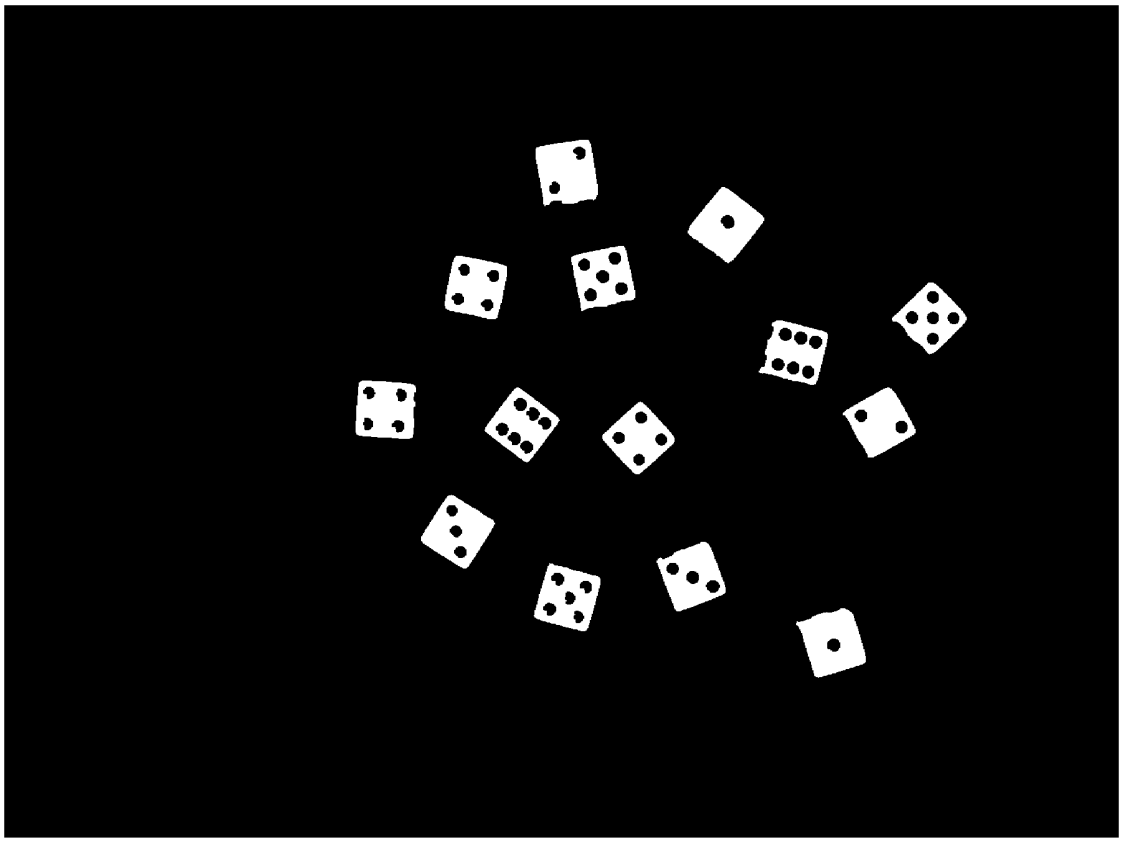
Name: Divyank Kulshrestha

8a. How did you do noise removal?

In order to remove noise, I isolated the green channel, converted it to grayscale image and then binarized the image using the function 'imbinarize', which gave a pitch black background with white dice (although with some imperfections). Then I used the function 'bwareaopen' to remove small objects from the binary image. And finally, using the function 'imclose' to perform closing (dilation followed by erosion) on the binary image cleared up the majority of spots in the dice. Below image is one example:



8b. How did you decide to threshold the image? Did you use graythresh? Did you use imbinarize?

I used the function 'graythresh' on the grayscale version of green channel to get a global threshold value. Then I used that value for the 'imbinarize' function to convert the grayscale image to a binary image.

CONCLUSION

After rotating the image as required (which was easy to do), next step was to separate the dice out of the background. In order to accomplish that, I binarized the image using a threshold value to get a heavily contrasted image with dice as completely white and background completely black.

To remove some connected components that were not required (noise), I used the function ‘bwareaopen’, which takes in a value argument and removes the components which have fewer pixels than that value. Then I performed closing using ‘imclose’, where dilation is followed by erosion, using a ‘disk’ structuring element to remove small holes and gaps. Using ‘bwlabel’ which labelled and returned all the connected components, which were dice in our image. I tried to use the ‘imfill’ function to clean up but it also filled up the black dots on the dice, so I had to remove it.

To isolate and work with each dice, I used the ‘regionprops’ function to obtain properties of each label in the labelled image aka dice. To draw a cyan box around each dice detected, I retrieved the x-coordinate, y-coordinate, height and width for each dice, and used those values as arguments for the ‘rectangle’ function, which draws a rectangle on the image.

To count the dots on each dice (value on the dice), I used the ‘imfindcircle’ function, which finds circles within a range of radii in an image using a circular Hough transform. The ‘objectpolarity’ argument was set to ‘dark’, which made it easy to detect black circles on a white dice. Count of the values was stored in a dictionary, which was finally printed on the console.