**Project 6 Coin Detection – Part 2 complete**

Name: Vivian Feng Period: 5 Date: 2/27/2023

Did you name your file l062.cpp (Lower case L, then 062)? Yes

Did you create the edge matrix? Yes

Did you create the gradient direction matrix? Yes

Did you create the imagev.ppm (visual of votes)? Yes

Did you use Bresenham's line algorithm for voting? Yes

Did you use the intersection with the border of the image or a similar approach when using Bresenham to vote? (fill in your answer)

I found the maximum radius of a coin. When voting with the Bresenham algorithm, I calculate the unit vector in direction of the gradient. Next, I vote along a segment with the length twice maximum radius centered at the edge pixel.

Does your application create imageCircles.ppm file? Yes

Does your application create imageCoins.ppm file? Yes

Does your code processes all command line arguments specified in the course materials? Yes

Does your code display on the screen and in results.txt a summary of your results? Yes

Did you test your code on terminal/gnu linux in c++11? Yes

Run your code (the same code you submit) on the easy image I provided -then paste here the following:

* The initial image ( the one I provided), the imagev.ppm, imageCircles.ppm and imageCoins.ppm you obtained running your code, copy paste here the content of your results.txt file your code created

For the easy image:

Command line including all parameter values used for this image (including the ones you created with explanation of what is their meaning): (not a printscreen!)

Command line arguments – the flags “-lt”, “-ht”, and “-TC” are ignored if they are not specifically overridden in command line.

-f image.ppm -lt 50 -ht 75 -ff imagef.ppm -fg imageg.ppm -TC 25 -fv imagev.ppm -fcc imageCC.ppm -tCircle20 -fCi imageCircles.ppm -fCo imageCoins.ppm -fR results.txt

-minCentDist 80

- minCentDist – distance in pixels between centers for two circles to be considered duplicates

- default settings for easy image:

I divided the image into four quadrants, and each quadrant has its own threshold for canny edge and vote counts. The sections of the image are divided as such: Diagram, engineering drawing

Description automatically generated

Threshold for Quadrant I:

* + Low: 50, High: 75 for Canny Edge. Votes threshold: 30% of the global maximum vote count

Threshold for Quadrant II:

* + Low: 50, High: 75 for Canny Edge. Votes threshold: 25% of the global maximum vote count

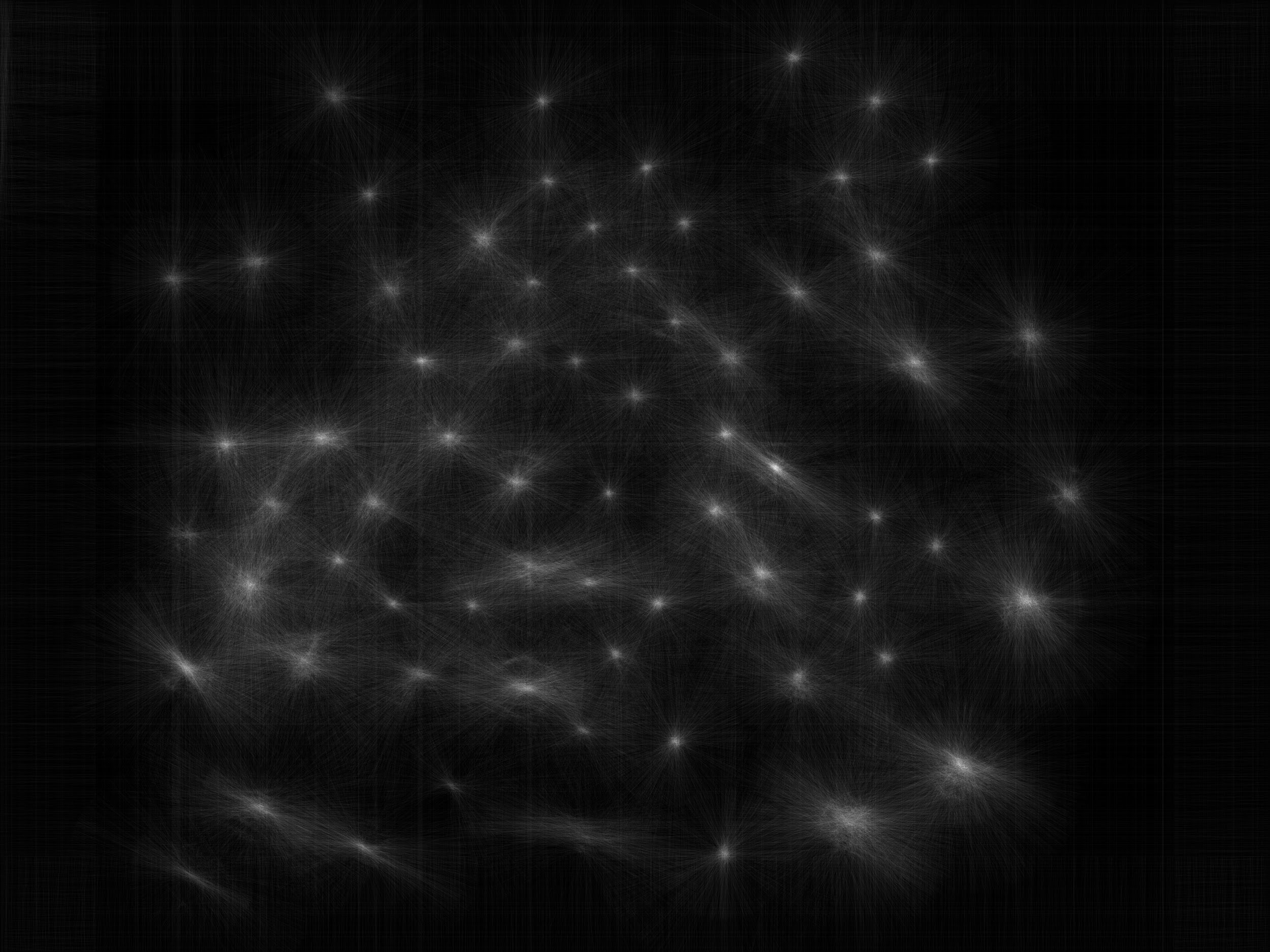
Threshold for Quadrant III:

* + Low: 75, High: 150 for Canny Edge. Votes threshold: 35% of the global maximum vote count
* Threshold for Quadrant IV:
  + Low: 50, High: 150 for Canny Edge. Votes threshold: 35% of the global maximum vote count

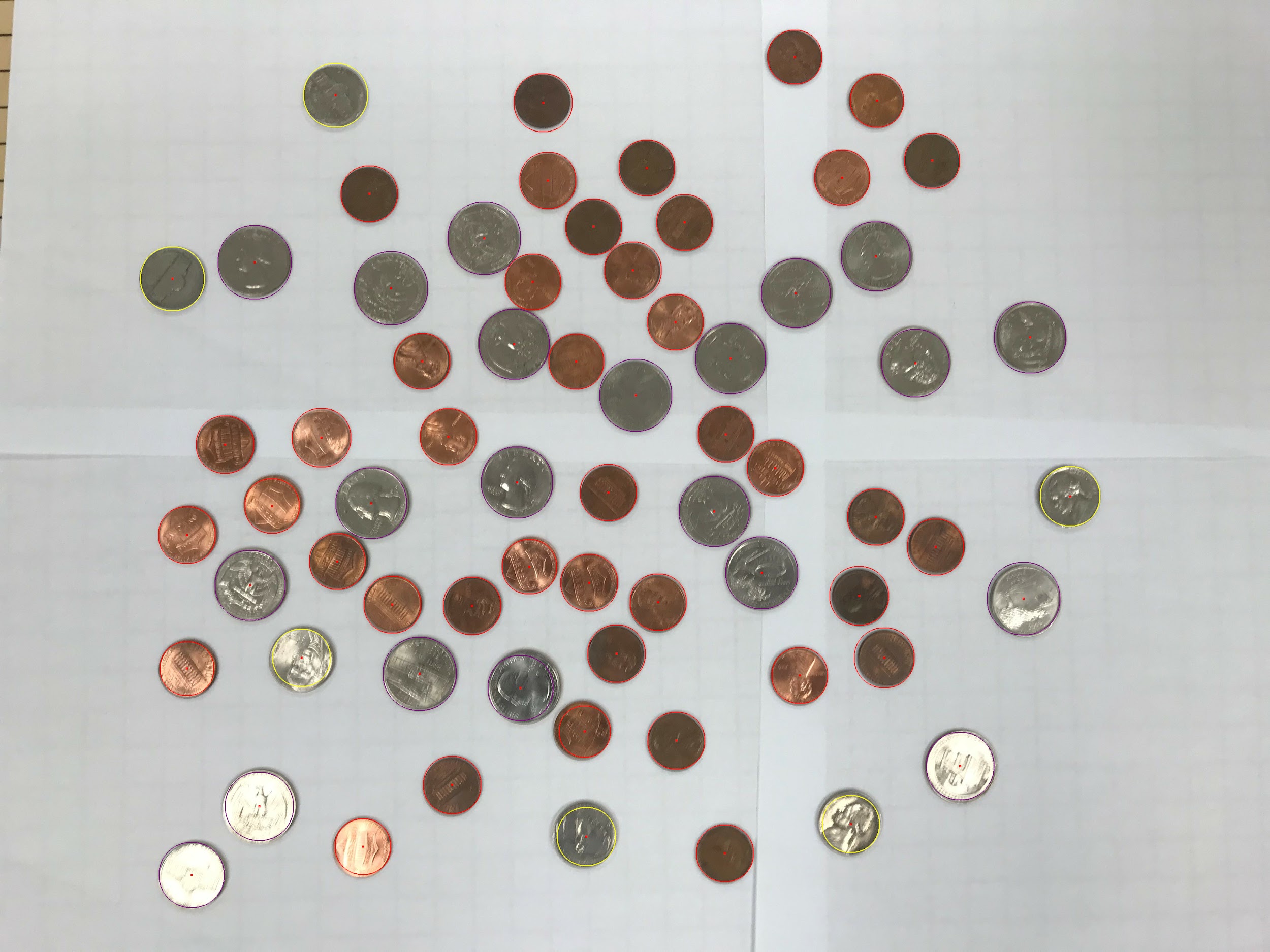
Initial image:



imagev.ppm:



imageCoins.ppm:



imageCircles.ppm:



Content of results.txt: (NOT a printscreen so I can copy/paste if I need)

penny - 41

nickel - 6

dime - 0

quarter - 21

half dollar - 0

dollar - 0

Total: $5.96

Did you attempt to make your code work against the other 2 images? No

If yes paste here the same information as you did above for the easy image: