Journal Report 6

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

Daily Log

Monday October 7

Figured out number format that satisfies both numpy and OpenCV Mat arrays. It is called "uint8," which is short for unsigned 8-bit integer.

Tuesday October 8

Now with background subtractor working, tested on original short pass identified in NCAA game video. It worked without subtractor, so logically it should work with subtractor. Luckily, it does. Went back to original video and tested original pass. Does not work. Returned to NCAA game video to look for passes. Identified two potential static camera passes.

Thursday October 10

Tested the two passes.

First pass works until it gets obscured.

Second pass works with subtractor. Tests to see if it works without subtractor: Does not.

Timeline

Date	Goal	Met
Sept 30	Create a background subtractor to find balls faster	Achieved but it is slow.
Oct 7	Apply background subtractor to find balls more accurately	Reached only if pass is "clean"
Oct 14	Make tracker more accurate, be able to track when pass is not clean	to be determined
Oct 21	Identify and track all players on the field	
Oct 28	Identify and track all players on the field	

Reflection

I figured out that the format number that prevented me from converting a numpy array to an OpenCV Mat object last week must have been a type of integer. BGR values are always reported in integer, not float or any other type. I tried int64, int32, and int16 again. But then I realized I was forgetting int8. It makes sense that uint8 ended working because in my binary black and white image, the only values I included were 0 and 255. The max number in an unsigned 8 bit number system is 2**8-1: 255.

With the background subtractor working, I tried to test the original pass, the one case that worked without the subtractor, using it as a control to compare it off. It worked luckily, which showed that the subtractor was not harming the existing tracker processes.

However, one thing I noticed was that the background subtractor was extremely slow. Each pass was about 30 frames or 1 second in a 30 fps video. However, the background subtractor took about 20-30 seconds for each frame. To track a one second pass, I usually was spending about 10-15 minutes. One thing is definite: I need to optimize the background subtractor while making it more accurate.

I found out that the background subtractor only assists in clean passes. I define clean passes in this context as a pass where the ball, during the pass, is always in front of the green grass background, never overlapping with any other color from the perspective of the camera. In the passes that did not work, the ball was passed over the red tracks in one of the passes, it was passed through a player with a white jersey in another, and the pass made the ball travel through a white yard line on the football field in the other.

While the pass only worked for a clean pass, it worked well. After finding a clean pass and testing to see if the background subtractor worked, which it did, in tracking the ball throughout the pass, I tested tracking the pass without the subtractor. The tracker got lost midway through the pass because the ball was accelerating as it descended into the player's stick. This shows that the addition of the background subtrator was not a waste.

Going forward, I see optimizing the tracker to be quicker and better at tracking the ball even with an interference of color background as my next goal.