current system and help future engineers fix any issues they encounter.
- Task 3: Switched to augmentation and background subtraction instead of pure augmentation for event detection in videos. Detected events are now more consistently recorded (previously they would be cut off early and the same event could be treated as a new event). This issue was caused because the previous implementation didn't have enough tolerance to allow for small "low event activation" periods which would cause it to cut then restart.
- Task 4: Wrote and tested routes for server api, handlers doing CRUD on database. Nodes should now be able to upload video files to the server and the frontend is set to be able to retrieve videos from the server backend which are stored in the database. A queue was also built to handle videos being sent to the server as multiple nodes will send videos to the server simultaneously. Overall the server api is in good shape to handle video processing but still needs work when it comes to frontend interaction.
- Task 5: Wrote and tested routes for client api. Routes that have been implemented include updating and retrieving the state config for a node. When updating the state of a node you do not have to send a full config, just the desired changes. This implementation references the newer version of node state which uses a state.json file.
- Task 6: We are not at a point where we can write the analysis CLI because we have not finished gathering the data. Once the server video processing system is completely stable we can begin implementing the tooling and make it available to the researchers. The data collection will still be delayed until the client-side code is implemented, however that should come soon with the completion of the IoT setup, the client API, and the client video recording and sending.
- Task 7: The IoT style setup is functional but it does not handle all edge cases just yet. The node broadcasts its own network that users can connect to. Once the users are on the network they can visit a webpage where they will be prompted to give their home network credentials to the node such that it can connect to their internet and begin sending data back to the server. Once the user enters valid network credentials the node will lose connection to its local network and connect to the user's network. The user will then receive a notification with information on how to access the node's services.

Discussion of contribution of each team member to the current Milestone:

- Tyler Turner:
    - Worked on server api
    - Implemented queue for video processing
    - Implemented the notification service
- Vincent Quintero:
    - Rewrote event detection software running on nodes
    - Improved image processing to more accurately detect potential events
    - Add event logging to event recording and processing
- Jean-Pierre Derbes:
    - Fixed None/null errors in state
    - Implemented part of client and server api
    - Worked on database model and implementation
- Charles Derbes:
    - Implemented IoT style setup
    - Implemented part of client api
    - Implemented part of logging

Task Matrix for Milestone 3:

Task	Tyler	Vincent	Jean-Pierre	Charles
Replace current C++ camera code	10%	35%	45%	10%
Implement Server API	50%	0%	50%	0%
Implement Client API	20%	0%	20%	60%
Begin writing CLI	30%	10%	50%	10%
IoT style setup	20%	0%	10%	70%
Classification	0%	33%	33%	34%
Start writing UI	20%	70%	0%	10%
Create setup process for node	75%	0%	25%	0%

Discussion (at least a few sentences, ie a paragraph) of each planned task for the next Milestone:

- Task 1: Ported event triggering and event recording from C++ to python. Improved detection algorithm and image processing. Camera recording code still needs to be fully implemented client-side, and the event recording still needs a bit more testing. Client side camera recording will be done with overlapping chunks that way if an event occurs on the edge of a chunk it will either be fully contained in the next chunk or be at the very end of the previous chunk. This does imply that we will have to empirically determine a maximum event length and set the overlap equal to that length. Also the bigger the chunks are the less chance the overlap is "needed".
- Task 2: The server's api is almost completed, it just needs some finishing touches. It still needs to be tested and certain endpoints are still missing pieces of their functionality. Some of the frontend-backend endpoints have still not been fully implemented, however we are focusing on client-server communication before moving fully towards implementing the UI. We have not yet stress tested the server with the amount of video input it would normally be required to handle and we plan to do that.
- Task 3: This is also very close to being finished, just need to fix some of the handlers. Once the handlers are fixed we will move on to testing the client api and integrating it with the server (checking status of all nodes etc.). We also will probably need to reevaluate how state works client-side since all of the client camera recording code will be overhauled and the state.json will no longer be relevant to the new implementation.
- Task 4: In this milestone, we should have enough data to start on the CLI so we are hopeful to start writing it. The CLI will essentially give researchers access to all of the internal functionality of the system. It will allow them to manually perform certain tasks through a suite of powerful tools, like generating composites or running images through the classifier and pulling the classification report.
- Task 5: Building an IoT style setup in python with FastAPI, HTML, CSS, and Javascript (Uvicorn for web server). This task is almost complete as there are only a couple of edge cases and hurdles to overcome. One such hurdle is figuring out a potential alternative to not requiring sudo permissions to connect to a wifi network from a node (which was done to connect the node to a wifi network programmatically). We also need to communicate with the server api to pull info on the user to contact them, which still needs to be implemented.
- Task 6: Working on a binary classifier to filter out uninteresting events such as airplanes, insects, etc. Stage one will be collecting all of the event videos recorded by the previous system and potentially combining them with external datasets. The dataset of videos will then be morphed into a dataset of composites which will then be reshaped such that every composite is the same resolution. We will then label the dataset and increase the size by doing horizontal and vertical image flips. Lastly we will work on tuning the actual model (CNN) to try and get an acceptable accuracy.
- Task 7: Once all of the previous tasks have been implemented and tested we can move on to the user interface. The user interface should meet all of the requirements specified in milestone 1. We have a couple of mockups for how the UI should look that we presented in past milestones and we plan to modify those based on the feedback we received in our presentations. Lastly it is important to reiterate that starting the UI with dysfunctional backend services is a bad idea and we want everything else to work efficiently before we build the UI.

- Task 8: This task will involve creating a workflow to set up and test a node after it has been built. This will include installing the OS, and required packages, and testing it to make sure it is ready to be shipped out. We will also be using Ansible to manage updates to the nodes after they have been shipped out. The project has been plagued with node specific issues where it is hard to determine if they come from software or hardware. This new workflow and the general push to move most of the code server-side should significantly alleviate this issue.

Dates of meetings with Client during the current milestone:

see Faculty Advisor Meeting Dates below

Client feedback on the current milestone:

see Faculty Advisor Feedback below

Dates of meetings with Faculty Advisor during the current milestone:

- Oct 2, 2024
- Oct 9, 2024
- Oct 16, 2024
- Oct 23, 2024

Faculty Advisor feedback on each task for the current Milestone:

- Task 1: Dr. Palotai has noticed fewer issues which means our solutions are working
- Task 2: Dr. Palotai has liked having the logs to help diagnose issues
- Task 3: Dr. Palotai is looking forward to replacing the C++ with an easier to maintain solution
- Task 4: Dr. Palotai is pleased with the progress being made on the server api
- Task 5: Dr. Palotai is also please with the progress being made on the client api
- Task 6: Dr. Palotai thinks that the CLI can wait until the event gathering to finished
- Task 7: Dr. Palotai really likes to idea of the user not needing to open a node when they receive it, so he is eager to get the IoT style setup finished

Faculty Advisor Signature: ________________________________ Date: ________