***EE/CprE/SE 491 SEMESTER 2 WEEKLY REPORT 2***

***1/30/2025 – 2/13/2025***

***Group number: 11***

***Project title: Slowpitch Softball Pitch Detector***

***Client &/Advisor: Nick Fila***

***Team Members/Role:***

***Andrew Vick - Machine Learning Integration***

***Casey Gehling - Client Interaction***

***Sullivan Fair - Individual Component Development***

***Ethan Gruening - Team Organization***

***Josh Hyde - Research***

***Cameron Mesman - Testing***

o **Weekly Summary**

* Over the last week, the team fully integrated the OpenCV code into iOS. We do not have an implementation of the tracking modules for Android, but we are currently working on getting a framework built for it. The team also researched multi-threading so the YOLO model can run separately from the KCF tracking. This should make our app more efficient and improve the runtime. Lastly, we developed a testing plan to collect the relevant data needed to improve our project. From here, we plan on continuing to refine our tracking code, improve the user experience of our app, and perform field tests.

o **Past week accomplishments**

* **Andrew Vick:**
  + This week, I worked on improving what data we can pull from the app while running the tracking code so that it’s easier to test for accuracy.
  + Got iOS app to return ball’s position (X, Y, Radius)
  + Working on getting a measurement for how long the tracking code takes to find, track, and update ball position, including measuring how many frames per second we are getting.
* **Sullivan Fair:**
  + With Andrew's help, I got the OpevCV code to run through Flutter onto my phone. This is a great step for the team as we can now move forward with testing a full integration of our app. The code seems to log data points quickly, but the video feed is quite laggy, which we are looking to fix. The YOLO model is the likely cause of the video lag since it is memory intensive.
  + To address the laggy video feed, I altered the code in an attempt to multi-thread it. The idea is if the YOLO model can run on a separate thread, that will reduce the lag in the video feed since there will be less processing done while doing general tracking with the KCF module. However, when I ran the altered code, the app crashed due to too much memory usage. I am new to multithreading, so I am currently working on familiarizing myself with the process and optimizing the current code to be more effective.
  + Andrew and I also took the lead in testing the tracking code in the app. We plan on conducting our first full test this week.
  + TLDR
    - Got the tracking code to run on my phone
    - Started to alter the code for multi-threading to separate the YOLO model from the general tracking
    - Formed a testing team with Andrew
* **Casey Gehling:**
  + Documented various development environment issues that have been encountered with potential fixes.
    - Ruby/Podfile errors encountered when navigating between branches.
    - Cache issues that persisted through branch changes.
  + Researched potential multithreading opportunities for native development as well as OpenCV optimization for IOS and Android. Did research for the IOS Accelerate framework which could potentially be used to speed up computation heavy functionality. Currently working towards integrating possible solutions
    - [Mobile OpenCV Performance Paper](https://www.researchgate.net/profile/Klaus-Schoeffmann/publication/261959725_OpenCV_Performance_Measurements_on_Mobile_Devices/links/02e7e53b8fbf00b214000000/OpenCV-Performance-Measurements-on-Mobile-Devices.pdf)
    - [IOS specific OpenCV Development](https://books.google.com/books?hl=en&lr=&id=cP1vDQAAQBAJ&oi=fnd&pg=PP1&dq=opencv+on+iphone&ots=3IXqn6MDtD&sig=FEeJekhUNGet3s9jkN3cO4TClgA#v=onepage&q=opencv%20on%20iphone&f=false)
* **Ethan Gruening**
  + Looking into implementing the working tracking code from the iOS testing to Android, the OpenCV Android SDK supports only a select few modules of the latest release. Although Android is able to use OpenCV to manipulate and analyze frames, it cannot use the tracking functions. There exists a opencv\_contrib modules where the tracking libraries exists. I started researching ways to edit the SDK or new imports to allow Android to use OpenCV tracking modules.
  + To prepare our application for testing, I implemented new analytical features on the camera screen to overlay the calculated height of a ‘found’ ball and the overlaid bounding box (circle) on the ‘found’ ball. I also started a function to later integrate with the C++ code to send the frame and record the frame rate.
  + I also made simple modifications to change the the default values to 12 (max height), 6 (min height), 6 (reference height) on the height setup.
  + In preparation for testing, I also wrote the full template for a single accuracy test, the test instructions, and our testing plan. Additionally, I compiled ideas for user experience testing.
* **Josh Hyde**
  + I mainly worked on providing and improving a max and min height visual line within our app camera. The goal was to implement lines representing the max and min heights which would aid in the potential height calculation and visual confirmation with the users to actually see why some pitches were called illegal or not. I first implemented the lines at random points to see if they worked, and then later made sure that these lines implemented the actual max and min heights used through the calibration set-up. I implemented these lines within the normal camera app screen that is used to record the pitches as well as the past pitches screen as it also will show the video of the pitches and it may be easier to identify illegal pitches afterwards when looked into. I still need to improve it as it provides a random third line I’m not sure why is used.
  + I also testing others and my app code on my android phone and confirmed that it works pretty smoothly on my phone, however I haven’t been able to run it yet with the opencv ball tracking code implementation of our app. However, the orientation of the phone using the app is a little off as it seems to be rotated 90 degrees and will most likely need to be changed or fixed.
  + Lastly, I added the illegal height calculation within the flutter app code, but it still needs to be tested as it relies on the opencv height detection code, and we just recently got that running on our flutter app.
* **Cameron Mesman**
  + This week, I continued to work on getting multithreading and asynchronous functions. Once we got some code working on that front, I moved on to researching ways to make use of available machine learning hardware on android and iphone. Currently, our code only makes use of the cpu in the phones which is inefficient for machine learning. If we can make use of the gpu on the phones, our code will run more efficiently.
* **Pending issues** 
  + Currently, there are a couple of issues with our app. The main issue is that running yolo on the phone without utilizing Apple’s ML framework causes significant frame hangs and stutters. We are also having difficulty translating the frame coordinates for the raw image data to the coordinate system flutter is using.

o **Individual contributions**

| **NAME** | **Individual Contributions**  *(Quick list of contributions. This should be short.)* | **Hours this**  **week** | **HOURS**  **cumulative** |
| --- | --- | --- | --- |
| Andrew Vick | Completed Flutter/ OpenCV integration for iOS | 6 | 67 |
| Casey Gehling | Documentation, Research | 5 | 70 |
| Sullivan Fair | Completed Flutter/OpenCV integration, Started multithreading, Formed testing team | 8 | 72 |
| Josh Hyde | Max and Min height visual lines, Android app running on phone | 8 | 72 |
| Ethan Gruening | New Flutter Screens, OpenCV in Android | 7 | 94 |
| Cameron Mesman | Researching C++ implementation on iOS | 6 | 67 |

**Plans for the upcoming week**

* Andrew Vick
  + Optimize processing pipeline
  + Improve metrics code returns for debugging/testing
  + Test
* Casey Gehling
  + Optimizations to current processing pipeline
  + Testing
* Ethan Gruening
  + Work on adding the OpenCV tracking modules to Android.
* Josh hyde
  + Work on fixing the bugs associated with the max and min visual lines
  + Try to fix the orientation issues on android phone
  + Fully implement and have opencv work with the app on my end as well as implement my code with it
* Sullivan Fair
  + Continue to optimize the multi-threaded code
  + Perform the first tests of our full integration
* Cameron Mesman
  + Find a way to access GPUs on iphone and android.
  + Work on code that can access GPU on iphone and android in one ML model