To: Dr. Satici, Nardos Ashenafi, Brian Higgins

From: Robotic Vision Team: Daniel Pullicar, Mason Cannon, Roscoe Ambrose, Haston LaGrone

Subject: Biweekly Update, 2/12/23 – 2/26/2023

Date: February 26, 2023

Hello again professors. I hope the new month is treating you all well. The following is a short update on progress our group has made since our last update.

# Work Completed

The final two weeks of February have been a slow roll for the team. There has been a significant amount of debugging dedicated to the separate implementations to try and overcome some unforeseen issues. This aside, we are optimistic we can get all the implementations up to speed in the coming week and move forward with our goals.

Daniel Pullicar:

* Created calibration chessboard specific to our constraints.
* Edited code to allow for placement anywhere on table.

Mason Cannon:

* Helping Dan out with the calibration and getting the GitHub branch setup to pull and push the code.

Roscoe Ambrose:

* Investigated async.io for running several instances of our program at the same time.
  + Not going to be used. Async.io is specialized for programs built around waiting for something.
* Begun working with Daniel to see how calibration will be merged with the main program.

Haston LaGrone:

* Fixed issue with camera not accepting parameter configurations without Logitech Capture being utilized. (Logitech software)
* Confirmed that the central mounted camera that the team already owned is capable of 60 fps and uses the same API back end as the Logitech BRIO, meaning no new cameras need to be purchased.
* Identified candidate Linux API backends to interface with camera.
* Manual adjustments available for focus, saturation, brightness, and exposure time.
* Discovered that exposure parameter will bottle neck FPS when being manually set.
* Identified number code for present cameras to assign parameters. Numerical values on Logitech capture did not correlate to values within OpenCV.setparams

Group:

* Identified issue in which running multiple cameras off one program is not feasible. We will need to initialize parallel programs, for each camera, on separate threads and feed data to a single location to accomplish desired functionality.
  + Output physical x-y data from each camera into Kalman filter to be averaged.

# Areas That Need More Attention

* Class implementation of detection program. The initial implementation does not accomplish the needed goals. Returned and assigned parameters must be modified.
* Use too many global variables. Will be fixed in class implementation of our program.

# Work Pending

The team’s focus remains on a first pass at two camera operation. After cleaning up some bugs and correcting the detection program class a first experiment will be performed by running two detection programs in parallel.

Daniel Pullicar:

* Test point-conversion for accuracy
* Generalize code and separate into smaller classes.
* Find fix for broken undistort function.

Mason Cannon:

* Going over the calibration and learning how it works. I am also following Daniel’s code and making sure it is correct and will work for what we want to do.
* Assist with testing point-conversion for accuracy.
* Assist with the coding the classes and other functions in the calibration.

Roscoe Ambrose:

* Assist Haston in implementing a class version of our program.
  + Give camera attributes.
    - Self.cameraVariables
    - Self.algorithmVariables
  + Reduce total number of global variables.
    - Getters and setters to access attributes of camera.
  + Allow us to call this program via the command line.
    - Will allow us to call multiple instances of the program for utilizing more than one camera while passing port # as a parameter.

Haston LaGrone:

* Finish class implementation of both detection programs (test program and skeleton version) with desired parameters.
* Tandem capture test. Initialize two instances of detection program and observe behavior.
* Further refine initialization of camera parameters. Current image is usable, but still dim in comparison to the auto-calibrated feed provided by Logitech capture.
* Investigate ROS node implementation (This will be done with Nardos’ assistance).

# Conclusion:

The prior two weeks were not as successful in some areas as the team would like. With that being said, the hiccups we ran into will allow us to further polish the final product. Looking at this with a silver lining, we did discover the ways in which we will not be implementing multiple camera detection. The method of utilizing multiple applications in parallel will not be as elegant of a solution, but much more robust.

# Prototype Abstract:

The objective of the Robotic Vision Team is to design a visual detection system for use with an air hockey playing robot. The program will use one or more webcams to observe a desired play space. The program uses a 3-step process to operate. It will facilitate an initial calibration designed to relate pixel coordinates of the cameras to spatial coordinates on the play space. The program then passes control to the operator. The operator will use a graphical user interface to assign parameters to a circle detection algorithm which will return the pixel coordinates of any objects matching said parameters. When calibrated to the desired puck dimensions, the program will return the physical coordinates of the puck’s center as it moves through the play space in real time.