Luisa Chiu

10/26/2023

CPE/EE 428

Homework 2: Magic Wand

**Qualitative Evaluation Questions:**

1. The ball detector seems accurate for the most part, but there are some moments where it doesn’t detect the ball or detects more than one when there is only a single ball. The ball detector fails when the ball moves too quickly for it to be determined as a circle, probably due to the fact that the quick ball movement is blurred in a frame. Another instance of the ball detector failing is when the ball gets too close to the camera; they detect it as more than one ball.
2. I believe it is not possible to correctly rotate the 3D box according to the ball’s orientation, because the image processing looks through each frame and detects circles. A 3D ball is circular no matter how we rotate it, so we can’t really determine rotation based off of a single ball moving. It seems that we can only determine translational movement rather than rotational movement. However, I think we could physically modify the magic wand to be a rectangular prism and have at least two small geometric shapes drawn close to both ends. Seeing how far the geometric shapes move can let us know how much the balls are rotating relative to the wand.

**Solution Summary:**

For this homework, I created the functions in magicwand.py and tested each function as I progressed to make sure they worked as expected. I started off with the detect\_ball function, using the cv2 library to apply a grayscale, gaussian blur, and hough circles to detect circles in each image. In calculate\_ball\_position, I used projection equations to calculate the world coordinates of the ball (X, Y, Z). This involved using the actual radius of the ball, the focal length, and the inputs (projected ball coordinates in image: x, y, r) to calculate the actual depth, Z, of the ball, and the actual position coordinates, X and Y, of the ball. Next, I developed the project function to produce the 2D location of ball projection in an image. After, I used the project function to assist my development of the draw\_line function and used cv2.line to display a line on the image. I then used the draw\_line function 12 times to create the draw\_bounding\_cube function that draws a bounding cube around the detected ball(s). Lastly for the magicwand.py file, I created my process\_frame function using the previously developed functions in a for loop, used to iterate through circles detected to draw a circle and bounding cube around the ball(s) detected. In the tracker.py, I used wand.process\_frame to output a list of circles detected. I also created another list to display the trajectory of one detected ball using matplotlib, and another list to obtain data for calculating the distance between balls in the wand.mov file. I then used the distance formula and used np.mean to calculate the average distance values between the two balls. I obtained an average distance of 36.9 cm.