CSE 573 Computer Vision | Ting Zhou
Homework 2 | 50169020

**1. Outputs of all images and the comparison of the two methods**

Parameters:

num_scales = 15; sigma = 2; scale_factor = sqrt(sqrt(2)); threshold = 0.1

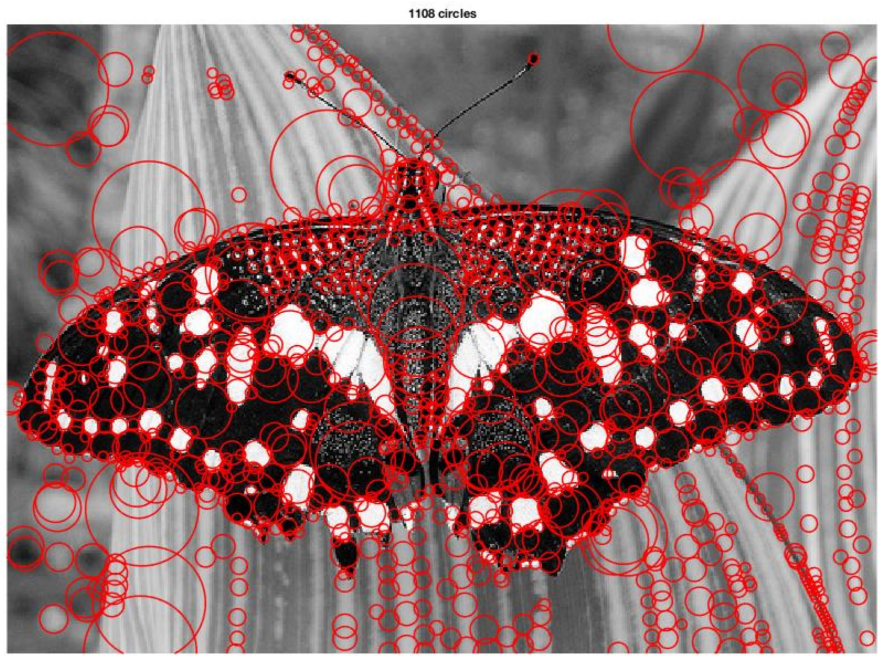


Figure1. Scale Space Generated by Increasing Filter Size (butterfly)

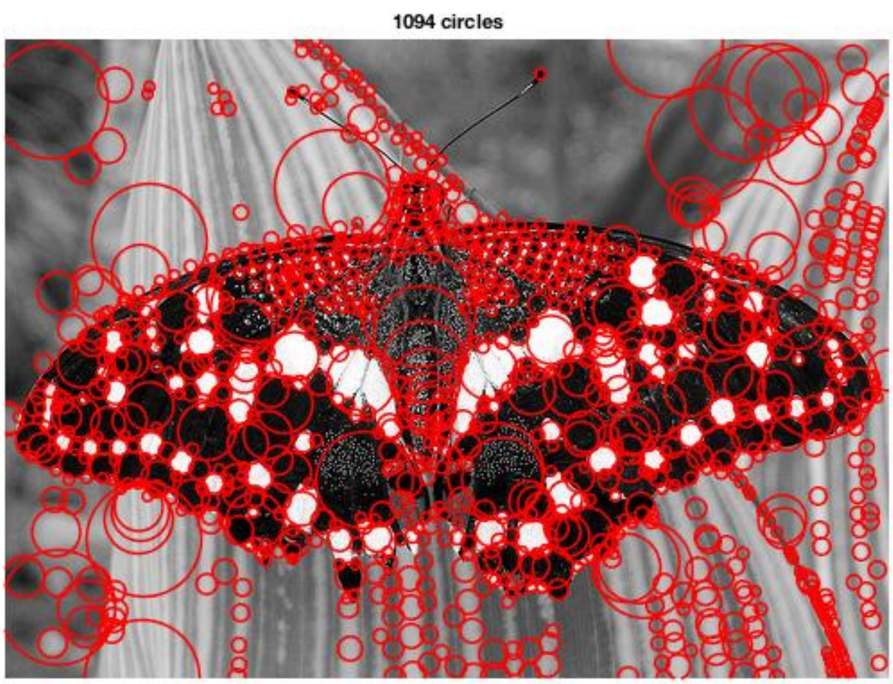


Figure2. Scale Space Generated by Image Downsample (butterfly)

Ting Zhou
50169020

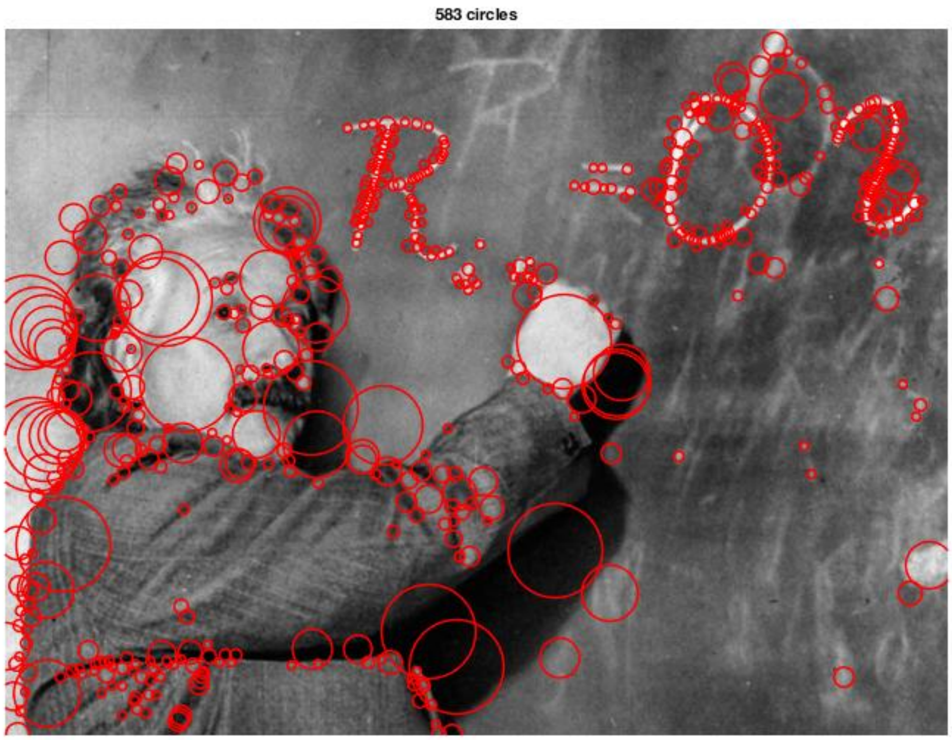


Figure3. Scale Space Generated by Increasing Filter Size (einstein)


Figure4. Scale Space Generated by Image Downsample (einstein)

Ting Zhou
50169020

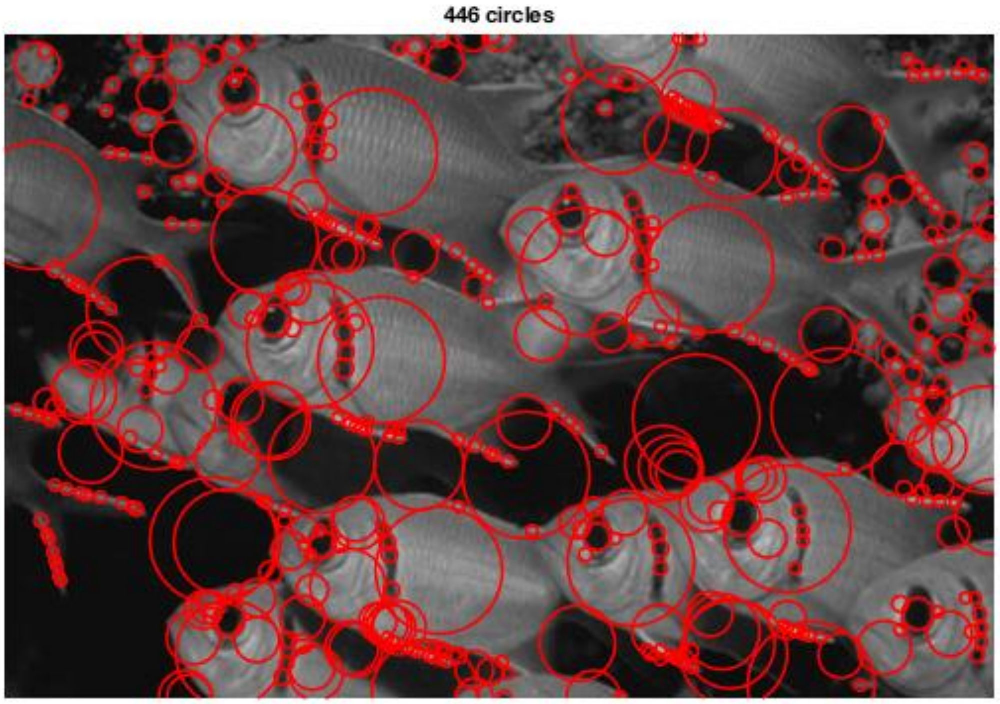


Figure5. Scale Space Generated by Increasing Filter Size (fishes)

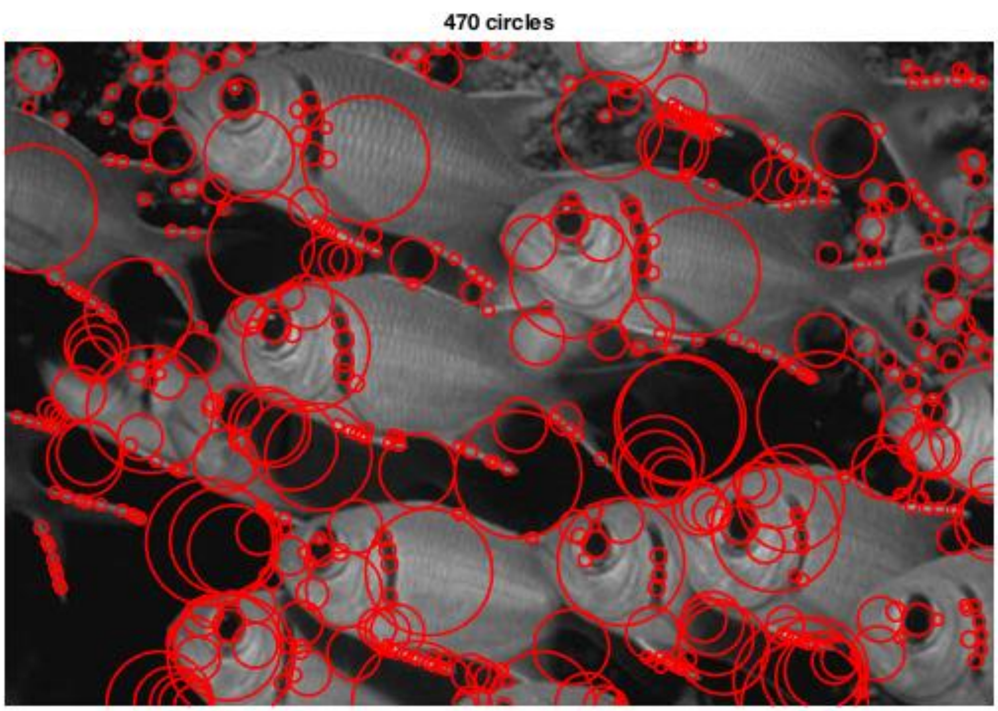


Figure6. Scale Space Generated by Image Downsample (fishes)

Ting Zhou
50169020

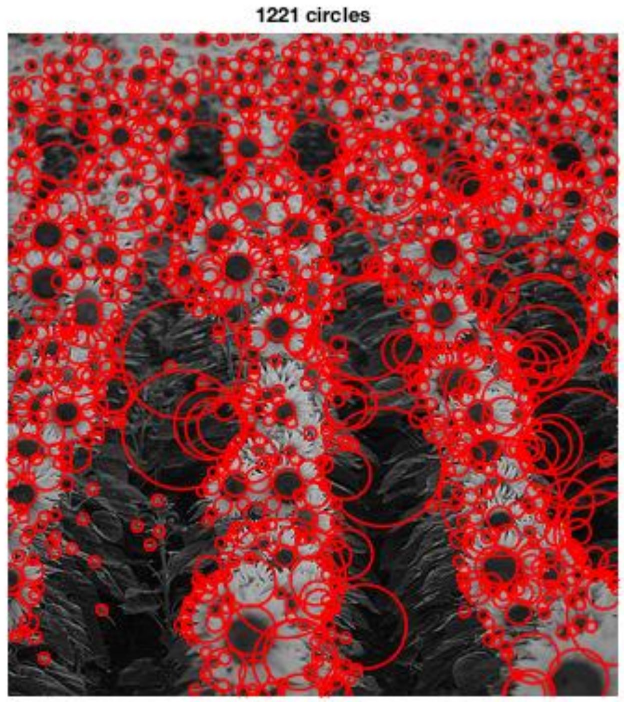


Figure7. Scale Space Generated by Increasing Filter Size (sunflowers)

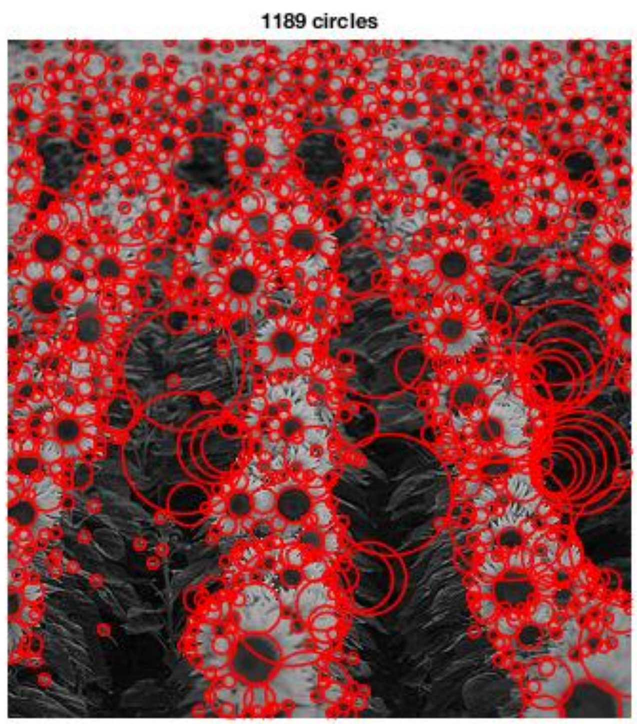


Figure8. Scale Space Generated by Image Downsample (sunflowers)

Ting Zhou
50169020

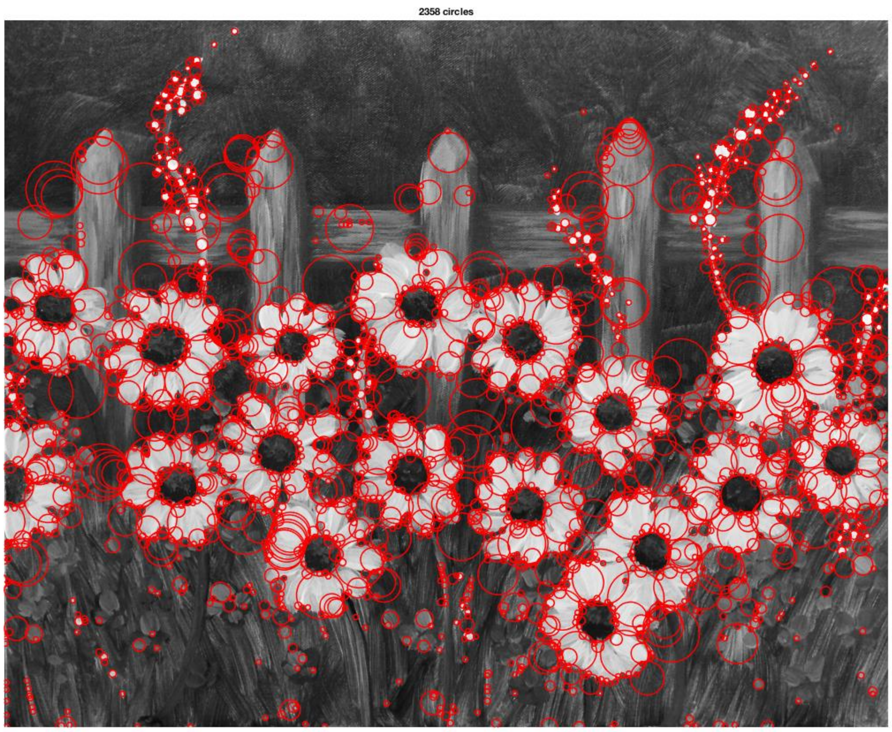


Figure9. Scale Space Generated by Increasing Filter Size (flowers)


Figure10. Scale Space Generated by Image Downsample (flowers)

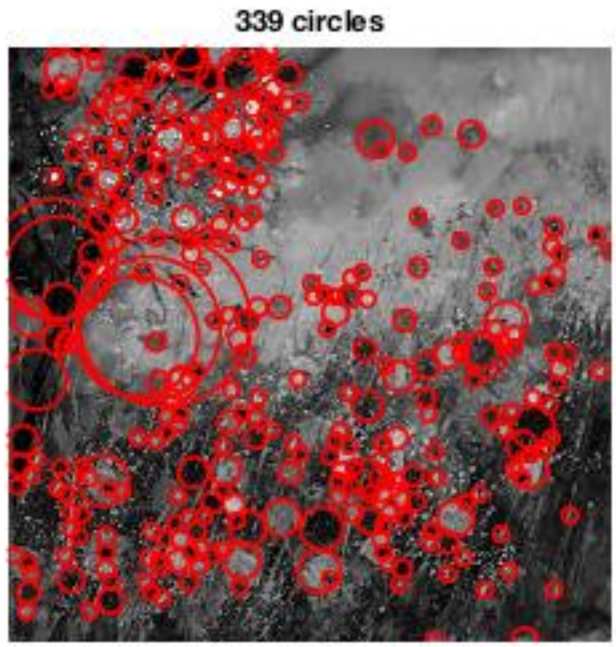


Figure11. Scale Space Generated by Increasing Filter Size (garden)

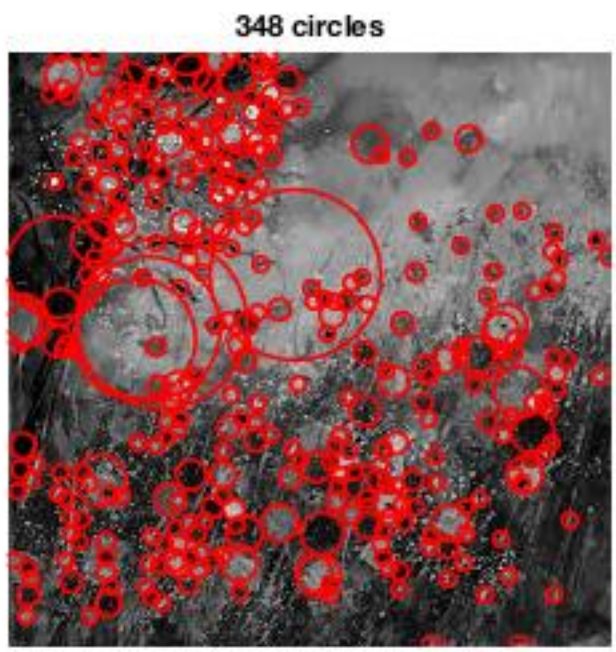


Figure12. Scale Space Generated by Image Downsample (garden)

Ting Zhou
50169020


Figure13. Scale Space Generated by Increasing Filter Size (corns)

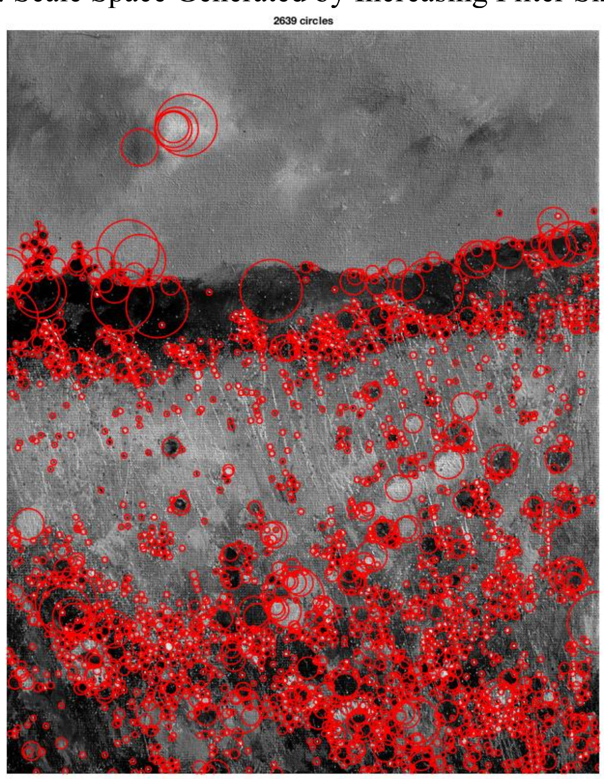


Figure14. Scale Space Generated by Image Downsample (corns)

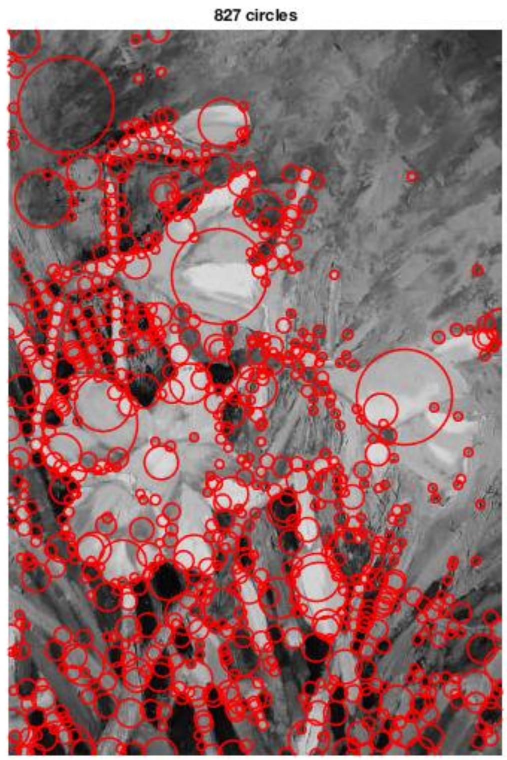


Figure15. Scale Space Generated by Increasing Filter Size (flora)

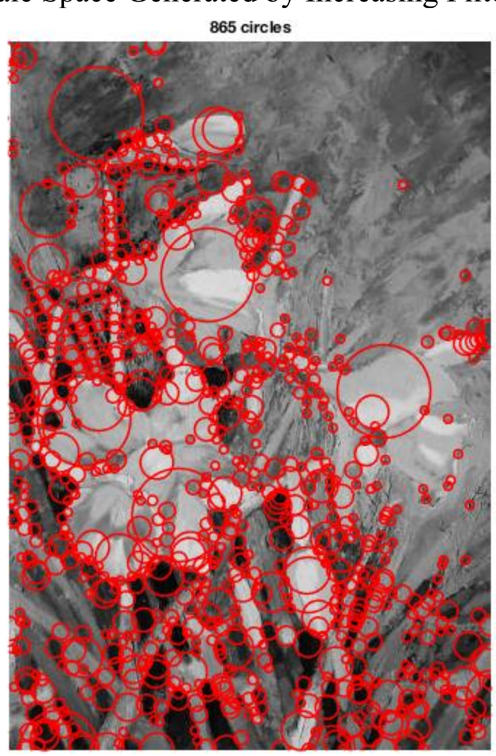


Figure16. Scale Space Generated by Image Downsample (flora)

Table1. Efficiency Comparison between Two Methods

| images | Filter Size Increase (sec) | Image Downsample (sec) |
|---|---|---|
| Butterfly | 1.901894 | 0.167626 |
| Einstein | 3.385510 | 0.916839 |
| Fishes | 2.034308 | 0.128886 |
| Sunflowers | 1.434900 | 0.108100 |
| Flowers | 8.511568 | 0.500033 |
| Garden | 1.214829 | 0.092617 |
| Corns | 5.138578 | 0.306491 |
| Flora | 1.921142 | 0.166048 |

**2. "Interesting" Implementation Choices**

- There are three interpolation methods for Matlab imresize() function which are 'nearest', 'bilinear' and 'bicubic'. I apply 'bicubic', because it's more efficient and works best. When I applied imresize() with 'nearest', almost all of the pixels are assigned as blobs, because the output pixel is assigned the value of the pixel that the point falls within. No other pixels are considered. The results by using 'bilinear' is okay, but offering more invalid blobs than applying 'bicubic'.
- To perform non-maximum suppression, there are three available functions to use which are nlfilter, colfilt and ordfilt2. Basically, the three functions can do the same thing that is to assign each pixel to the maxima of a specific block size. I choose to use ordfit2, because it's easier to apply than the others and faster.
- I use abs() function to the scale space of LoG filtered reponses, because when doing non-maximum suppression the negative values will all be eliminated. If without get the absolute value for the scale space, the blob detector is going to only detect the dark gray blobs and avoid those light gray blobs. Furthermore, the more invalid edges are going to be detected as blobs if we don't take the absolute value of the scale space.

**3. Determination of Parameters**

The required parameters in scale-space blob detection are:

- The number of scales
- Sigma, the standard deviation of Gaussian
- k, the scale factor
- filter size of LoG
- threshold of the maxima
- radius of the blobs

In order to keep the filter size as odd, I set filtersize = 2 * ceil(3 * sigma) + 1
The initial sigma is set to 2 and I use 15 levels.
After multiple trials, the scale_factor is set to sqrt(sqrt(2)).
The radius is depends on sigma and the level the blobs belong to,

radius = sqrt(2) * sigma * scale_factor^(i - 1)

where i is the level#.

Before determining the parameters, we need to know blobs are scale-invariant features. The Laplacian has a strong response not only at blobs, but also along the edges, which means some spots are not repeatable in different scale. Thus we need to set the parameters well in order to detect more real blobs and less edges-blobs.
The finally parameters in my code is,
num_scales = 15; sigma = 2; scale_factor = sqrt(sqrt(2)); threshold = 0.1
Actually, the parameters should be adjusted with respect to different images.

**4. Discussion of Results and Future Improvement**

Just like the statement mentioned in the previous section, the Laplacian has a strong response not only at blobs, but also along the edges, which means some spots are not repeatable in different scale. Thus we need to eliminate the edges incorrectly detected as blobs by computing Harris response at each detected Laplacian region. Also, for better performance, the difference-of-Gaussian pyramid should be implemented.

References:

David G Lowe. Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision, 60(2):91–110, 2004.