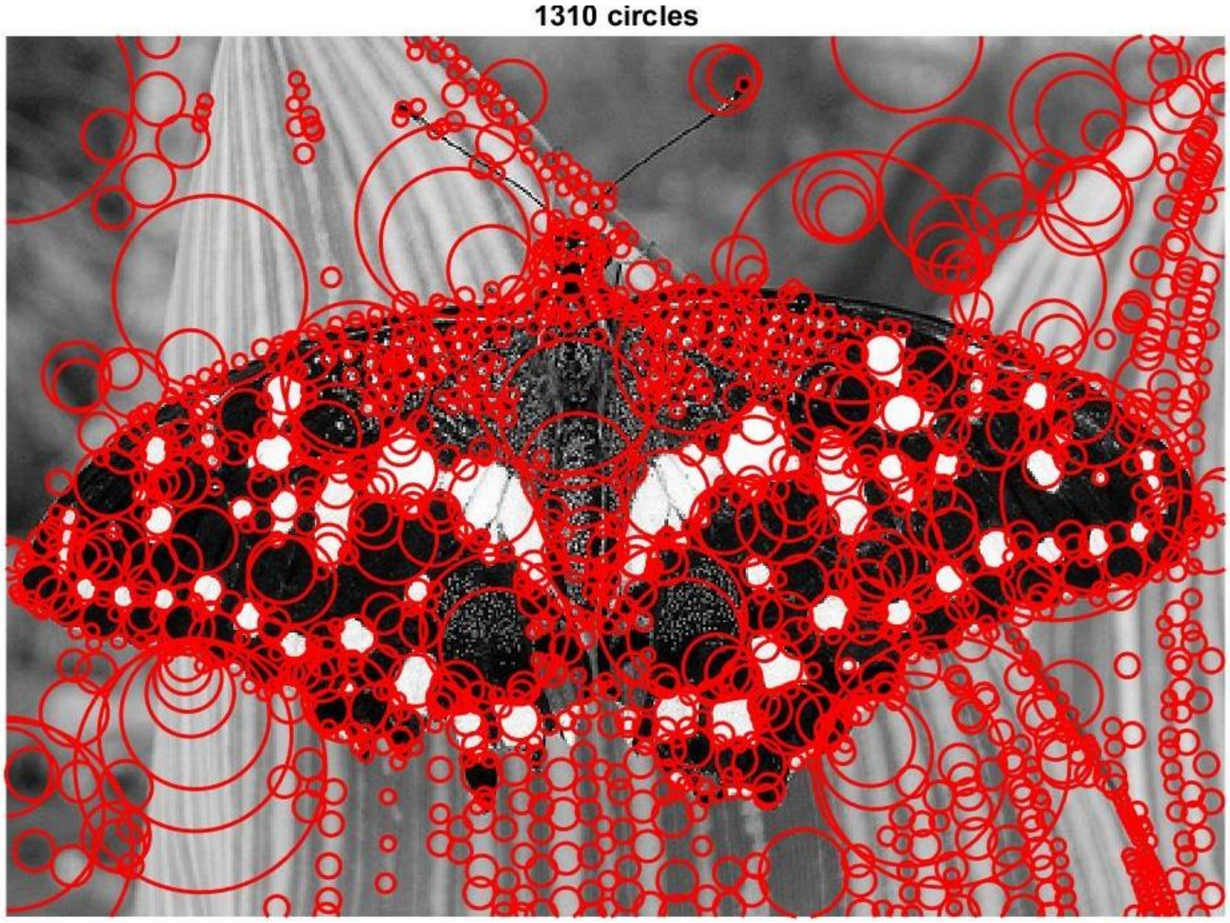
Computer Vision and Image Processing
Homework 2: Scale-space blob detection

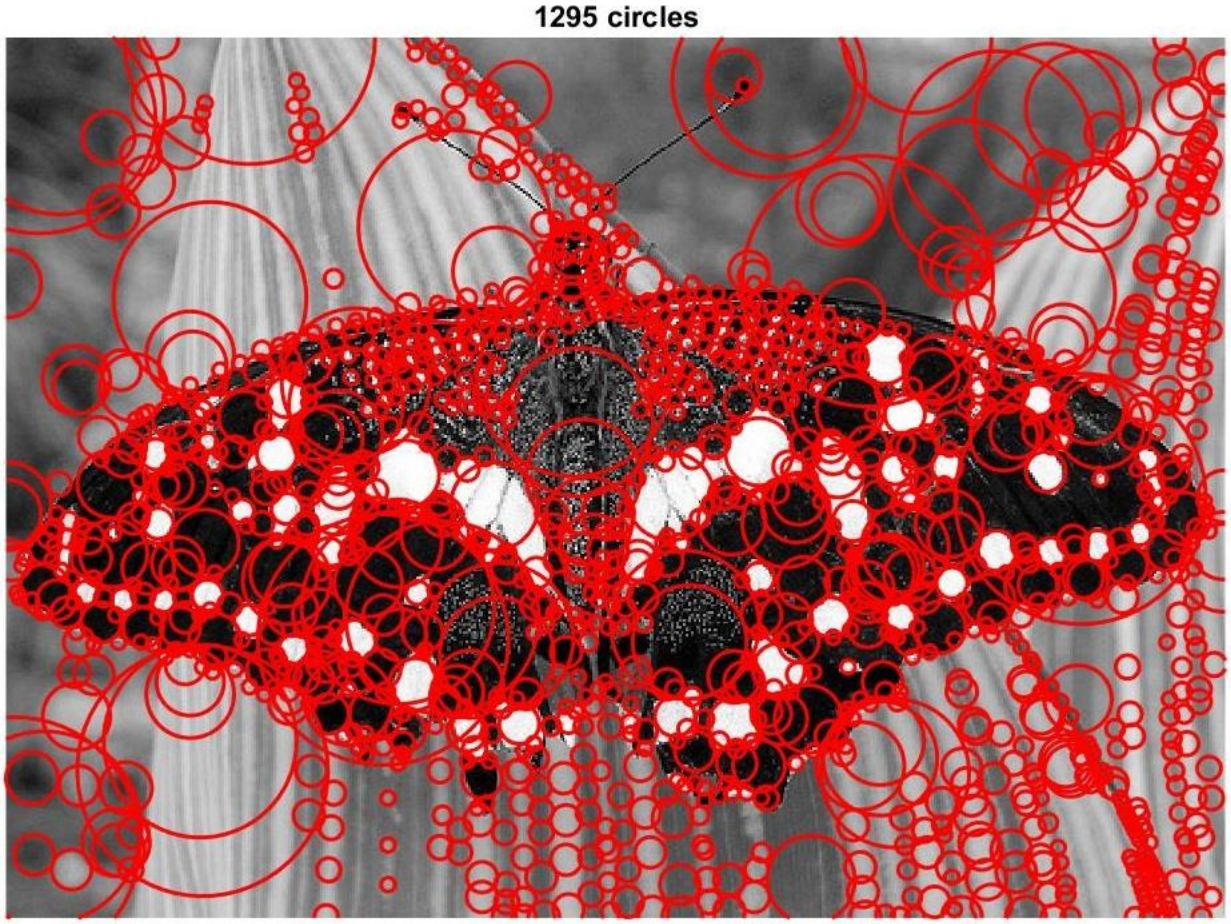
1. Below are the output images of circle detector with the original image [Newly Added one]: First output images are with down sampling and second output images are with increasing sigma/kernel size respectively. Time comparison is provided with each result.

Butterfly.jpg

Image	Used Implementation	Elapsed Time (Seconds)	No of Circles
Butterfly.jpg	Increasing sigma/kernel	2.447257	1295
	Down Sampling Image	0. 344062	1310



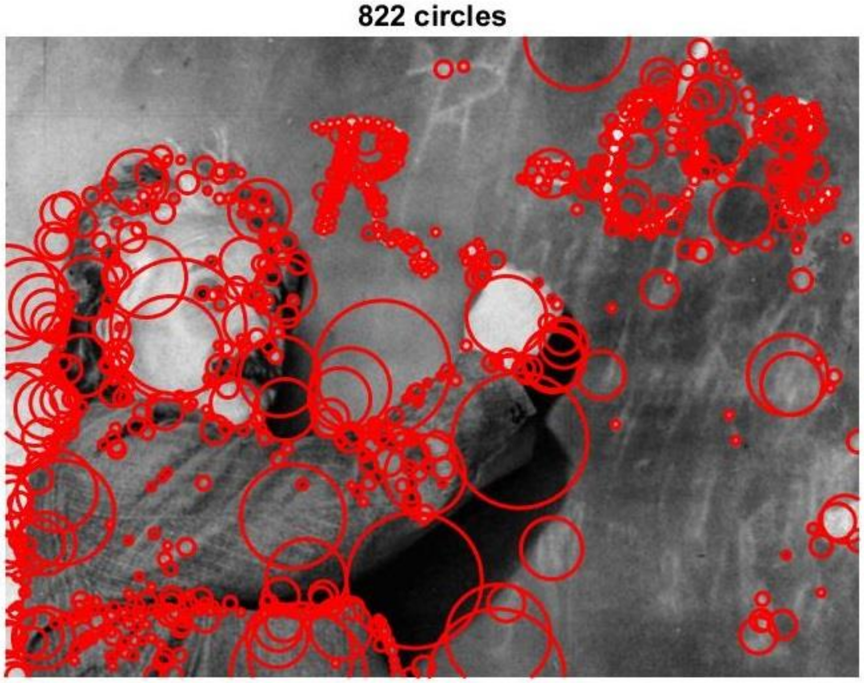
Implementation using Down Sampling.



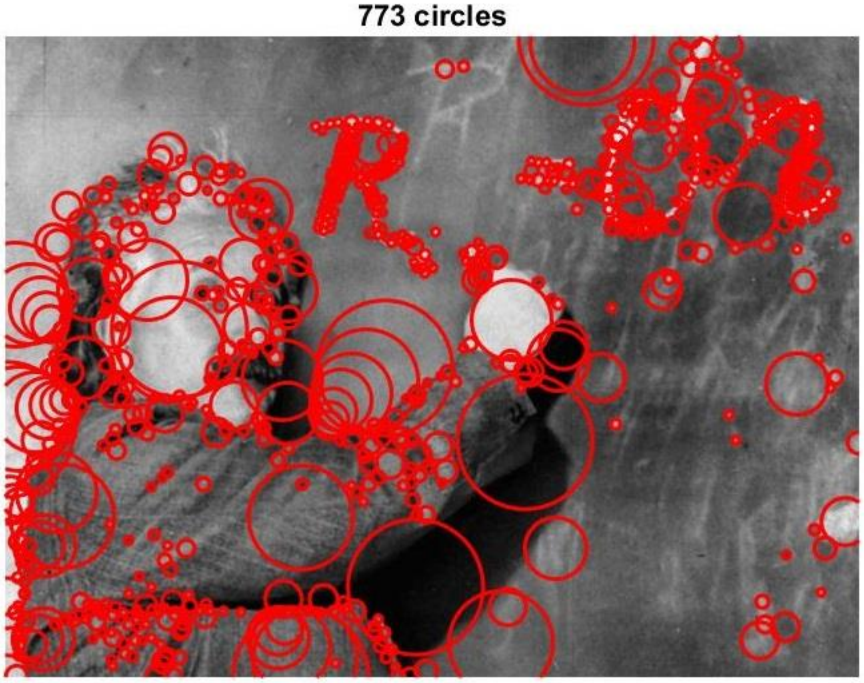
Implementation using increasing sigma/kernel

Einstien.jpg

Image	Used Implementation	Elapsed Time (Seconds)	No of Circles
einstein.jpg	Increasing sigma/kernel	4.068043	773
	Down Sampling Image	0.533505	822



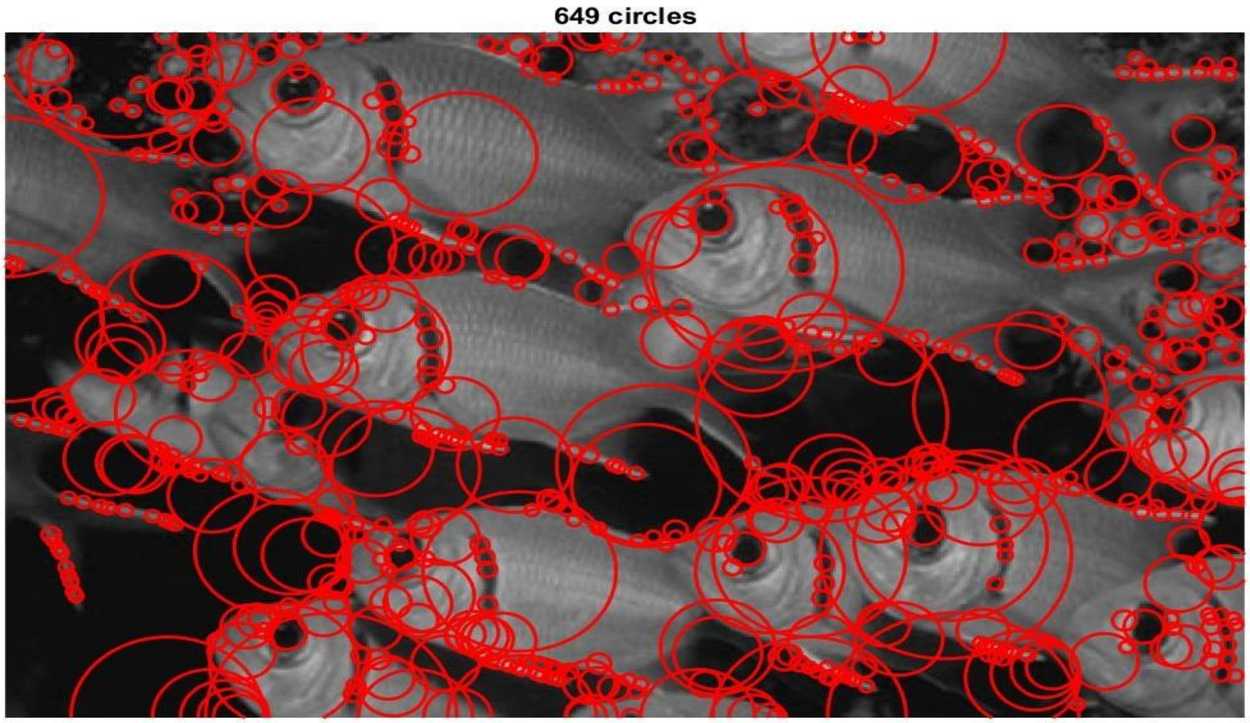
Implementation using increasing sigma/kernel



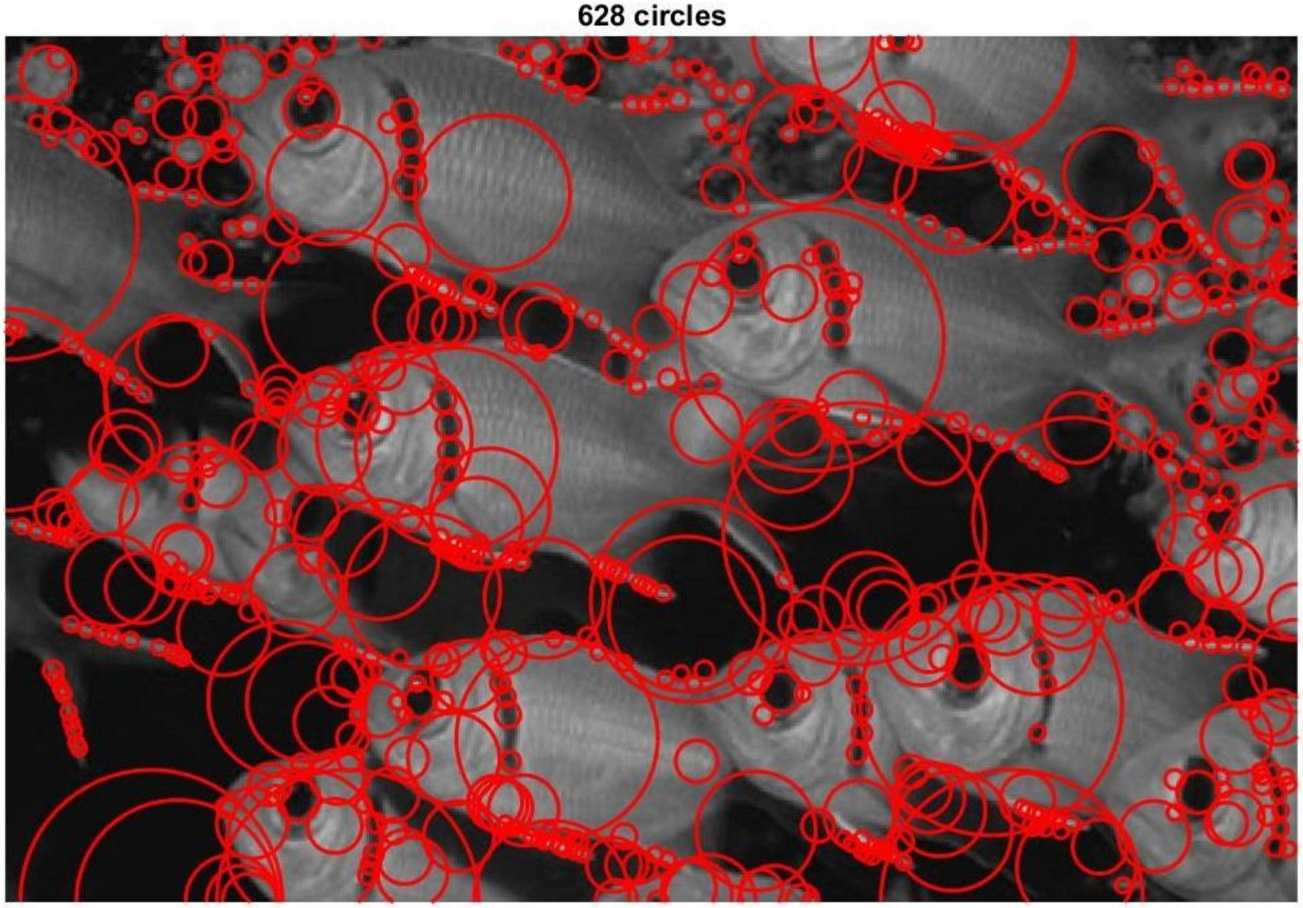
Implementation using down sampling

Fishes.jpg

Image	Used Implementation	Elapsed Time (Seconds)	No of Circles
fishes.jpg	Increasing sigma/kernel	2.373438	628
	Down Sampling Image	0.336204	649



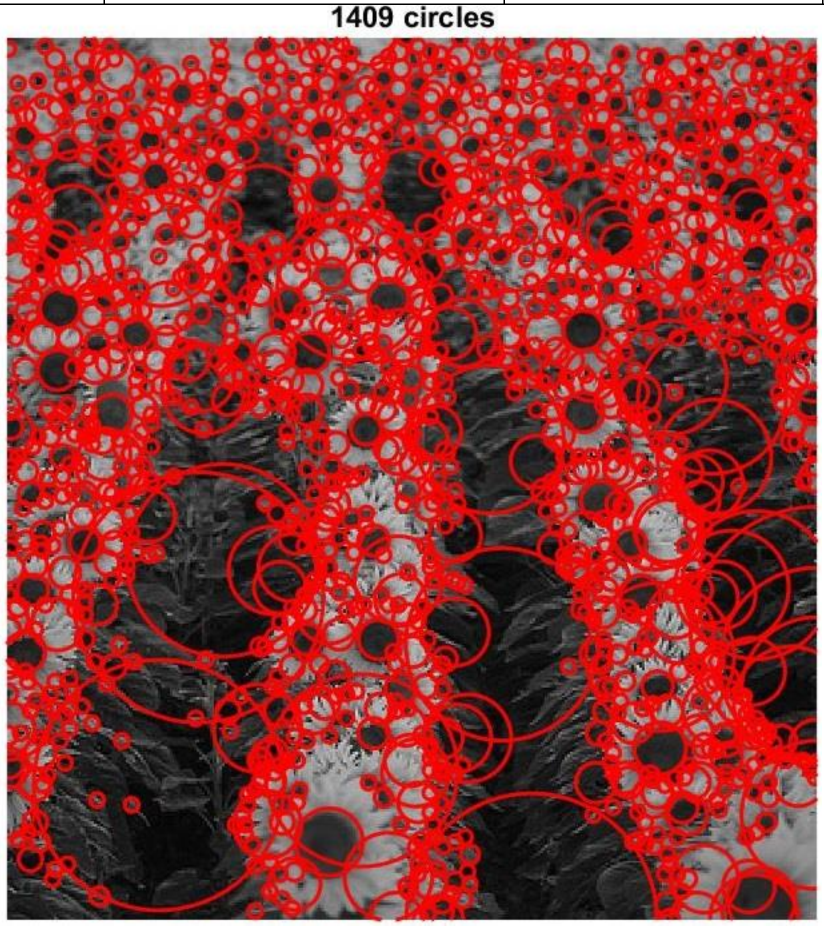
Implementation using down sampling image



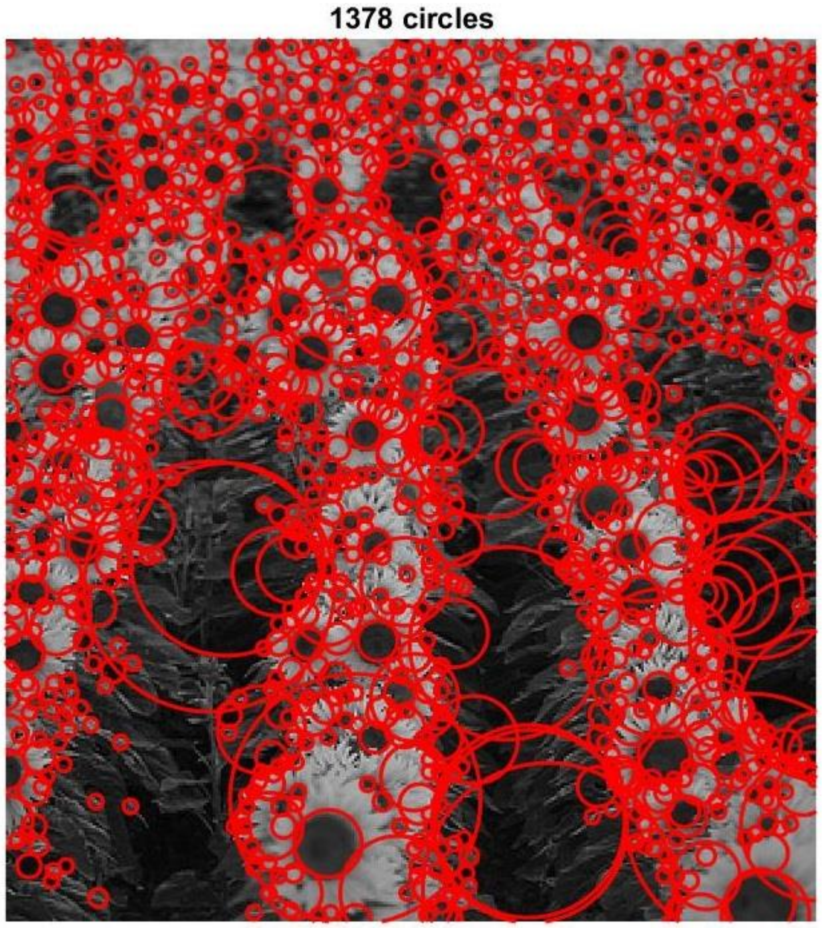
Implementation using increasing sigma/kernel

Sunflowers.jpg

Image	Used Implementation	Elapsed Time (Seconds)	No of Circles
sunflowers.jpg	Increasing sigma/kernel	1.632530	1378
	Down Sampling Image	0.256577	1409



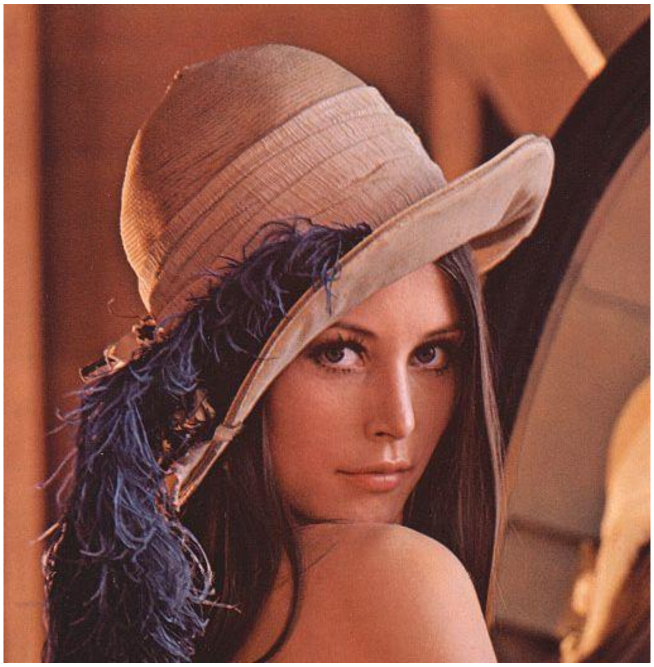
Implementation Using down sampling

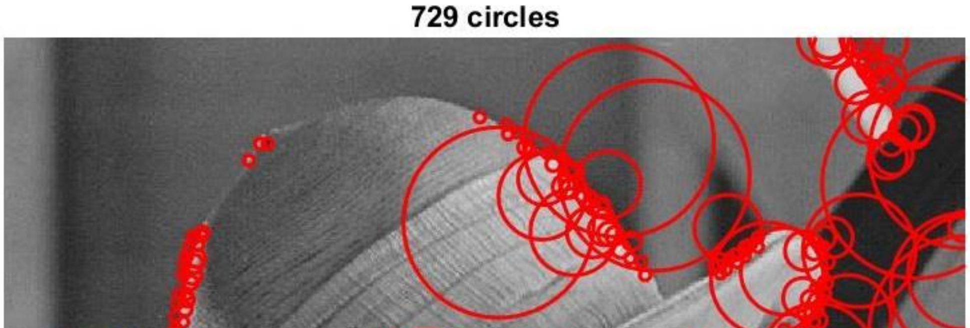


Implementation using increasing sigma/kernel

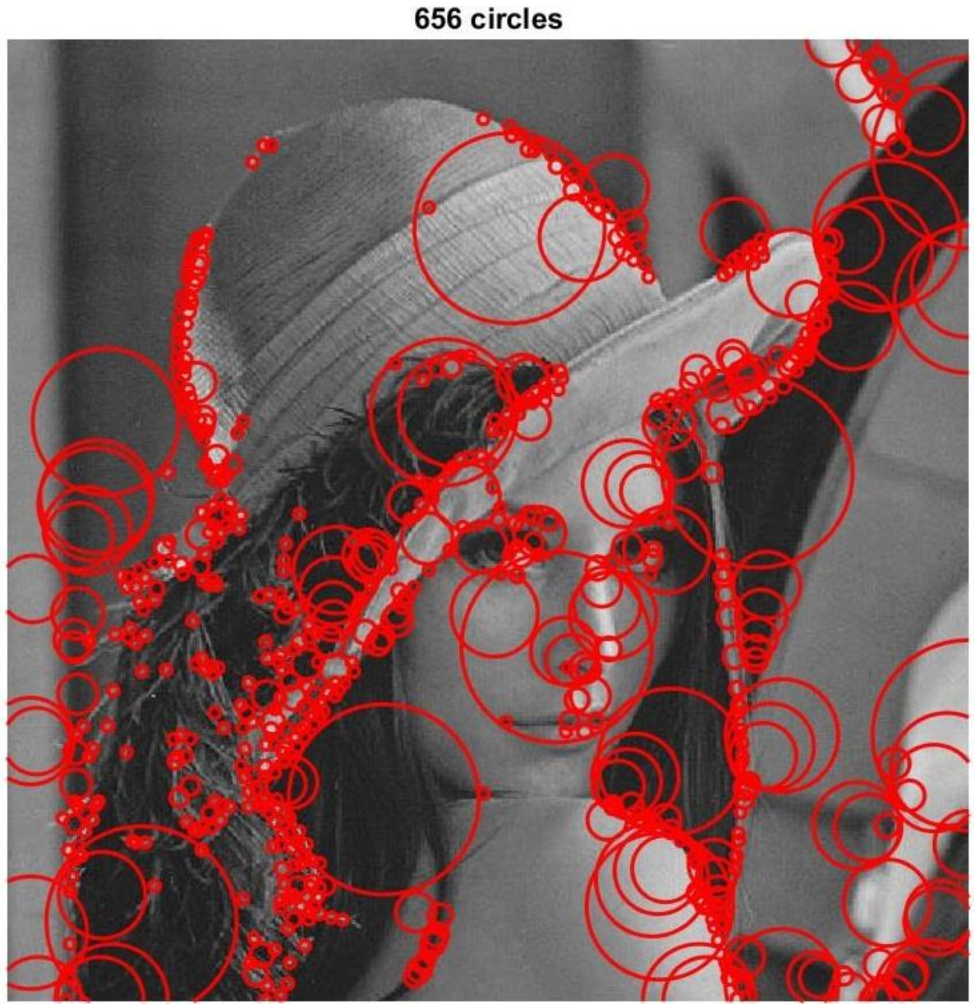
Lena.jpg

Image	Used Implementation	Elapsed Time (Seconds)	No of Circles
Lena.jpg	Increasing sigma/kernel	3.475315	656
	Down Sampling Image	0.465286	729





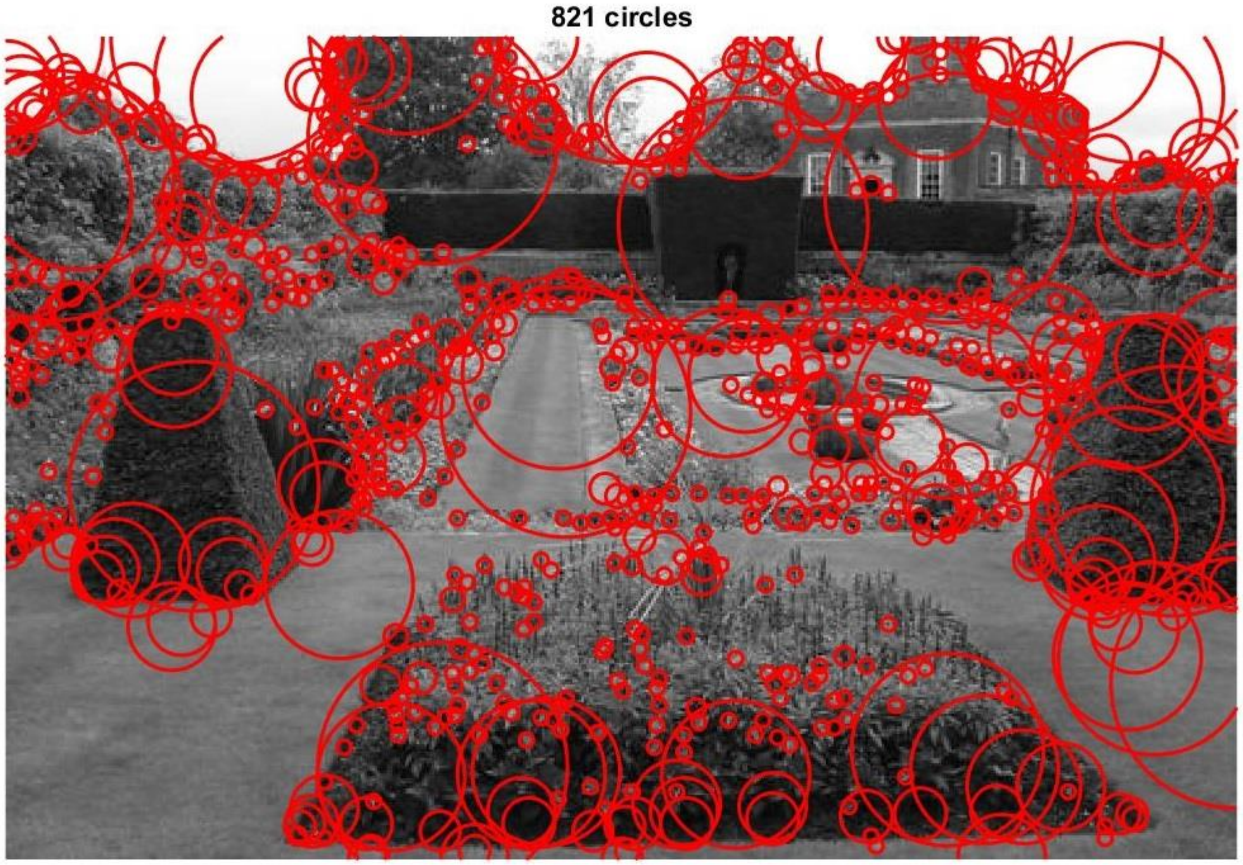
Implementation using down sampling



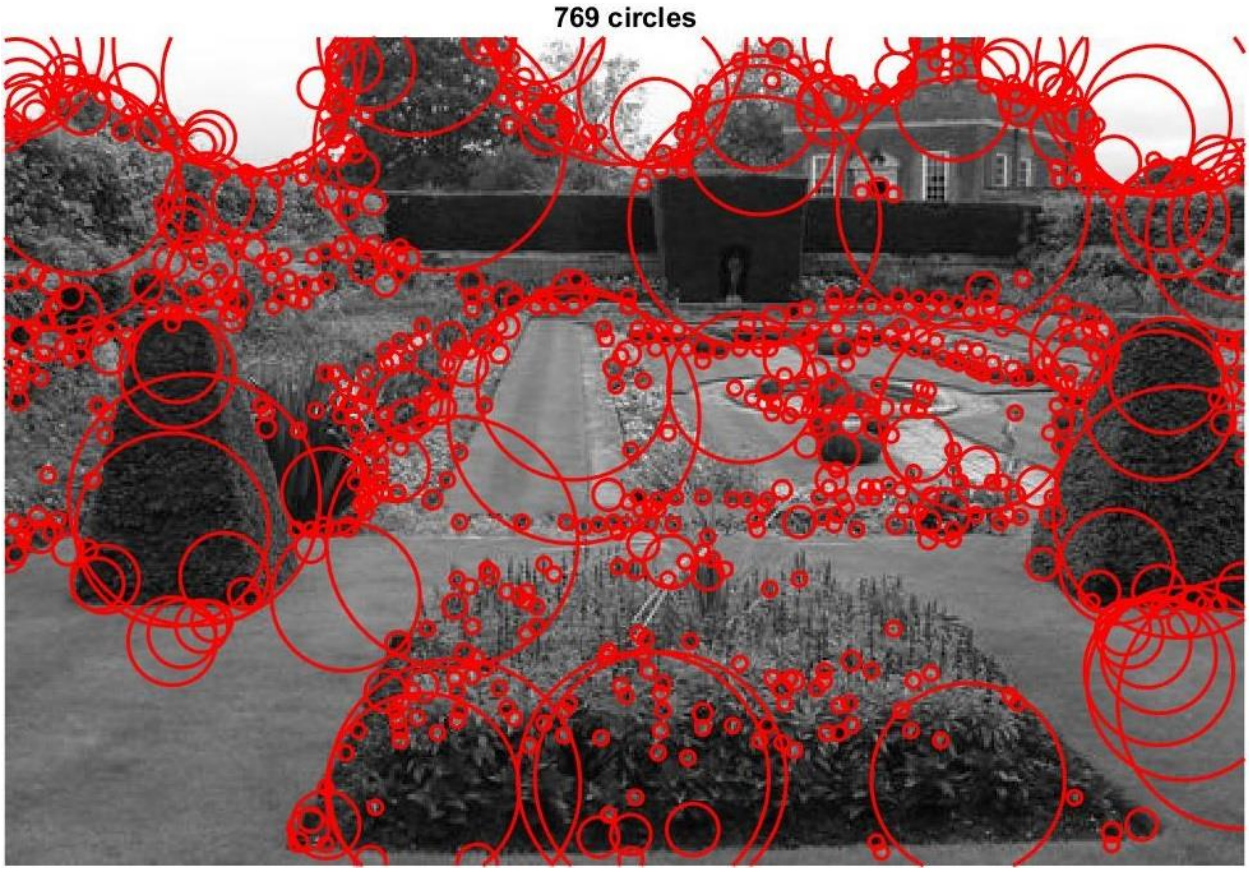
Implementation using increasing sigma/kernel

Garden.jpg

Image	Used Implementation	Elapsed Time (Seconds)	No of Circles
garden.jpg	Increasing sigma/kernel	2.301280	769
	Down Sampling Image	0.330301	821



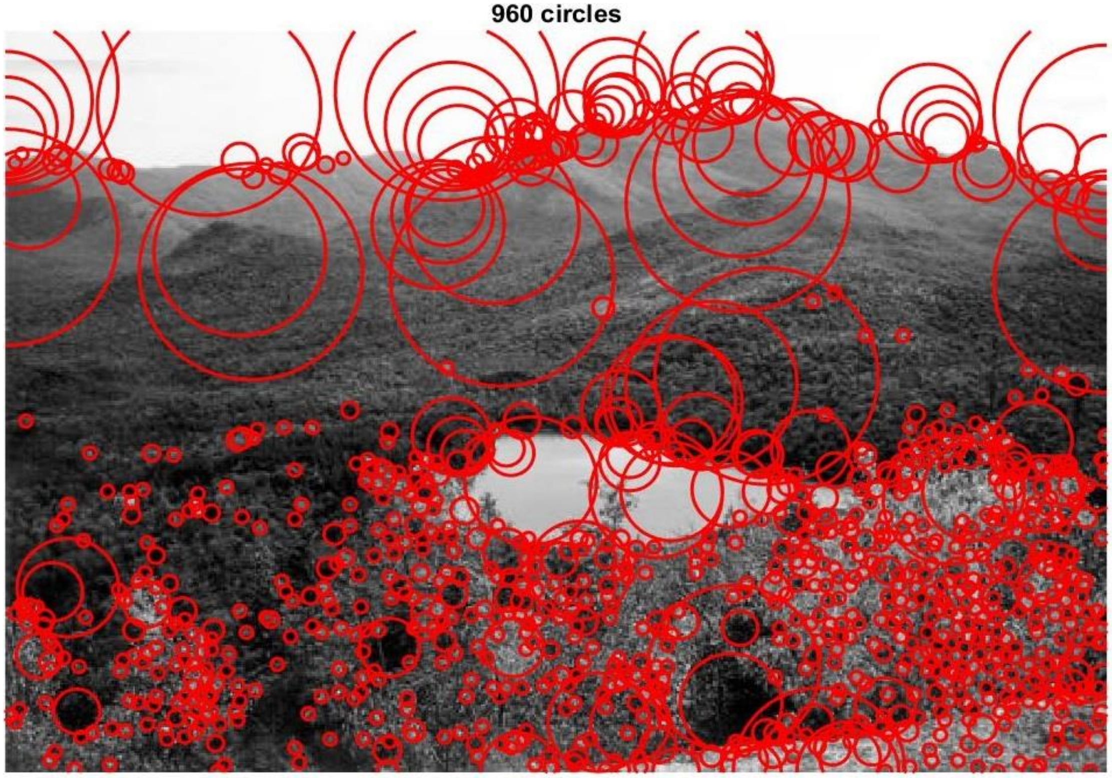
Implementation using down sampling



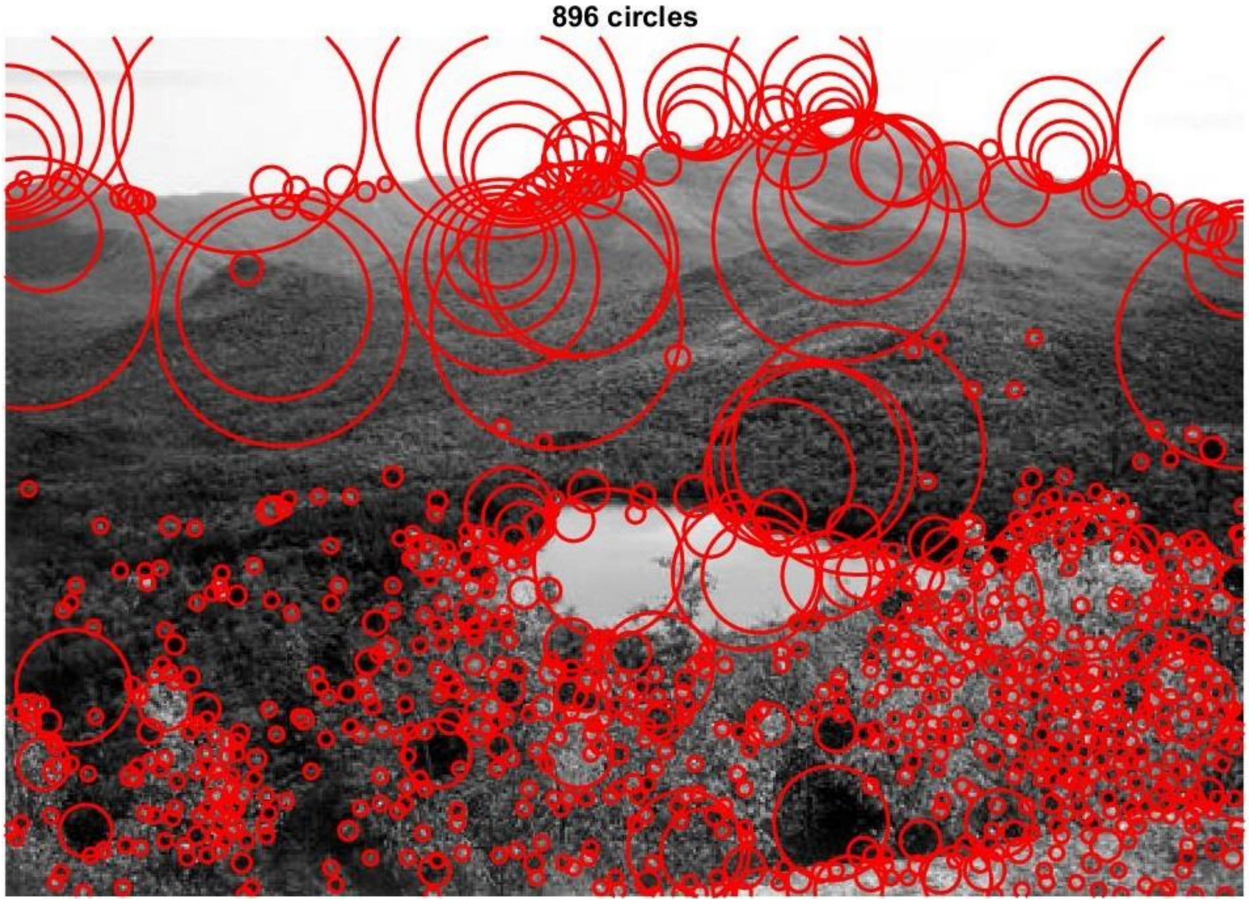
Implementation using increasing sigma/kernel

Mountain.jpg

Image	Used Implementation	Elapsed Time (Seconds)	No of Circles
mountain.jpg	Increasing sigma/kernel	2.470367	869
	Down Sampling Image	0.337340	960



Implementation using down sampling

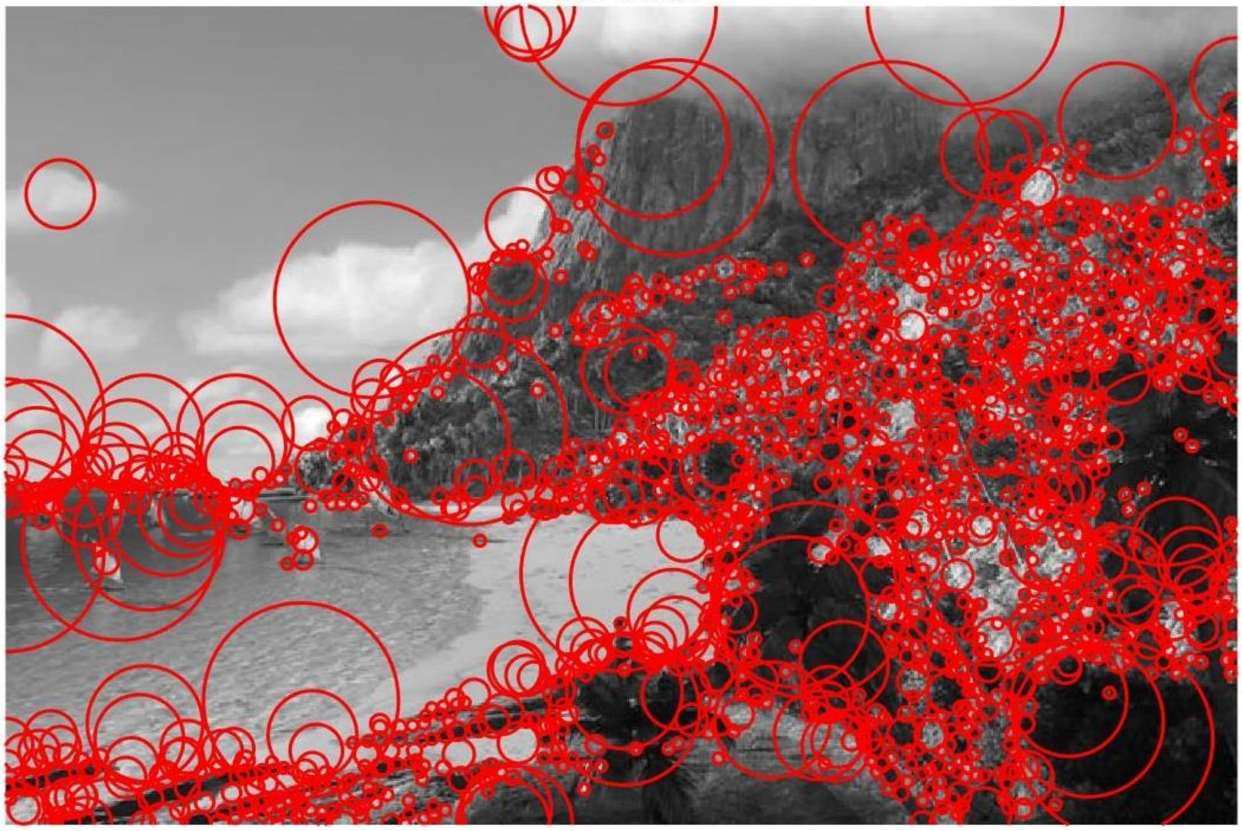


Implementation using increasing sigma/kernel

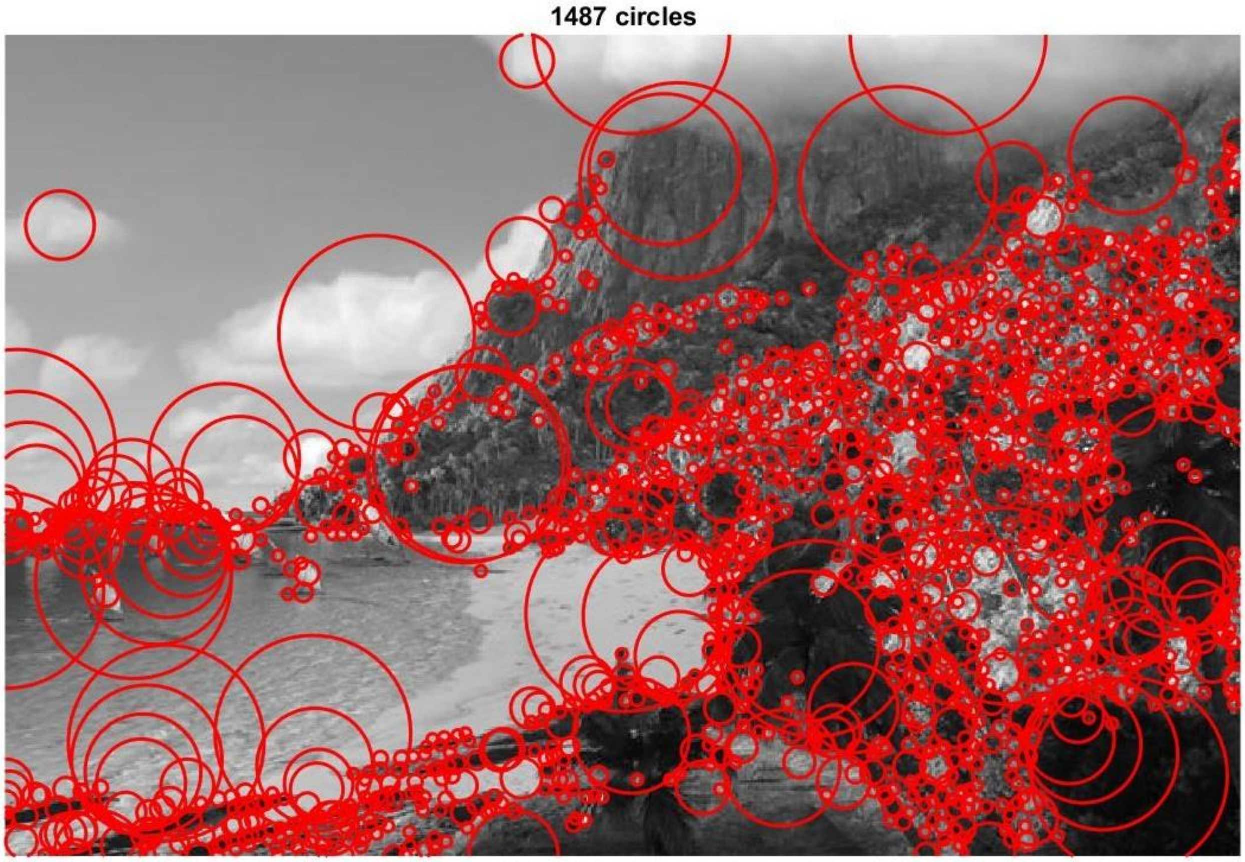
Ocean.jpg

Image	Used Implementation	Elapsed Time (Seconds)	No of Circles
ocean.jpg	Increasing sigma/kernel	2.470367	1487
	Down Sampling Image	0.337340	1520

1520 circles



Implementation using down sampling



Implementation using increasing sigma/kernel

Final Results:

Image	Used Implementation	Elapsed Time (Seconds)	No of Circles
butterfly.jpg	Increasing sigma/kernel	2.447257	1295
	Down Sampling Image	0. 344062	1310
einstein.jpg	Increasing sigma/kernel	4.068043	773
	Down Sampling Image	0.533505	822
fishes.jpg	Increasing sigma/kernel	2.373438	628
	Down Sampling Image	0.336204	649
sunflowers.jpg	Increasing sigma/kernel	1.632530	1378
	Down Sampling Image	0.256577	1409
Lena.jpg	Increasing sigma/kernel	3.475315	656
	Down Sampling Image	0.465286	729
garden.jpg	Increasing sigma/kernel	2.301280	769
	Down Sampling Image	0.330301	821
mountain.jpg	Increasing sigma/kernel	2.470367	869
	Down Sampling Image	0.337340	960
ocean.jpg	Increasing sigma/kernel	2.470367	1487
	Down Sampling Image	0.337340	1520

In all the cases down sampling image implementation takes significantly less time.

2. An explanation of any \interesting" implementation choices that you made.

Implementation details for both the methods are as below:

homwwork_2_solution_main: This is the main file from where all other codes have been called. Here we can specify all the parameter values like sigma, scale factor, threshold. We can test with different parameter value and different set of images very easily from here.

blob_detection: This script having both the implementation detail for blob_detection. Here I have created two functions for calculating scale space with increasing sigma/kernel size and down sampling image by scale factor.

downsampling_image: This provides implementation using down sampling of image technique

increasing_sigma: This provides implementation using increasing sigma/kernel value.

nonmaximum_supressed_values: Non maximum suppression using ordfilt2(cim,9,ones(3,3)) with window size 3x3. Then calculating the maximum response value over the 3rd dimention scale and selecting maximum.

show_all_circles: implementation of this already provided with the homework

For non-maximum suppression I am using ordfilt2 with 3x3 window size. And calculating max value among the all scale to get maximum along scale space.
In down sampling image implementation, I am using imresize method with default bicubic interpolation.

Both the implementation giving almost same results but down sampling the image is very fast. Using large sigma/kernel size computation is get increased significantly.

As we know that:
Laplacian (blob) response is invariant w.r.t. rotation and scaling.
Blob location and scale is covariant w.r.t. rotation and scaling

So blob will be get detected irrespective of scaling.

By using Laplacian filter the magnitude of the Laplacian response will achieve a maximum at the center of the blob, provided the scale of the Laplacian is “matched” to the scale of the blob. To get maximum response, the zeros of the Laplacian have to be aligned with the circle. This will help determining the radius of all the blob points.

3. An explanation of parameter values you have tried and which ones you found to be optimal.

In my implementation I have taken below values for final results:
initial sigma = 2
scale factor = 1.3
threshold = .007
No of Scale Space = 12

I have tried multiple combination for above values and below are the result of time and number of block detected with different value and time taken to execute the both implementation:

With sigma = 3 : blob_detection(img, 3, 1.3, .007, 12);
For Butterfly image:
For filtering Image with increasing sigma/kernel : Elapsed time is **5.149455** seconds.
No of detected blob : 875

For filtering Image by down sampling: Elapsed time is **0.302557** seconds.
No. of detected blob: 876

With sigma = 1 : blob_detection(img, 1, 1.18, .007, 12);
For Butterfly image:
For filtering Image with increasing sigma/kernel : Elapsed time is 0.457900 seconds.
No of detected blob : 2533

For filtering Image by down sampling: Elapsed time is 0.334096 seconds.
No. of detected blob: 2694

With sigma = 1 : blob_detection(img, 1, 1.5, .007, 12);
For Butterfly image:
For filtering Image with increasing sigma/kernel : Elapsed time is 9.861793seconds.
No of detected blob : 2174

For filtering Image by down sampling: Elapsed time is 0.330170seconds.
No. of detected blob: 2370

With sigma = 2 : blob_detection(img, 2, 1.3, .004, 12);
For Butterfly image:
For filtering Image with increasing sigma/kernel : Elapsed time is 2.509917 seconds.
No of detected blob : 1387

For filtering Image by down sampling: Elapsed time is 0.356615 seconds.
No. of detected blob: 1423

With sigma = 2 : blob_detection(img, 2, 1.3, .007, 10);
For Butterfly image:

For filtering Image with increasing sigma/kernel : Elapsed time is 1.708114 seconds.
No of detected blob : 1330

For filtering Image by down sampling: Elapsed time is 0.412076 seconds.
No. of detected blob: 1339

After analyzing multiple results by changing sigma, scale factor, threshold and number of scale space over multiple images I have chosen the appropriate value of these for final results submitted in this report. If we increase the threshold value higher then number of detected blobs are less. If we increase scale factor then value of sigma will be high over the iteration then kernel size will increase and it will increase computation time rapidly. With increase no of iteration blob will be detected more with large size. Sample statistics provide above with few variation.