COMP 558 Assignment 2
Yige Feng 260803777

Question 1

gaussian, sigma=4

gaussian, sigma=8

gaussian, sigma=12

gaussian, sigma=16

heat equation, sigma=4

heat equation, sigma=8

heat equation, sigma=12

heat equation, sigma=16

Total diffusion time is given by 0.5*sigma^2. For question 1, I used Matlab standard function imgaussfilt to do smoothing with gaussian. Then I implemented a separate function for heat equation where I calculate the gradients and perform numerical update for n iterations. I repeated for sigma values 4,8,12,16. (all with fixed iteration of n = 200 times.) The generated images are as above, we can see that for the same sigma value, heat equation produces blurrer images and also, in the images generated by heat equation, the outlining edges in the bottom left fades to dark as sigma increases. However, the bottom left corner, began to turns white as sigma increases.

I also took the difference between the result obtained with heat equation and the result obtained by the gaussian. The numerical values are as below: we can see that as sigma increases, the difference is growing larger.

Lastly, I changed the number of iterations(n = 50,100,200,400). As we can see from below, as number of iterations increase, difference between results obtained by the two methods are getting smaller and smaller.

Based on the data I got, I think the discrepancies occur between results obtained by the two methods is because the number of iterations of the value update in the heat equation method is too small. As we increase the number of iterations, the difference will becomes smaller.

Question 2

```
>> Q1
Number of iterations:
    50

difference total for sigma 4 =
  814.3846

difference total for sigma 8 =
   1.2465e+03

difference total for sigma 12 =
   6.1293e+15

difference total for sigma 16 =
   7.3669e+32
```

```
>> Q1
Number of iterations:
   100

difference total for sigma 4 =
  811.7403

difference total for sigma 8 =
   1.2348e+03

difference total for sigma 12 =
   1.8678e+03

difference total for sigma 16 =
   1.7321e+21
```

```
>> Q1
Number of iterations:
   200

difference total for sigma 4 =
  810.7769

difference total for sigma 8 =
   1.2292e+03

difference total for sigma 12 =
   1.8601e+03

difference total for sigma 16 =
   3.0021e+03
```

```
>> Q1
Number of iterations:
   400

difference total for sigma 4 =
  810.3787

difference total for sigma 8 =
   1.2265e+03

difference total for sigma 12 =
   1.8563e+03

difference total for sigma 16 =
   3.0006e+03
```

Part 1:
Below are result for the gaussian pyramid. I use imgaussfilt to perform filtering and then resize the image.

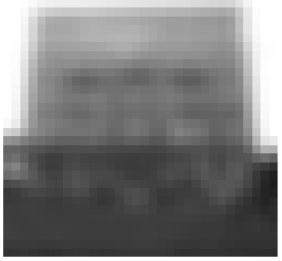
gaussian, level 0, sigma = 1

gaussian, level 1, sigma = 2

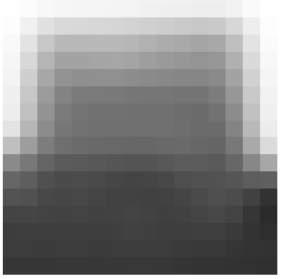
gaussian, level 2, sigma = 4

gaussian, level 3, sigma = 8

gaussian, level 4, sigma = 16

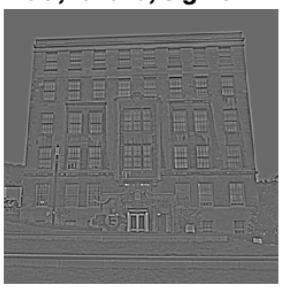


gaussian, level 5, sigma = 32



Part 2:
Below are results for the laplacian pyramid. I upsampled the successive level to the same size as the current level and then calculated the difference.

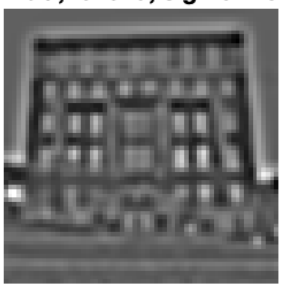
DoG, level 0, sigma = 1



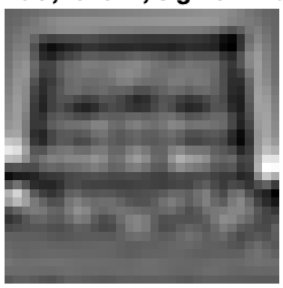
DoG, level 1, sigma = 2

DoG, level 2, sigma = 4

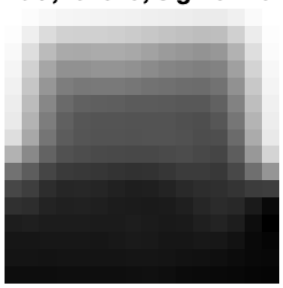
DoG, level 3, sigma = 8



DoG, level 4, sigma = 16



DoG, level 5, sigma = 32

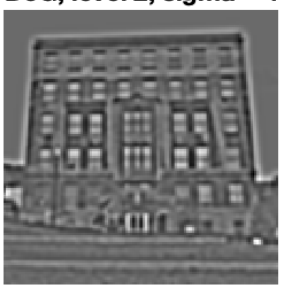


Part 3:
Comparing the images formed by convolving with difference of gaussians and convolving with laplacian of gaussians directly, we can see that difference of gaussians produced slightly darker images but apart from that there is no other differences.

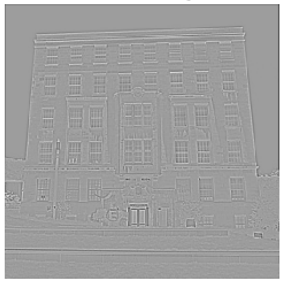
DoG, level 0, sigma = 1

DoG, level 1, sigma = 2

DoG, level 2, sigma = 4



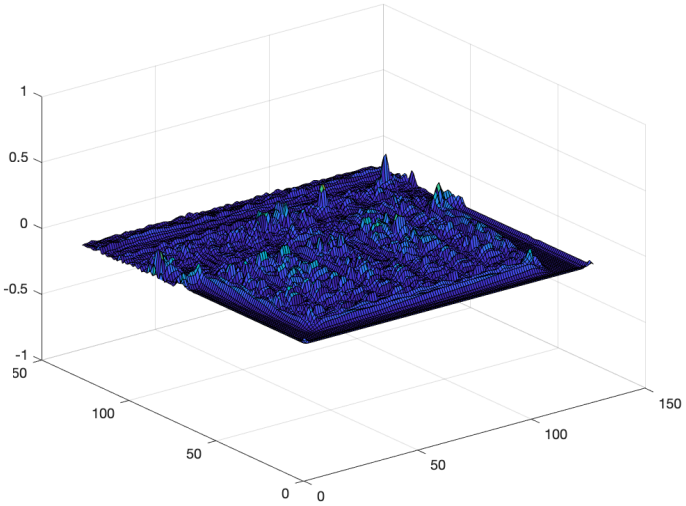
LoG, level 0, sigma = 1

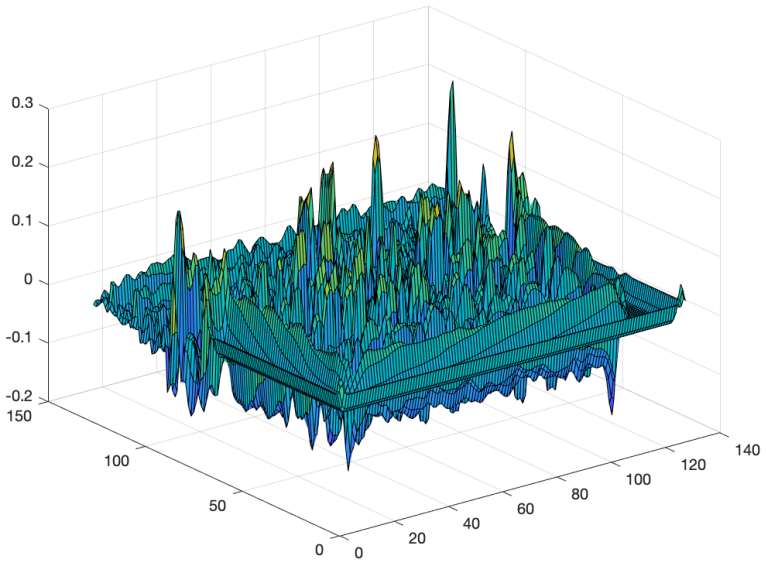


LoG, level 1, sigma = 2

LoG, level 2, sigma = 4

And from the visualization graph formed by taking difference and the absolute difference between DOG and LOG, we can see that there is barely any difference between the two(approaches zero).





Quantitative explanation:

The relationship between laplacian of gaussian, $\sigma^2\nabla^2 G$ and difference of gaussian D, can be derived from the heat diffusion equation:

$$\frac{\partial G}{\partial \sigma} = \sigma\nabla^2 G.$$

Thus, $\sigma\nabla^2 G = \frac{\partial G}{\partial \sigma} \approx \frac{G(x, y, k\sigma) - G(x, y, \sigma)}{k\sigma - \sigma}$

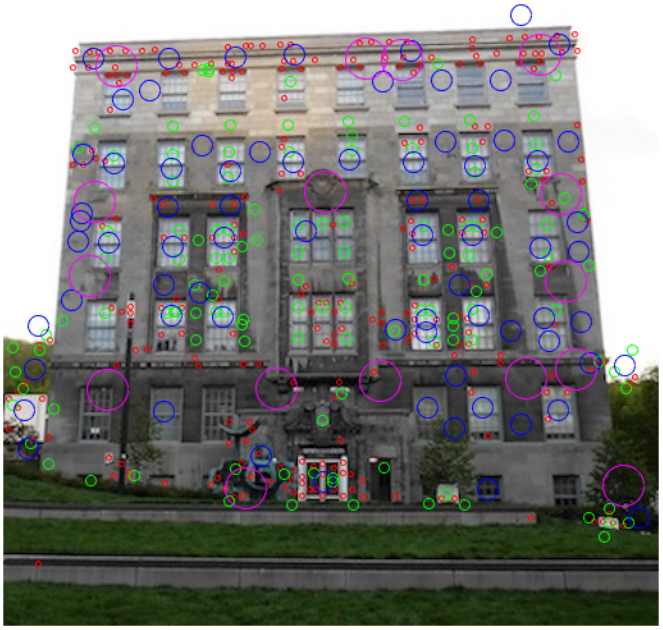
$$\Rightarrow G(x, y, k\sigma) - G(x, y, \sigma) \approx (k-1)\sigma^2\nabla^2 G$$

In this question, we use $k=2$

$$\Rightarrow G(x, y, 2\sigma) - G(x, y, \sigma) \approx \sigma^2\nabla^2 G.$$

Part 4:
For this part I first save all the levels of DoG into a 3d matrix. Then I iterate through the inner levels. In each level, I iterate through each pixel and save all the points in the 3*3 neighborhood above and below into an array(26 in total for each pixel). Then I choose the point if (1)it is not equal to zero; and (2)either it is larger than the max of the array, exceeds the mean of array by a certain threshold, or it is smaller than the min of the array, below the mean of array by a certain threshold. And finally, I overlay those points on the original image in different colors and radii based on the sigma value associated with the level it comes from.

I also rotate the image by 20 degrees and resize it to 1024*1024. And run the algorithm again. The results are approximately the same.



Question 3:
My idea in deciding the inliers in this question is that: firstly, I represent the line model with the equation R = x*cos(theta) + y*sin(theta). In each iteration, I randomly generated an index and use this index to reference into the pre-processed vectors and get the x,y value of the location and the gradient orientation theta. Then I calculate R based on these values. Then for every other edge point E*, I calculate the value of its R* and comparing it to R, if their absolute difference is smaller than a certain threshold, and the absolute difference between their gradient orientation is smaller than a certain threshold, then I choose this point. And in the end of each iteration, if the number of inliers is larger than every previous model, then I update the best model.

The xy-plot I generated is as below: