Journal Report 5
10/7/19-10/14/19
Austin Huang
Computer Systems Research Lab
Period 1, White

Daily Log

Monday September 30

Tried to apply binary black and white image to tracker. Does not accept frame because 'depth' error. Looking for a way to modify depth. Turns out it is because frame is somehow now CV64F. Need to find a way to change that. Personal laptop bluescreened towards end of class. Booted up and worked fine at home.

Tuesday October 1

Tried to convert binary image to grayscale to change CV64F to 32F, but the cv2.convert method does not accept CV64F either. How does changing the bgr values of a Mat array change the depth of the array? Does not make sense.

Thursday October 3

Continue struggling with depth parameter. Figured it has something to do with numpyarray and conversion to array with opencv, an inherent python problem. I hate computer. Bluescreened again, don't know what causes it, wasn't running any code and didn't physically move computer. Researched more about dtype and opencv channel type with school laptop.

Timeline

Date	Goal	Met
Sept 23	Apply tracker to work in other passes	Need way to apply tracker to fast moving objects.
Sept 30	Create a background subtractor to find balls faster	Achieved but it is slow.
Oct 7	Apply background subtractor to find balls faster	to be determined
Oct 14	Locate and mark a lacrosse ball when it is visible in frame	
Oct 21	Identify and track all players on the field	

Reflection

My main problem this week was trying to apply the tracker to the binary background subtracted frame I had generated. This was due to the depth parameter of my frame not matching what the algorithm in OpenCV wanted. The error was:

Unsupported depth of input image: 'VDepth::contains(depth)' where 'depth' is 6 (CV64F)

This led me to believe that the frame I had generated was a CV64F. Upon further research, I learned that this means the array I had was a 64-bit matrix of 1 channel of floats. This didn't make sense because I had designed my matrix to only contain either 255 or 0. When I debugged, I discovered the array elements all has decimal points after the number. So I deduced that the array when instantiated probably defaulted to float.

I first tried to convert back to grayscale, but the convert method in opencv also did not accept CV64F matrices.

I tried to mess with np.array() and the dtype of the numpy array to control the type of data I put in: int32, float26, and uint32, all did not work.

Converting from a nparray to an opencv array was much harder than I had thought. Last year in Computer Vision, I worked in c++ and I did not have this trouble because everything was stored in a Mat object, which was a opencv object in c++. However, in python, when I first started, I researched that the equivalent of Mat objects were numpy arrays and so I had thought the two were synonymous. However, even if they are, handling the two and making sure they are consistent with each other has proven to be more difficult than I could have imagined.

This week I plan to do more careful debugging and research. Hopefully I am not stuck, for I still do not have a great understanding of the relationship between dtypes and opencv matrices channel types. My plan moving forward if I remain stuck is to make the image non-binary, but still filtered to hopefully satisfy the method parameter.