CS543 Assignment 2

Your Name: Siddharth Garg
Your NetId: ssgarg2

Part 1 Hybrid Images:

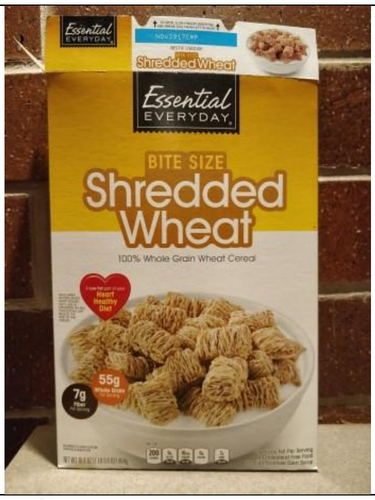
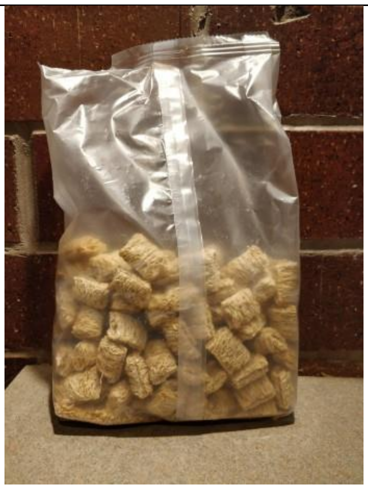
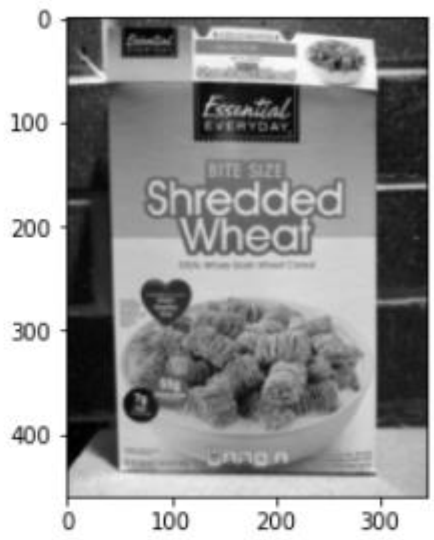
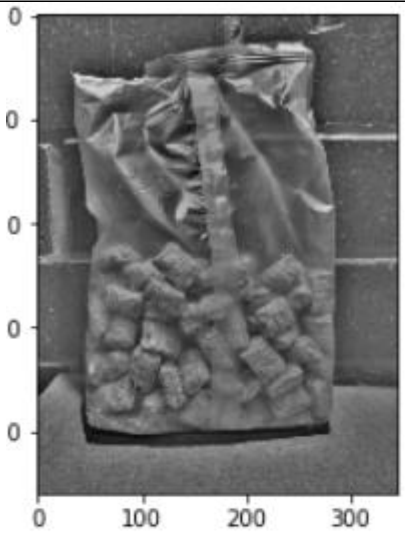
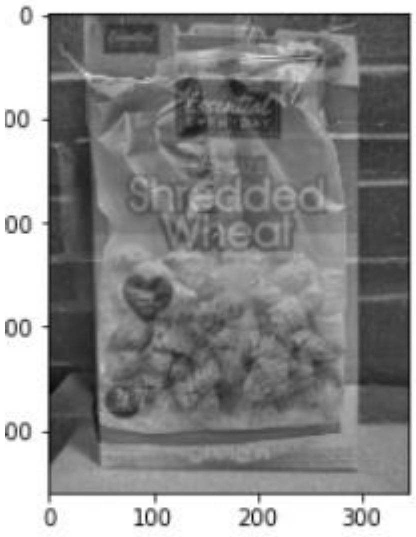
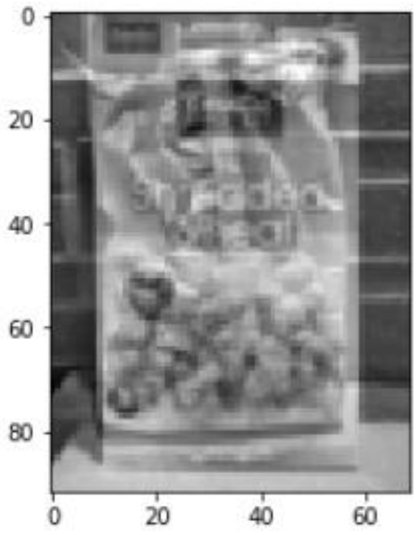
You will provide the following for ***3 different examples*** (1 provided pair, 2 pairs of your own):

- two input images
- two *filtered* input images
- two generated hybrid image (at different resolutions, similar to images C and D above)
- two σ values (one for each filter)

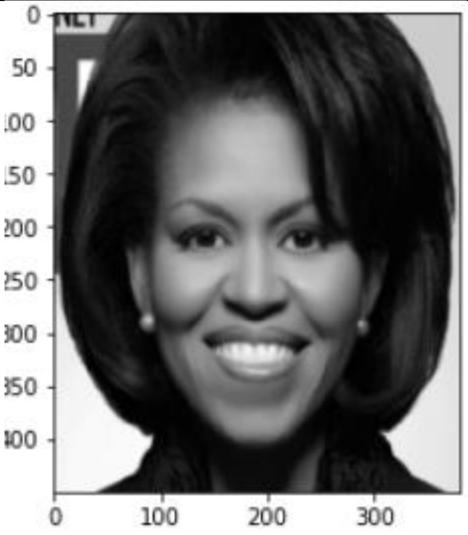
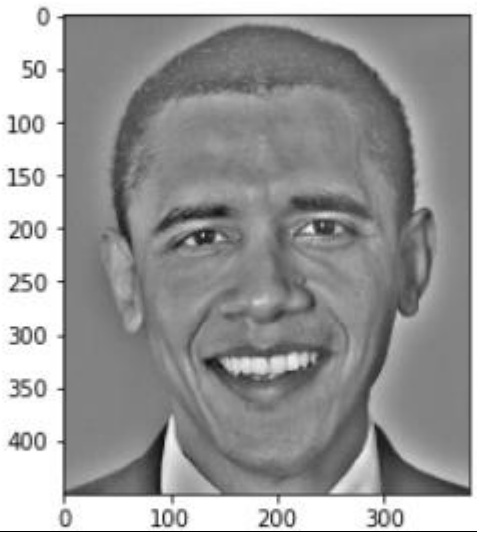
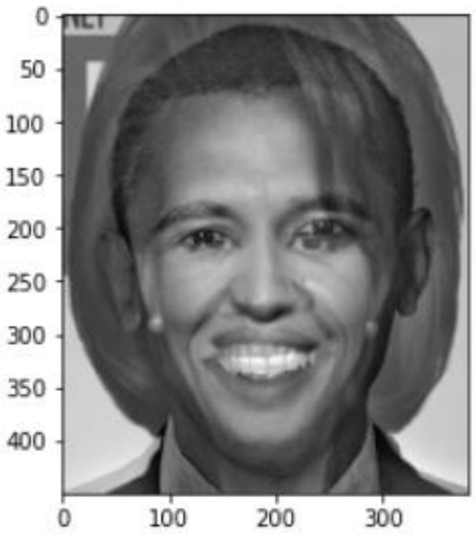
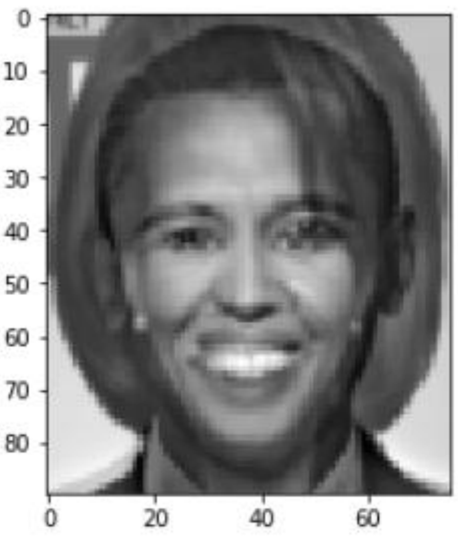
You will provide the following as further discussion overall:

- Explanation of how you chose the σ values
- Discussion of how successful your examples are + any interesting observations

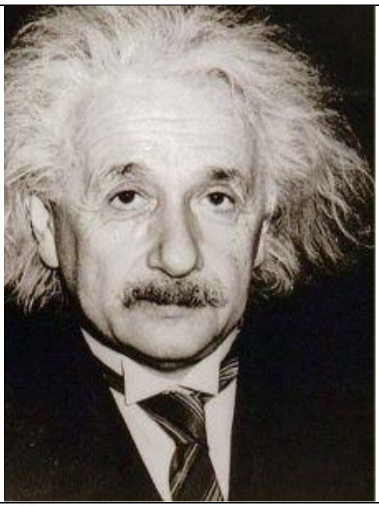
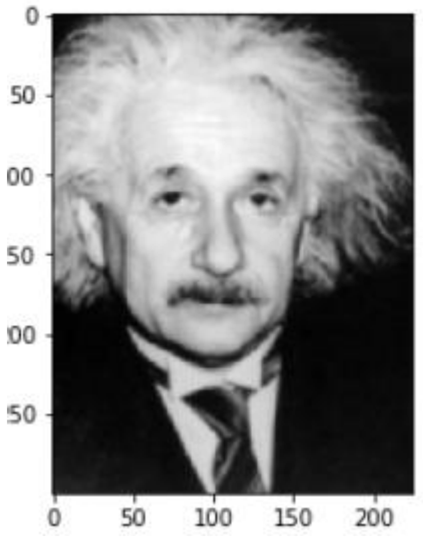
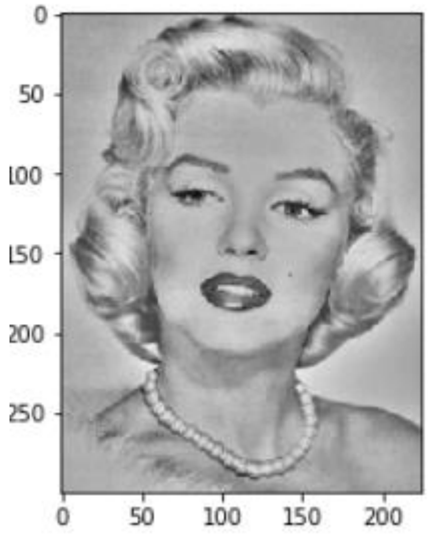
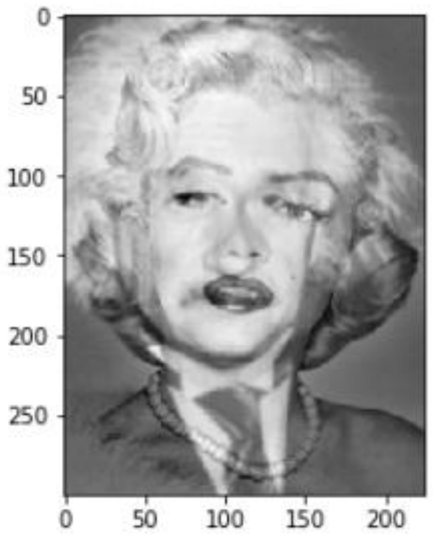
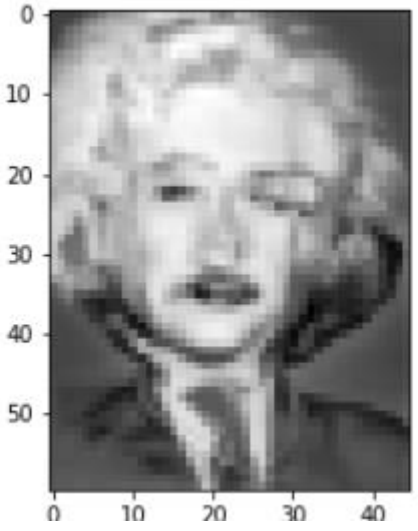
Example 1:

Input Images		
Filtered input images		
Two generated hybrid images		
Sigma values	1	15

Example 2:

Input Images		
Filtered input images		
Two generated hybrid images		
Sigma values	1	15

Example 3:

Input Images		
Filtered input images		
Two generated hybrid images		
Sigma values	1	15

Discussion: The sigma values were chosen through trial and error to have a good balance between the first and second images. The values of 1 and 15 for the low-pass and high-pass filters seems to work well for the examples I have used.

It was hard to perfectly center the images and so, the hybrid images do not look perfect. This process is much harder to do with faces as all the features have to be matched up perfectly for the image to look reasonable.

Part 2 Scale-Space Blob Detection:

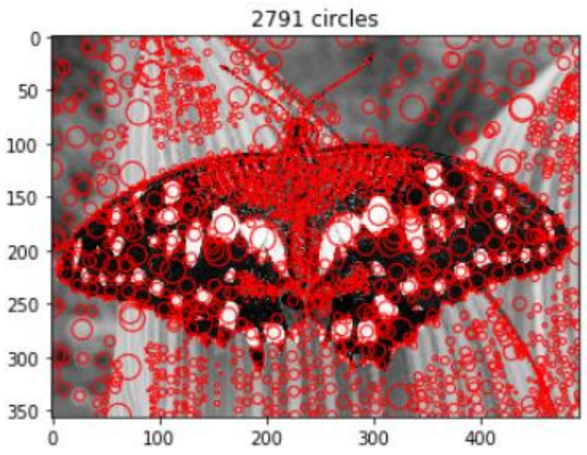
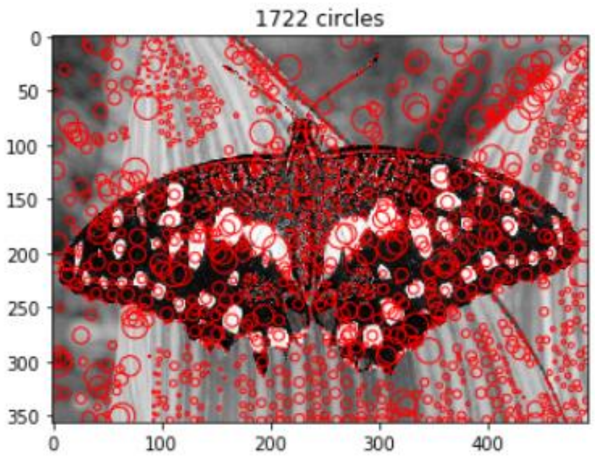
You will provide the following for ***8 different examples*** (4 provided, 4 of your own):

- original image
- output of your circle detector on the image
- running time for the "efficient" implementation on this image
- running time for the "inefficient" implementation on this image

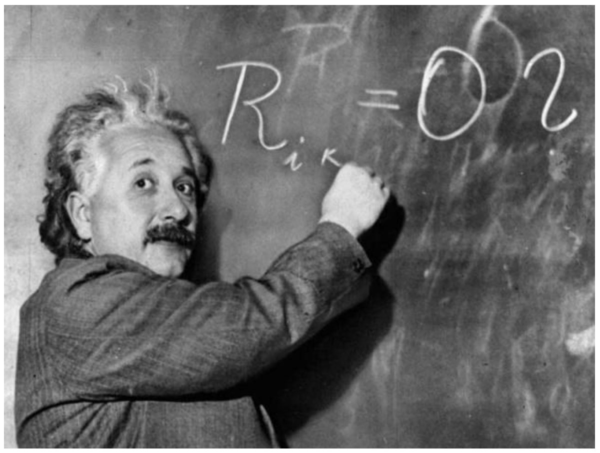
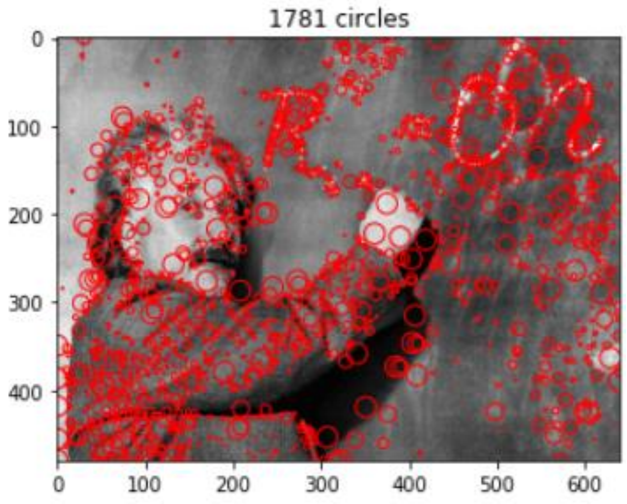
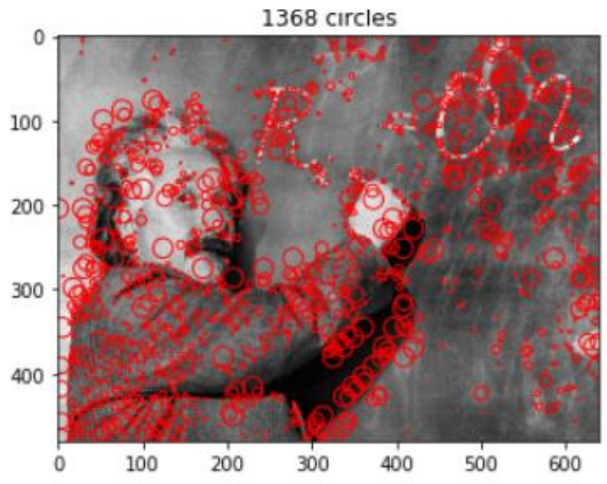
You will provide the following as further discussion overall:

- Explanation of any "interesting" implementation choices that you made.
- Discussion of optimal parameter values or ones you have tried

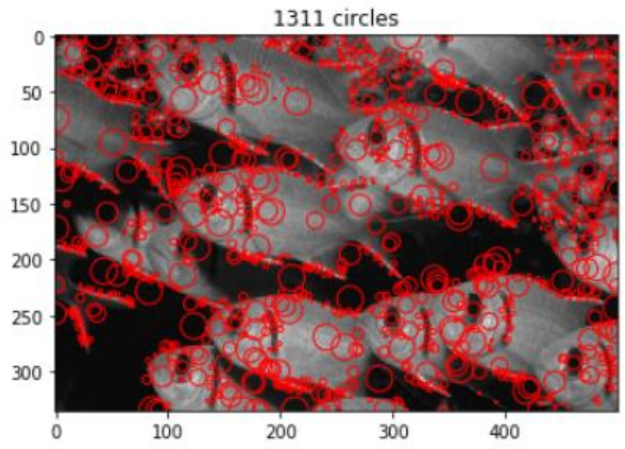
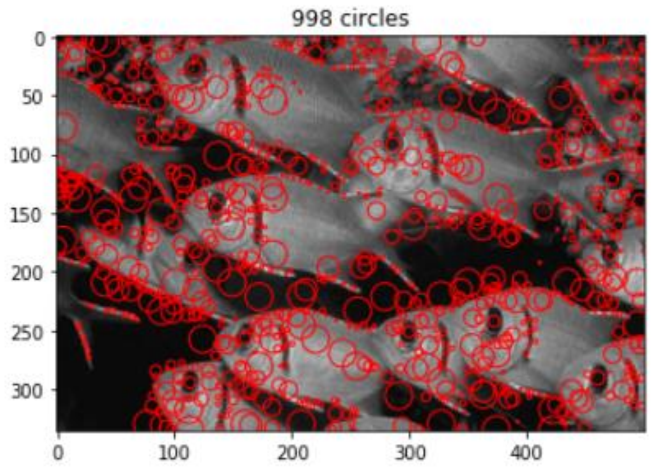
Example 1:

Input image		Runtime (seconds)
Output image (Inefficient)		2.431
Output image (Efficient)		2.339

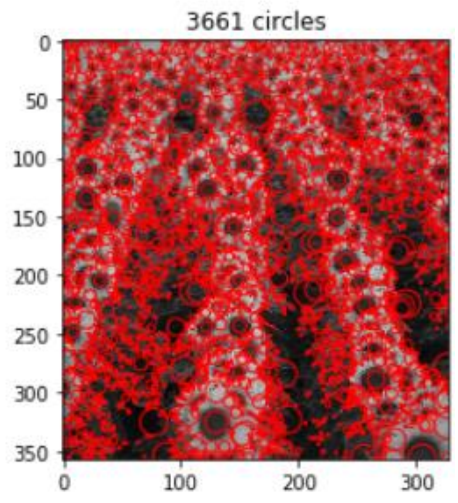
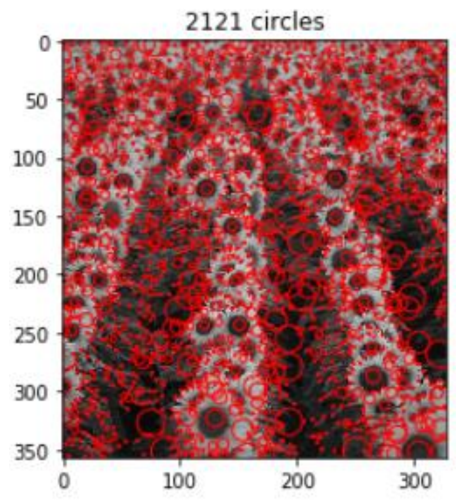
Example 2:

Input image		Runtime (seconds)
Output image (Inefficient)		4.276
Output image (Efficient)		4.064

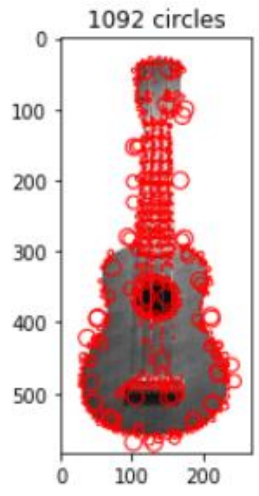
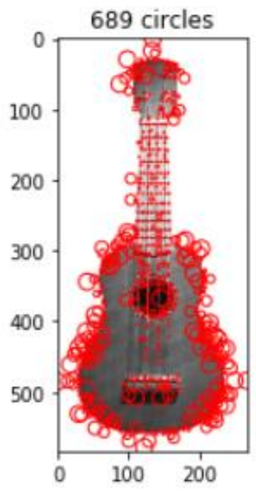
Example 3:

Input image		Runtime (seconds)
Output image (Inefficient)		2.185
Output image (Efficient)		2.459

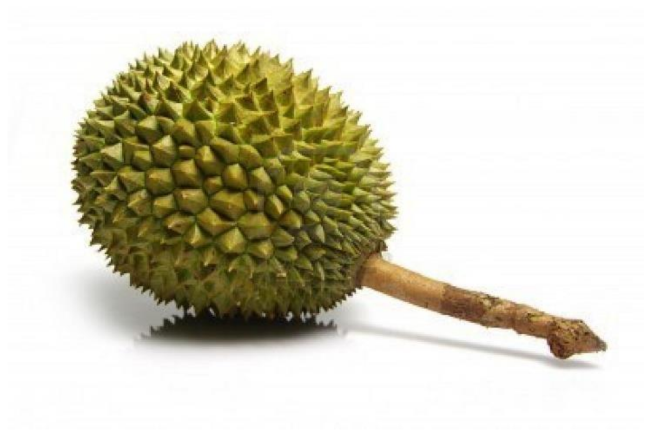
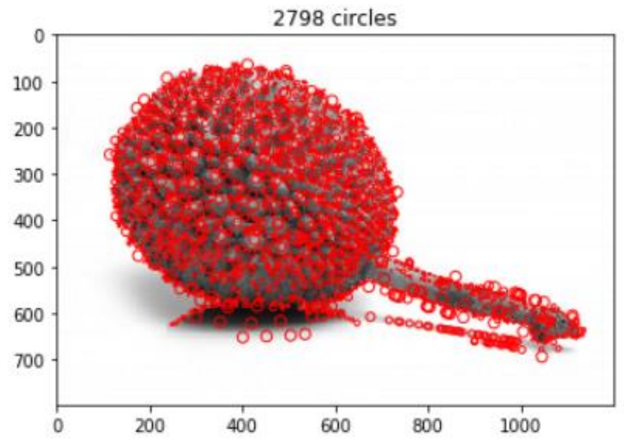
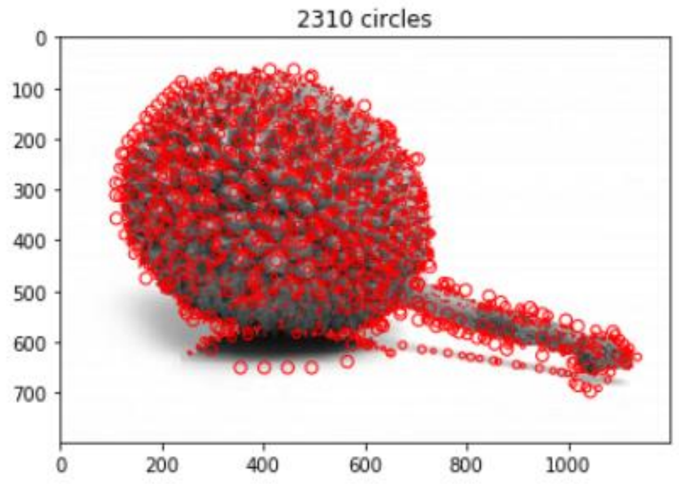
Example 4:

Input image		Runtime (seconds)
Output image (Inefficient)		1.536
Output image (Efficient)		1.565

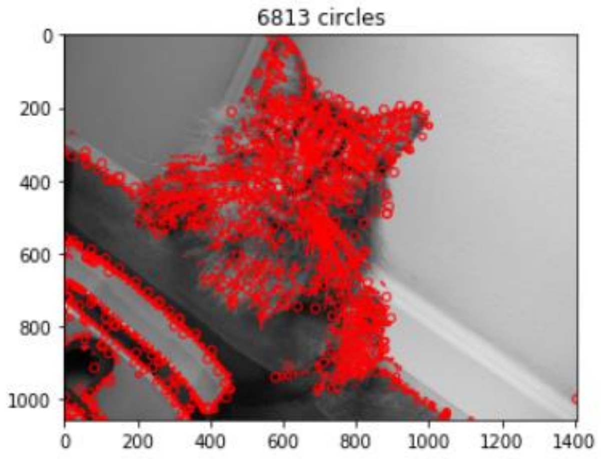
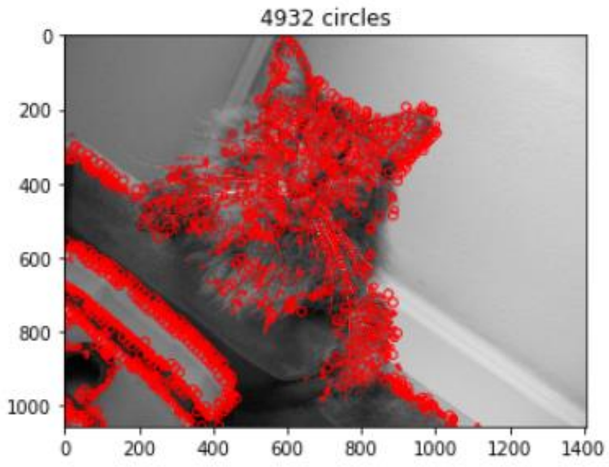
Example 5:

Input image		Runtime (seconds)
Output image (Inefficient)		2.038
Output image (Efficient)		2.129

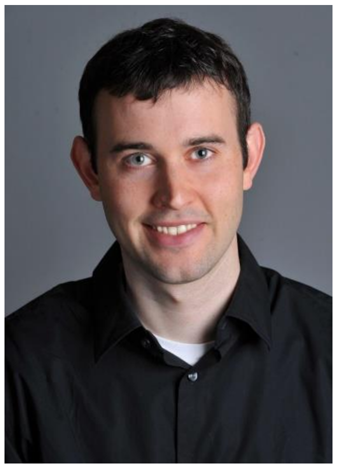
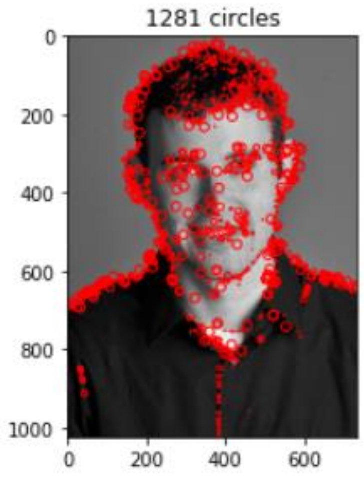
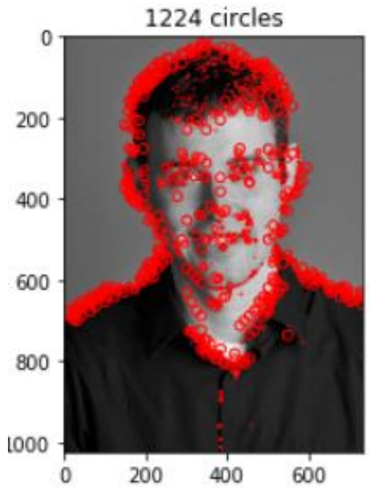
Example 6:

Input image		Runtime (seconds)
Output image (Inefficient)		12.992
Output image (Efficient)		12.401

Example 7:

Input image		Runtime (seconds)
Output image (Inefficient)		19.171
Output image (Efficient)		20.561

Example 8:

Input image		Runtime (seconds)
Output image (Inefficient)		11.145
Output image (Efficient)		10.429

Discussion:

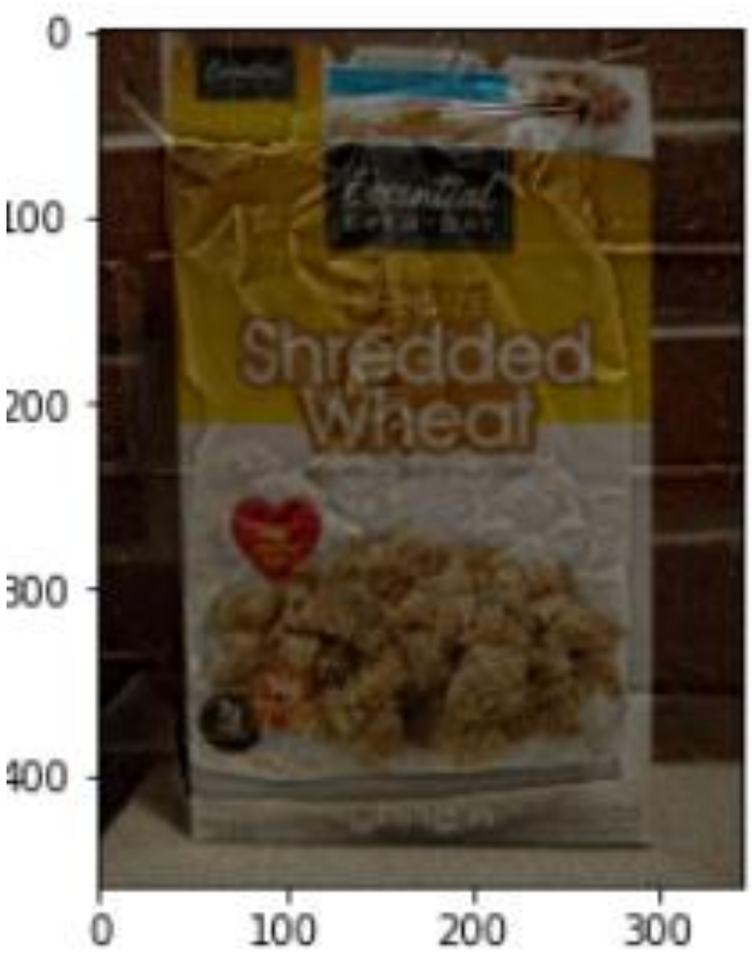
I tried different values for n and k but settled on 12 and 1.25 as this took a reasonable amount of time to compute the blobs and the results looked quite like what I would expect for each image.

Bonus:

Hybrid Images Extra Credit

- Discussion and results of any extensions or bonus features you have implemented for Hybrid Images

  Implemented Hybrid images for colored images as well by applying the filter to individual images first, then stacking them together to form a colored output image. This image is a bit darker than expected but the hybridization can be seen clearly.



Blob-Detection Extra Credit

- Discussion and results of any extensions or bonus features you have implemented for Blob-Detection