Journal Report 4

9/30/19-10/7/19
Austin Huang
Computer Systems Research Lab
Period 1, White

Daily Log

Monday September 23

I researched how to apply a background subtractor to a dynamic background. Ran across idea to use color similarity. Instead of using patterns or trying to detect closer vs. farther entities, color seems to be the way to go in this problem.

Tuesday September 24

Talked to Mr. White. Confirmed my idea to use background subtractor for similar colors (grass color). Researched how to quantitatively calculate color similarity/difference. Started to implement. Found out that OpenCV Mat properties are BGR, not RGB.

Thursday September 26

Implemented algorithm successfully. Toggled threshold options. Settled on 90 after color distance is subtracted from the average. Time is a major problem; I need to work on optimization if this is a valid strategy moving forward. Still need to test if this makes tracker more accurate.

Timeline

Date	Goal	Met
Sept 16	Locate and mark a lacrosse ball when it is visible in frame	Was able to track a ball in a hand-off pass.
Sept 16	Apply tracker to work in other passes	No, need way to apply tracker to fast moving objects.
Sept 23	Apply background subtractor to find balls faster	to be determined.
Sept 30	Locate and mark a lacrosse ball when it is visible in frame	
Oct 7	Identify and track all players on the field	

Reflection

I opened the debugger to see the field that stored the color value for individual pixels inside the Mat object in OpenCV. Turns out it is one big numpy 3 dimsional matrix (720, 1820, 3), which is the resolution of the frame, each cell having three values for one of each rgb value. I had thought that the values of the pixels were RGB, but then realized during a debugging session that the color of the pixel was darker than the green pixel that was supposed to appear. I looked into the specific frame, and there was more red value than green value. I finally researched and discovered that OpenCV stores values as BGR. This is because back when the library was first created, many camera manufacturers and software developers used BGR. This was the precedent that never changed even when then norm became RGB.

The actual algorithm I implemented for background subtraction involved color subtraction. Because the camera was dynamic, the built in background subtractor in OpenCV would not work, neither would many of the conventional algorithms developed before. I realized that a background subtractor for a dynamic camera must involve detecting elements common to all frames, even though the camera was moving. Color is good because the lacrosse game is played on a green field; most of the background is a green field.

For every 100 pixels, I found the rgb value of the frame, then I averaged the value. Research says that besides conventional distance method (sqrt(r squared + b squared + g squared)), there exists an e-delta method that is more accurate to what humans perceive. However, I figured that at least for the first step, using the conventional distance formula would be sufficient. After I found the average pixel value (which should be close to the color of grass because most of the frame is grass), I proceeded to test every pixel in the frame by finding the distance between that specific pixel and the average pixel. If it was greater than the threshold, it would be considered foreground, and if it was within the threshold, then it was considered background. I turned this into a binary image of black and white. After adjusting the threshold, my algorithm works well. However, it is extremely time consuming. Each frame takes about 10 seconds to work. Going forward, if I decide to use something like this to improve tracking of the ball, I will need to optimize this algorithm.

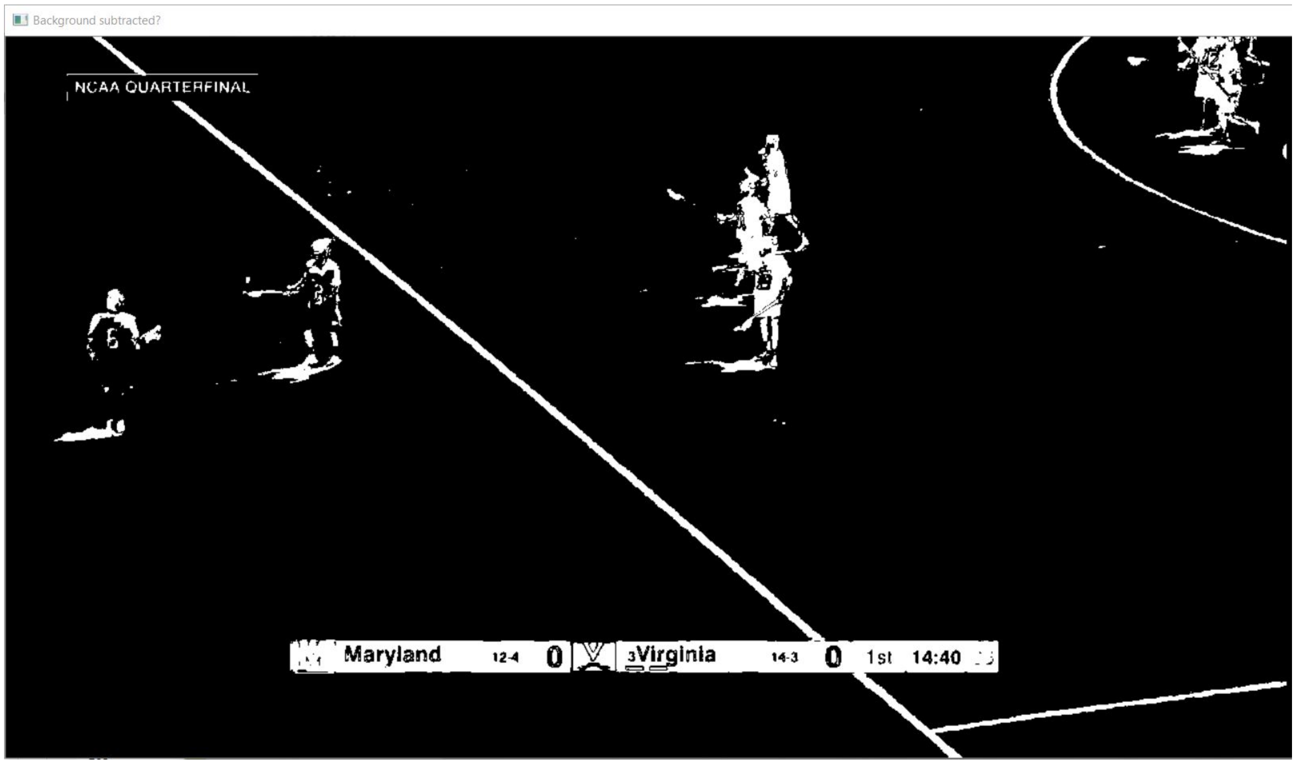

Figure 1: Frame with background subtracted using color average algorithm.

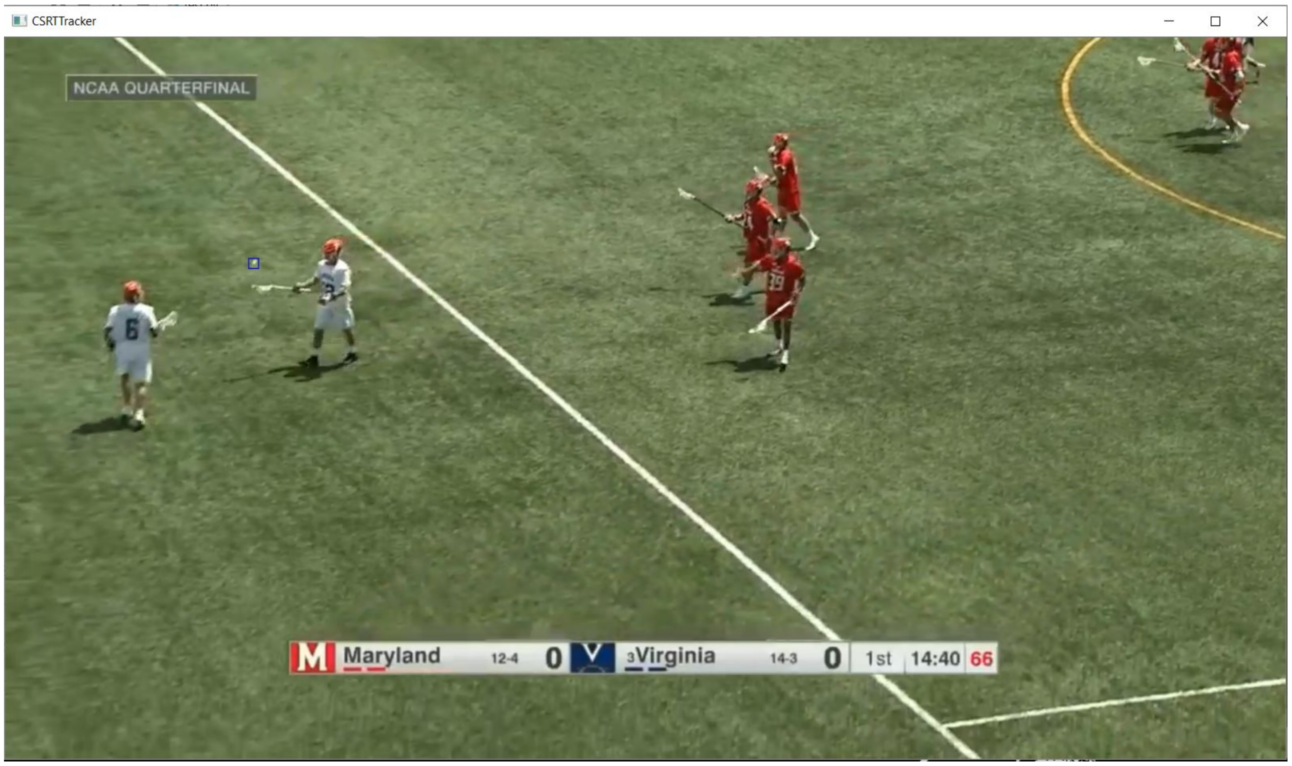

Figure 2: Original frame with the ball enclosed in a bounding box.