ΜΗΧΑΝΙΚΗ ΟΡΑΣΗ

Assignment 2: Blob Detection

Lab Team 23:
Apostolopoulos Athanasios -Iakovos
Perakis Georgios
Tzortzi Maria-Eleni

In this Assignment, we implemented a part of the Scale invariant Feature Transform algorithm proposed by David G. Lowe. We filtered seven different input images in order to locate the key points of each image.

Difference of Gaussians Pyramid

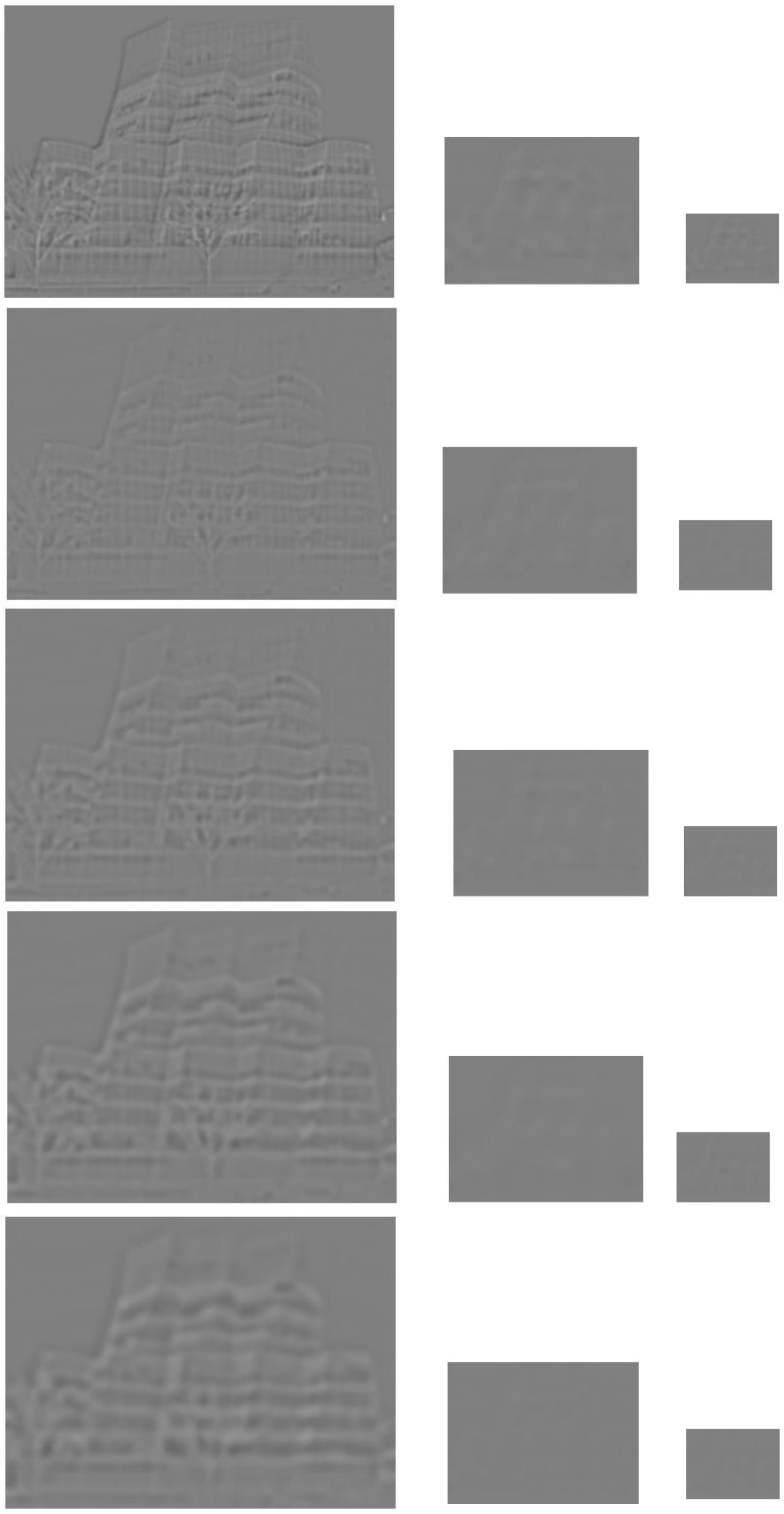
First, we filtered the images in order to create the Laplacian of Gaussians Pyramid. Our initial scale space consisted of three octaves with six images in each octave. To create the scale space two different approaches were followed.

In the first approach the original image was filtered using a Gaussian filter with increasing standard deviation. More specifically, the initial standard deviation (sigma) for the first octave was 1.8 which was then incremented by a factor of k=1.41 (square root of two) for every following image. Instead of down sampling the last image of each octave to create the next we squared the sigma of the gaussian filter. This approach gave the same results, as we “up sampled” the filter instead of down sampling the image, and did not have to up sample it later in order to find the key points.

In the second approach we filtered the initial image with a gaussian image once. Then the resulting image was down sampled and up sampled by a growing factor of k. In each octave the up sampling and down sampling factor was increased in order to simulate the distortion caused by a gaussian filter. During this procedure bilinear interpolation was used in order to reconstruct the original sized image form the down sampled one.

Both methods described above were timed using the tic and toc commands. The method involving Gaussian Filtering timed from 0.16-0.17 seconds whereas the up sampling/down sampling method timed at 0.12-13 seconds(Calculation time without image printing). The main reason for the timing difference is that the first method involves many convolution calculations involving big Kernels. Due to the increasing of the standard deviation of the Gaussian Filter, its size had to be big in order to contain the “important” information of the Gaussian bell.

The Difference of Gaussians Pyramid was created by subtracting every two subsequent levels of the scale space. The resulting pyramid consisted of 15 images in 3 levels. For the key point localization, we used the Difference of Gaussians Pyramid that was created using the first of the two methods described above, as we felt that this approach was closer to the proposed sift algorithm implementation. The results of the method one described above are shown in the next page.



Local Extrema Detection

In order to detect the local maxima and minima, the function Extrema Detector was created. In this function, each sample point is compared to its eight neighbours in the current image and eighteen neighbours in the adjacent images on its left and right in the octave. The coordinates of these potential key points are stored into a vector and later are reduced using a component of the Harris algorithm.

Local Extrema Thresholding

In order to reduce the extremas and keep the invariant to rotation ones, we created the function HarrisDetection, where we calculated the R matrix of the image as described in the Harris algorithm. Here we have to note that the part of the function harrisDetector, which concerns the calculation of the R matrix, was found online and was tailored to fit our needs. The link for the video containing the original code is: https://www.youtube.com/watch?v=I_PkOXmzS8s&ab_channel=Closure

We then thresholded the key points based on their R values and after a bit of experimenting with the thresholds we found that the best threshold values for each image are the ones below:

Image	Threshold
sunflowers.jpg	0.00008
fishes.jpg	0.00000005
green_leaf_leaves.jpg	0.000005
fylla1.jpg	0.00007
building.jpg	0.00005
Japan_building.jpg	0.00005
Japan_building1.jpg	0.00007

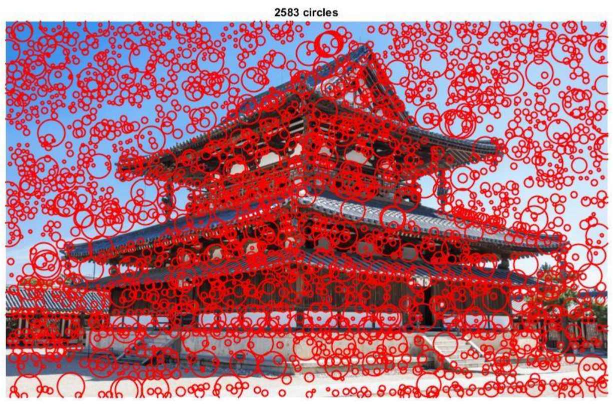


Figure 1: Image without Harris key point elimination

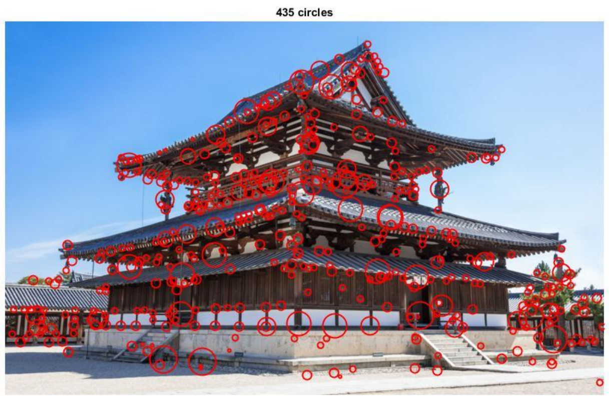


Figure 2: Image with Harris key point elimination

Final Results and Conclusions

To test our code, we used both the given pictures plus some more found on the internet. The final results of our algorithm for each picture are shown below. Our algorithm seems to produce satisfying results after many tweaks on the paramaeters of our code (k , sigma ,threshold). We have to note that our code does not compute the threshold for each picture. This threshold was tailored for each picture by trial and error.

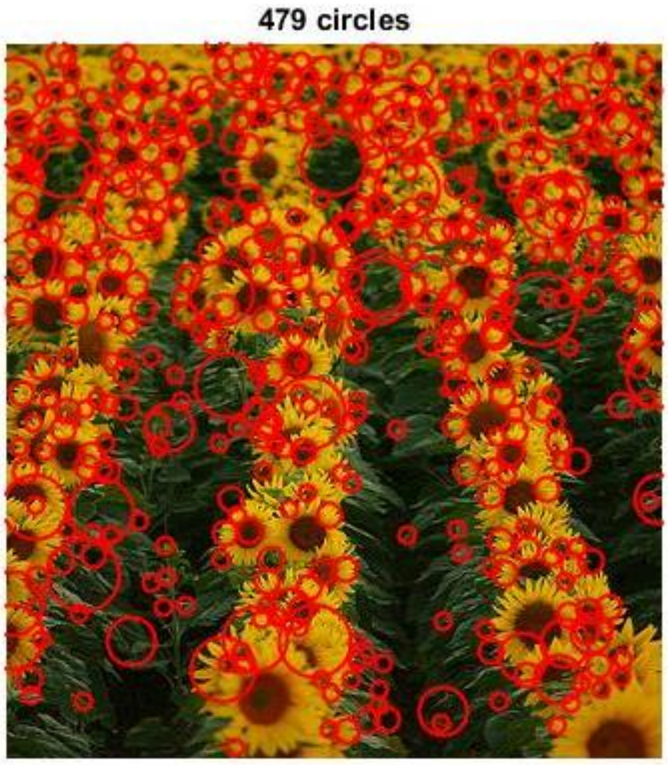


Figure 1: sunflowers.jpg Threshold=0.00008

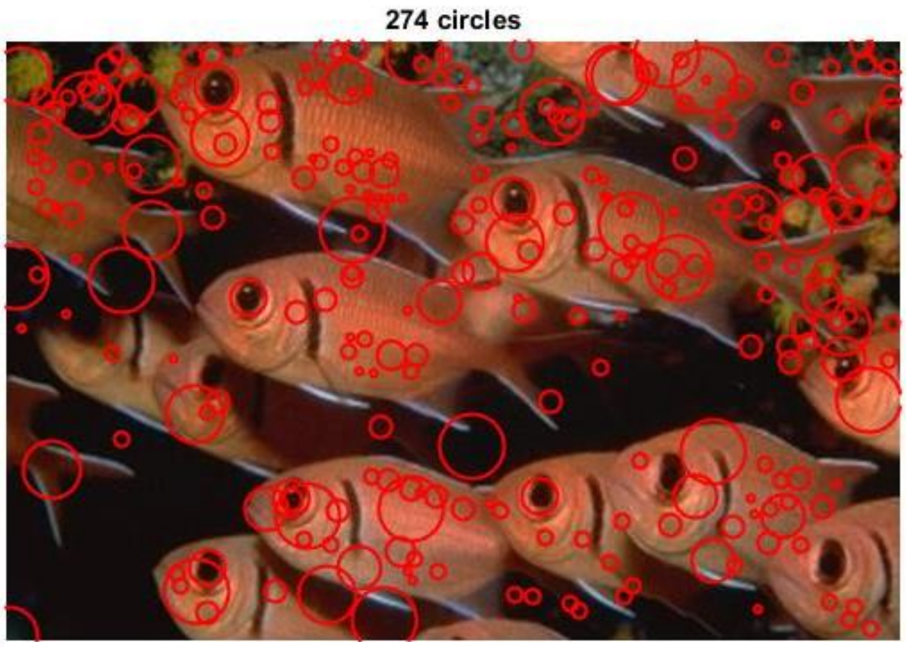


Figure 2: fishes.jpg Threshold=0.00000005

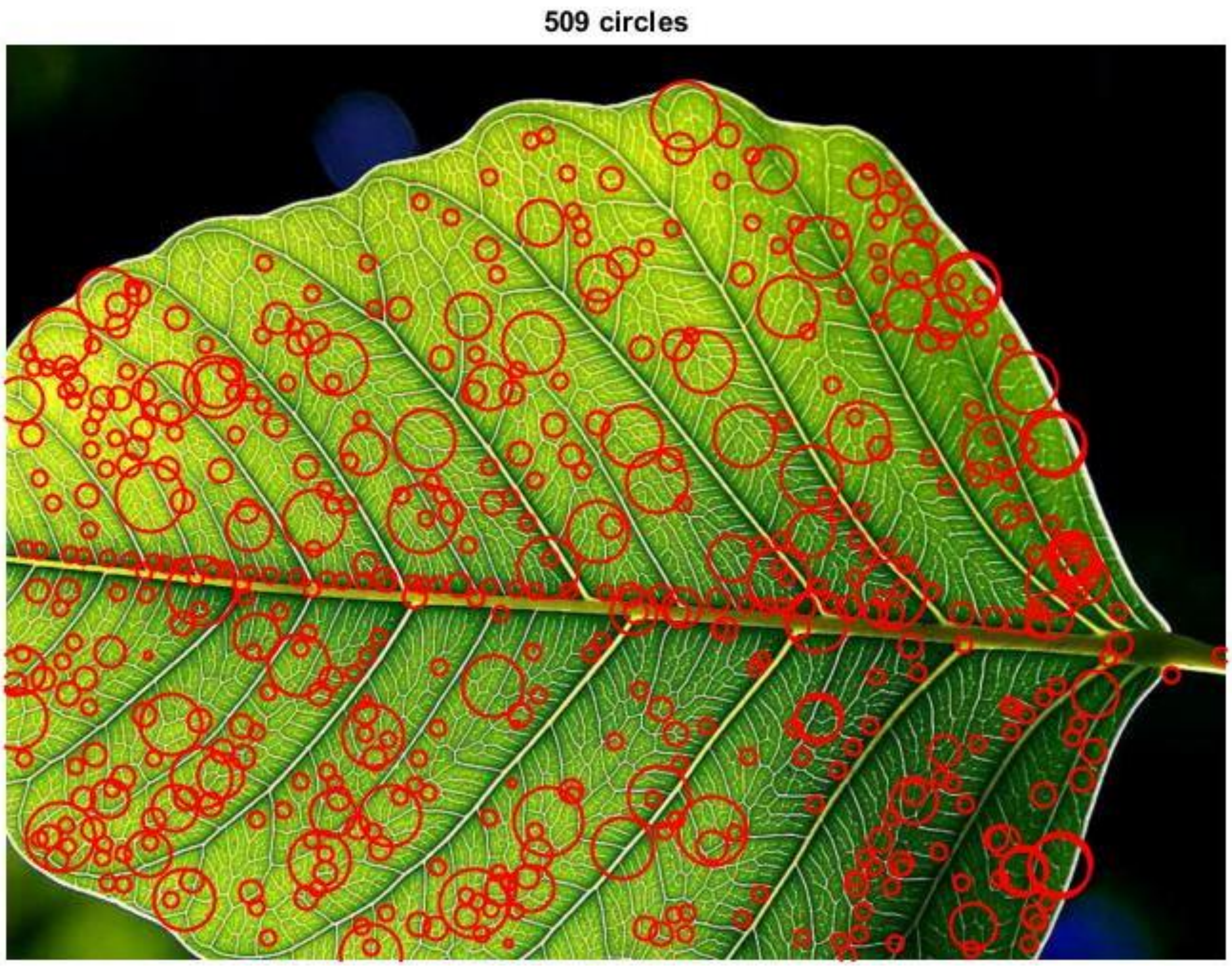


Figure 5: green_leaf_leaves.jpg Threshold=0.000005

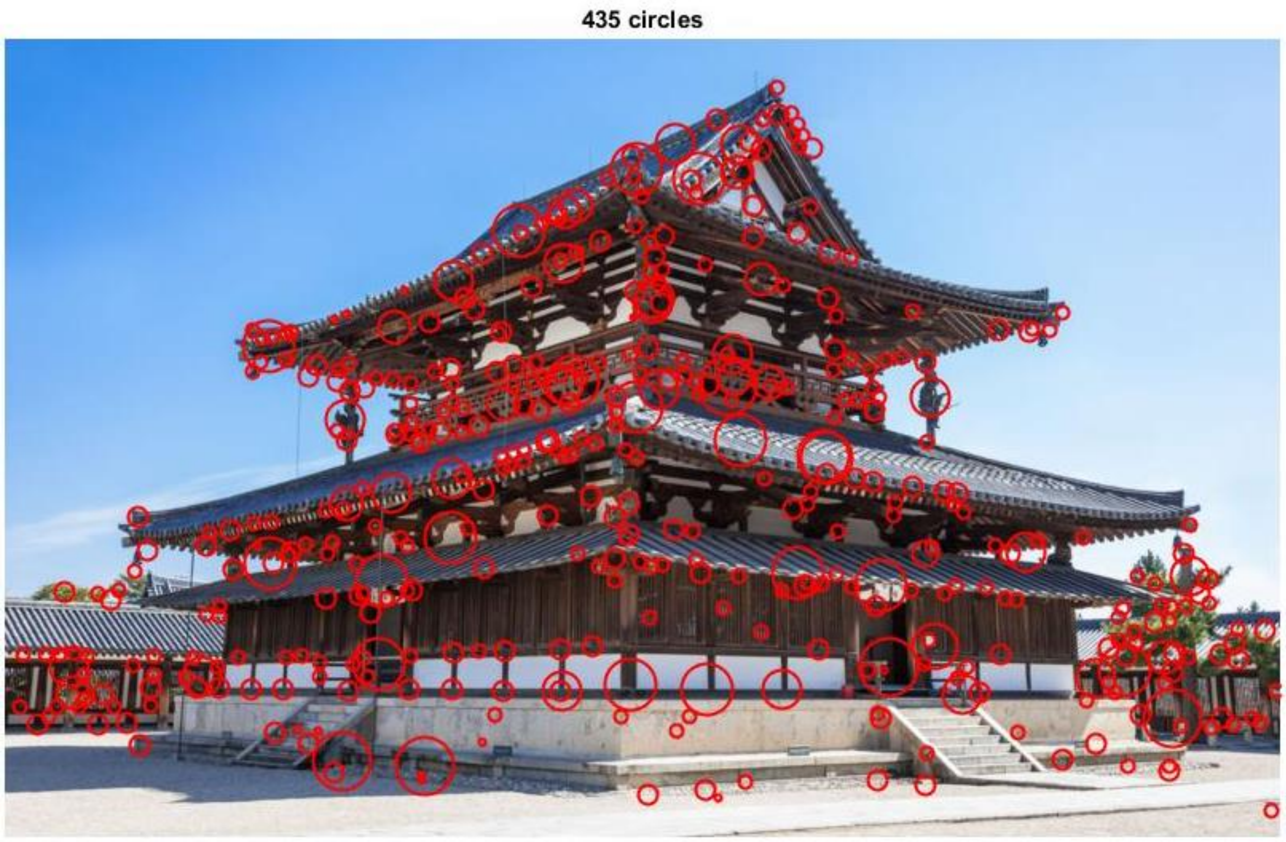


Figure 6: Japan_building.jpg   Threshold=0.00005

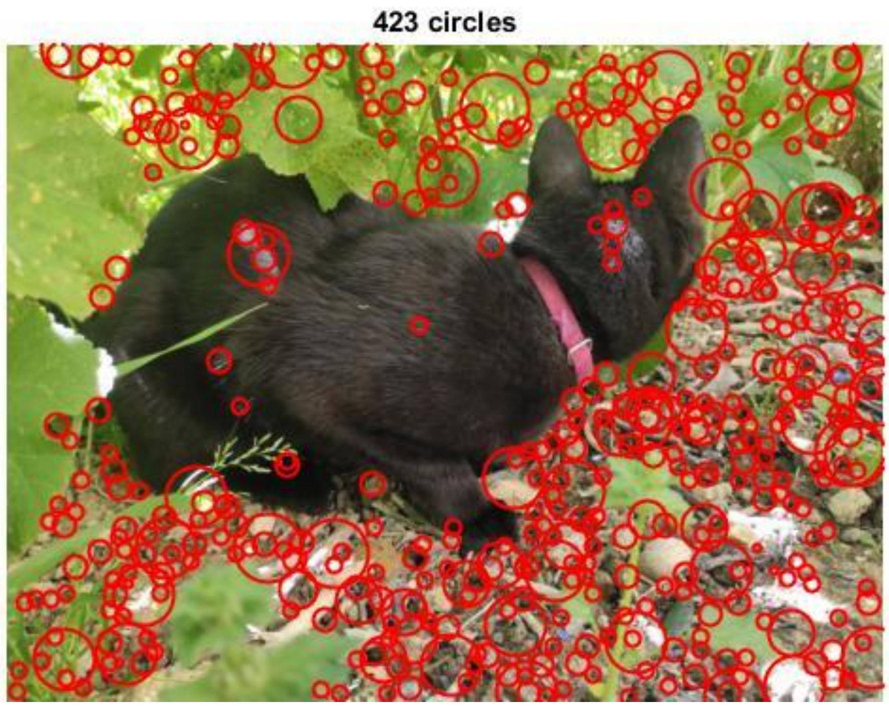


Figure 7: fylla1.jpg Threshold:0.00007

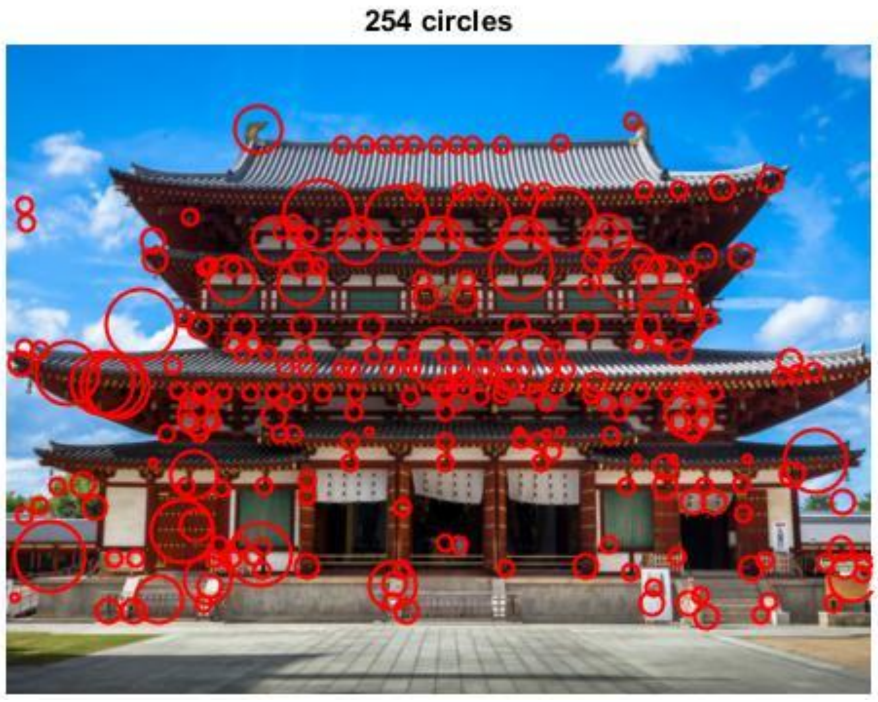


Figure 8: Japan_building1.jpg Threshold=0.00007

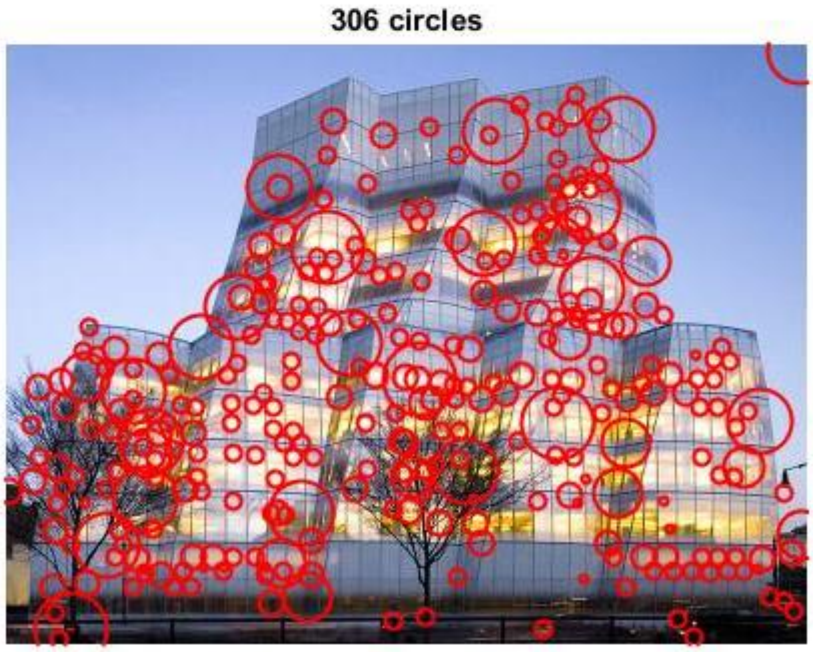


Figure 9: building.jpg   Threshold:0.00005