EE/CprE/SE 491 WEEKLY REPORT 06

10/19/24 – 10/25/24

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

- **Weekly Summary**
  - This week, after testing our object/height detection scripts with on-field data, we researched new methods of calculating height and identifying the data points, calibration, and conditions we need to make accurate calculations in various environments reliably. In preparation for integrating our scripts with the app, we are beginning to translate our python scripts into C++.
  - Additionally, after prototyping our screen sketches and integrating the camera plugin within the Flutter app, we are moving into the development of the screen functionality. In preparation for testing object/height detection scripts within the app, we are conducting more research and testing on asynchronous and background C++ execution.

- **Past week accomplishments**

  - **Andrew Vick:**
    - For this week I started switching what we had for object detection and tracking over to C++. So far I have been able to get the basics of OpenCV working and now need to implement our specific detection logic. Since we've been having issues with detecting the ball using color, I have reworked our color bounds to be tighter to the real color of the ball. To better our non-machine learning detection, I am also working on integrating movement-based detection so that the ball is only tracked when moving and the camera has the ball's color in the frame. Finally, we also tested my height calculations, and they are wrong if we want to use a ML model to predict the distance of the ball, we will need to rework how we use the ball's position in frame and distance to the camera to find its height.
    - TLDR
      - Working on C++ code, we can use in the Flutter app to interface with the camera.
      - Implementing detection and tracking for the ball in C++ code.
  - **Sullivan Fair:**
    - This week, I finished my testing of the MOSSE detection script. I found that there were many inconsistencies in the detection especially in different lighting. I recreated the script without MOSSE, used the HSV color space, and applied a blur to the frame. With this, I was able to get much more accurate detection, but there were still issues in different lighting. We will need to take those factors into account when further developing Andrew's detection script.
    - I also began researching ways to integrate C++ into a Flutter application, which was one of the next milestones we decided would be good for the team.
    - TLDR
      - Finished testing the MOSSE algorithm
      - Recreated detection script without MOSSE with better results
      - Researched ways to integrate C++ into flutter

- **Casey Gehling:**
    - This week, I finished proofing our Flutter app camera POC, demonstrating how our camera functionality will work within our final app. I also spent some time creating a system so that this process can be easily replicated in the future.
    - I also spent time researching how to properly integrate our eventual C++ script into Flutter. I found that Dart, the language used primarily by Flutter to build apps, contains an API called the *ffi* api that allows Dart scripts to easily access and call certain functionality from C++ libraries. Work on this integration can be done once our C++ script has been finished.
    - I also spent some time researching some potential solutions for a multi-device approach to our application in the scenario where we want to integrate multiple cameras. I found that Flutter has a simple Bluetooth API (BLE) that will aid us in seamlessly connecting our application to other potential devices for purposes such as transmitting video feed via UDP.
- **Ethan Gruening**
    - This week, I worked to write the slides for Lightning Talk #4, defining our project management techniques through AGILE and how it defines our task decomposition, key milestones, and key setbacks.
    - I additionally created the calibrate-distortion branch in GitHub. The calibrate_distortion.py code has two methods that can be called.
        - The first is calibrate_camera_live, which takes in and displays a live camera feed. Every three seconds (by default), the camera can detect a checkerboard pattern (9x7 by default), record the frame, and display it to the right. After 15 frames (by default) are collected, the camera is shut down, and the images are displayed on the screen. The distortion data is recorded into a JSON file using OpenCV's calibrate_camera method.
        - The second method in calibrate_distortion.py is the undistort_image method that takes in an image and a path to a distortion data file (written by calibrate_camera_live), and it uses OpenCV's method to undistort the image and return it.
        - The methods within calibrate_distortion.py have proper documentation and clear comments outlining the code's functionality for further collaboration.
    - Exiting the brainstorming and research stage in many of our

developmental tasks, I merged the current camera-integrated Flutter application, the current ball detection script, the color calibration script, and the distortion detection and removal script to reflect our progress within our main branch.
    - Finally, I continued research on other height calculation methods that define our environmental constants and setup requirements that ensure accurate height calculations. I began collaborating and discussing our problem with mathematics students to brainstorm potential solutions.
- **Josh Hyde**
    - This week I worked on finally finishing my work into the negatives and benefits of using machine learning compared to non-machine learning. I have come to the conclusion that machine-based-AI learning for the most part will be a more reliable and more accurate way to go for this project. However, there are some non-machine based learning techniques that can be incorporated that could be useful such as just the object detection part of the project. However, with the amount of changing and differing variables within every field and camera, machine based learning models will more work much better and more accurately for a wider range of differing data and inputs.
    - Additionally, I put work into trying to understand and get more comfortable with using flutter, to get a better understanding of it and to start looking into different ways to use flutter on our app not only for the IOS version, but also for the android version because it would be beneficial for our flutter app to incorporate both IOS and android users in the future.
    - Lastly, I put a little bit of time into trying to revise a version of a non-machine based code that can accurately detect the ball and incorporate it into a height detection model as well. Because the non-AI based version works better in only very certain conditions, it worked pretty well up close in my room but not as much using the softball video data we took on the softball field.
- Cameron Mesman
    - This week, I caught up on all things flutter. Having previously just worked on the screen sketches for the app, I felt a bit behind on how Flutter worked and how we're going to use it. I read through a bunch of the docs and went through some of the tutorials to get a good overview of how FLutter works and how we could implement our app.
    - I looked into everyone else's screen sketches from last week and started compiling what I thought were the best features of each of them into improved sketches.

- **Pending issues**

  We will need to work on converting our detection an tracking logic over to C++, and find a framework to develop an Android and IOS app that can run C++ code on the phone. We are also still running into issue with getting accurate distance data and then using that to find the height of the ball.

- **Individual contributions**

NAME	Individual Contributions *(Quick list of contributions. This should be short.)*	Hours this week	HOURS cumulative
Andrew Vick	OpenCV testing, Object detection, photogrammetry research	4	34
Casey Gehling	Flutter camera POC, C++ integration research, BLE research + flutter (for multi-device approach)	5	34
Sullivan Fair	Finished testing MOSSE script, created non-MOSSE script, Flutter/C++ research	5	33
Josh Hyde	ScreenSketches for app, Research into non-AI based object detection	4	33
Ethan Gruening	Height detection research, OpenCV lens distortion calibration, GitHub management.	7	37
Cameron Mesman	Flutter overview, screen sketches improvement	5	20

- **Plans for the upcoming week**
  - Andrew Vick
    - Continue on C++ code and Flutter integration
    - Improve non-ML object tracking
    - Refine height calculation and explore new ways to find the distance to the ball.
    - Centralize our code
  - Casey Gehling

    - Integrate C++ script into flutter app when finished
    - Start work on developing our UI into app from screen sketches
    - Get working recording and storage implementation for past pitches (consider temporary caching videos within the timeframe of a game)
- Ethan Gruening
    - Prototype a new height detection solution
    - Integrate calibration techniques into the ball_track.py method
- Josh hyde
    - Look into more flutter app work and android version of it
    - Look into alternative or more consistent ways to find the height of a ball in a video
- Sullivan Fair
    - Further research integrating C++ into Flutter
    - Create a simple “Hello, World!” program in C++ and attempt run it in our app as a proof on concept
- Cameron Mesman
    - Continue working on screen sketches
    - Work with the rest of the team on testing non-ml solutions and testing our current solutions