6CCS3COV – Computer Vision, Coursework 1

Wonjoon Seol, Computer Science, 1600465

November 4, 2018

By submitting this coursework, I declare that I understand the nature of plagiarism as defined in the department handbook and that the content of this coursework submission is entirely my own work.

3. Smoothing Masks

3.1. Box masks

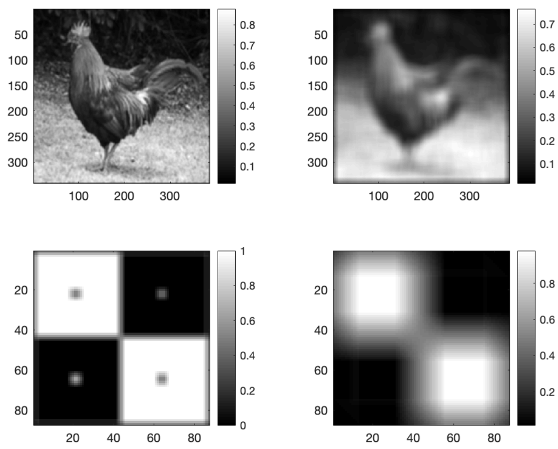

Figure 1: Box mask of size 5, 25 applied to 2 images

The first image is rooster convolved with box mask 5x5. The smoothing applied to the first image but not as much as box mask size 25x25. The box mask every pixel by the average of itself and its surrounding neighbours. This generates a blurred image.

The bigger the kernel is, the more its surrounding neighbours are. Hence more blurring occurs. Same applies to box image.

3.2. Gaussian mask

3.2.1. Gaussian mask in comparison with Box mask

This time the images are convolved with gaussian masks. The size of each mask is chosen with 99% confidence interval. The heuristic is to use $[\mu - 3\sigma, \mu + 3\sigma]$ in the Gaussian distribution to cover most of its values.

In the case of s.d. 1.5: $1.5 \times 6 + 1 = 9$. The +1 part is to make the odd square kernel, to follow the convention of the axis of symmetry with its origin on the centre pixel of the kernel. Similarly. s.d. 10 was chosen with a size of 61.

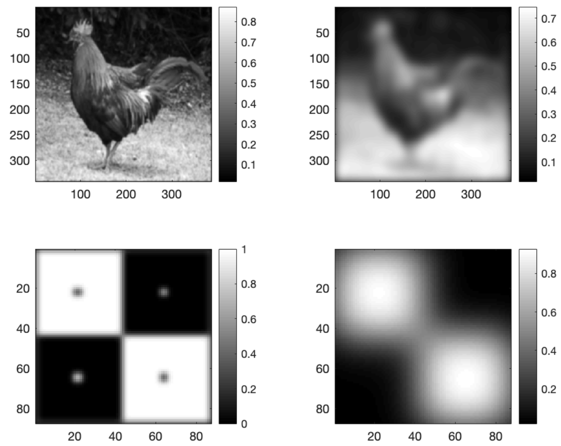

Figure 2: Gaussian mask s.d. 1.5 and 10

In the case of Gaussian filters, further away pixels get lower weights in contrast to box mask where it simply averages its neighbouring pixels by giving equal weights. The Gaussian filter is better at separating bands of frequencies as can be seen from the result. Gaussian is a low-pass filter which preserved more details than the box mask and the sharp cutoff of the box mask generates a noisier output.

3.2.2. Separable 2D kernel

1D: Elapsed time is 0.186675 seconds. 2D: Elapsed time is 0.238582 seconds.

2D-Gaussian convolution takes more time because it is more computationally expensive. 2D-Gaussian requires computation of $m \times n$ multiplications whereas in separable kernels the calculations are reduced down to $m + n$.

The result can be checked to be equal using the following formula:

```
abs(norm(result1D - result2D)) < 1e-12
```

Due to the double precision error, we compare whether the difference between the Euclidean norm of two vectors is smaller than epsilon, in this case, $1e - 12$. The result running the code returns true. So they are equal.

Therefore, for separable convolution 2D, it is better to separate it into two kernels to reduce its required computation.

4. Difference Masks

4.1. Difference Masks (1-D)

[-1, 1] is a first derivative mask that finds difference between next pixel value and current pixel value. In the case of $y = sin(x)$ where $x = 0, 0.1, 0.2, \cdots, 2\pi$. So y is simply $[sin(0), sin(0.1), \cdots, 2sin(2\pi)]$.

So we apply 1-D convolution using mask of size 2:

$$y[n] = \sum_{k=n-1}^{n} x[k] \cdot h[n-k]$$

While applying convolution, the mask is effectively rotated so it becomes [1, -1], this gives negative gradient of $sin(x)$. So the result is $-cos(x)$.

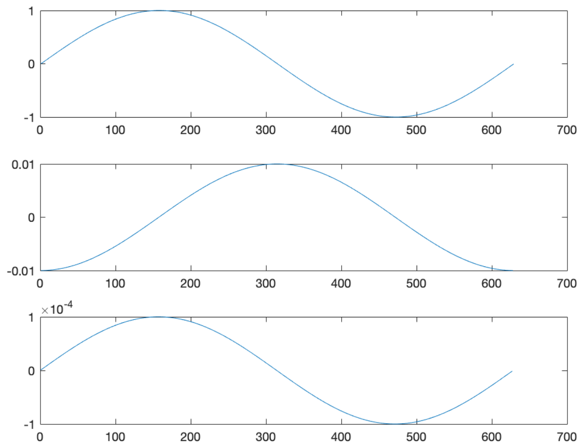

Figure 3: Difference Mask 1d

Similar logic to above, [-1, 2, -1] mask find negative second order derivative function of $sin(x)$. So it becomes $sin(x)$, the same as the original graph.

4.2. Difference Masks (2-D)

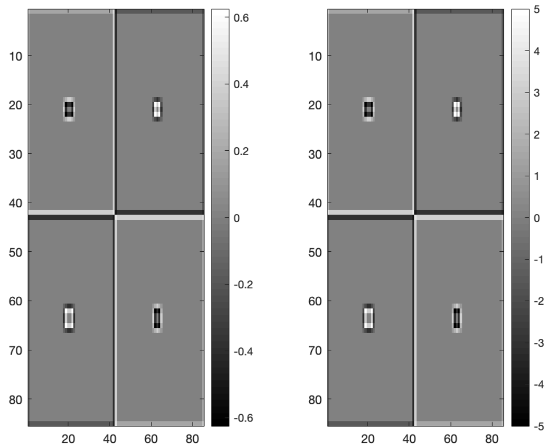

Figure 4: Difference Mask 2d

The Laplacian mask is good at detecting intensity discontinuities in all orientation. Both are positive Laplacian operator, which detects an outer edge of an image.

The image appears to be the same as the proportion between the pixels in kernel remain the same. However, as can be seen from the colour bar, it has a different scale.

4.3. Other Difference masks

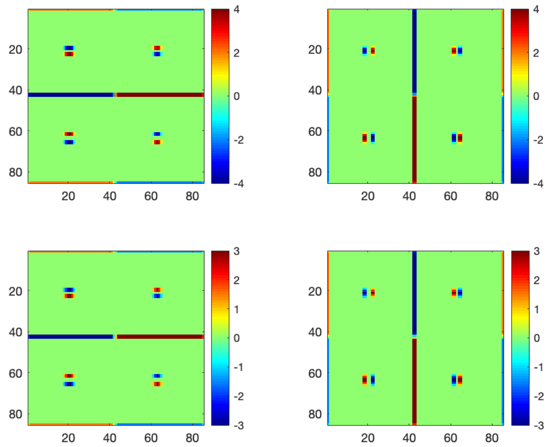

Figure 5: 1. Sobel, 2. Sobel', 3. Prewitt, 4. Prewitt'

5. Edge Detection

5.1. Gaussian derivative masks

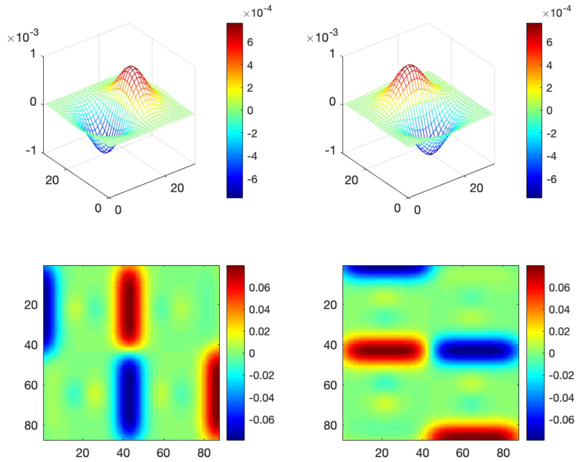

Figure 6: Horizontal mask and vertical mask

The value at high intensity discontinues are around 0.06 and -0.06. The first image, horizontal mask detects horizontal intensity changes. Whereas the vertical mask is more sensitive to vertical intensity changes. The box has significant intensity differences when changing colour from white to black and vice versa. The centre dots are also detected ($|0 - 0.02|$) but it is a lot weaker than black-white change.

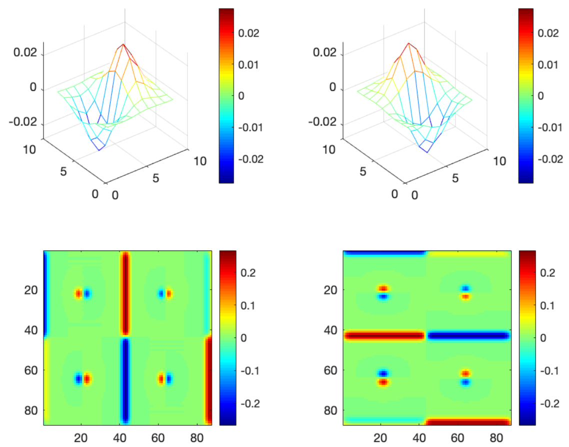

Figure 7: Gaussian derivative masks s.d. 1.5

This time, a smaller standard deviation mask is used. Since the mask is smaller and fewer pixels are involved in the smoothing process, there are sharp intensity differences along the edges. The values are around 0.2 and -0.2, a lot higher than standard deviation size 5.

This time, because the smaller standard deviation is used, it detected finer features - centre dots in the middle.

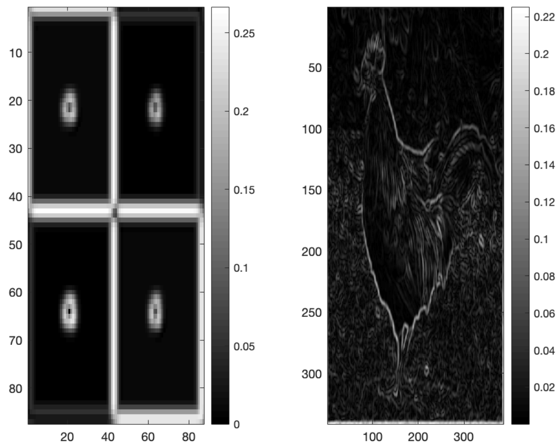

Figure 8: L2-norm generated

5.2. Laplacian of Gaussian (LoG) mask

For the first image, the edge values are between 0.05 and 0.1, also between -0.05 and -0.1. The values near the edge are somewhat close to 0 for the centre dots.

The second image with higher s.d. Intensity at the edge is at 6 and -4 ($\times 10^{-3}$). Near the edge, the values are the same. The edge at the centre dot is much weaker (0-2) than the edges between 4 rectangles.

Similar to the last result, smaller the standard deviation finer features such as centre dots are detected. When it is larger, the centre dots have much lower value. So larger standard deviation detects larger scale edges.

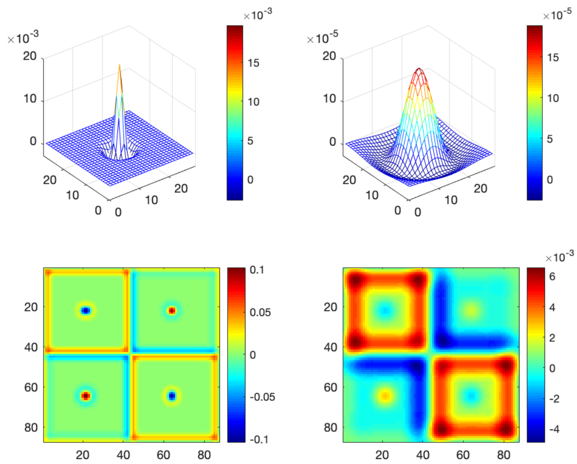

Figure 9: LoG mask

5.3. Difference of Gaussians (DoG) mask

```
lap1 = ones(3) / -8;
lap1(2,2) = 1;

gau6 = fspecial('gaussian', 31, 6);
gau3 = fspecial('gaussian', 31, 3);
dog = gau6-gau3;

gau5 = fspecial('gaussian', 31, 5);
log = conv2(gau5, lap1, 'sam');

log_sim = log./max(max(log));
dog_sim = dog./max(max(dog));
answer = sqrt(sum(sum((dog_sim-log_sim).^2)));

min = answer;
start_sd = 3;
end_sd = 6;
for i = 3:0.1:5.9
    for j = i+0.1:0.1:6
        dog = fspecial('gaussian', 31, i) - fspecial('gaussian', 31, j);
        dog_sim = dog./max(max(dog));
        temp = sqrt(sum(sum((dog_sim-log_sim).^2)));
        if (temp < min)
            min = temp;
            start_sd = i;
            end_sd = j;
        end
    end
end
disp("Similarity: " + min)
disp(start_sd + ", " + end_sd)
```

standard deviation : 4.9, 5.1, Similarity: 0.47865

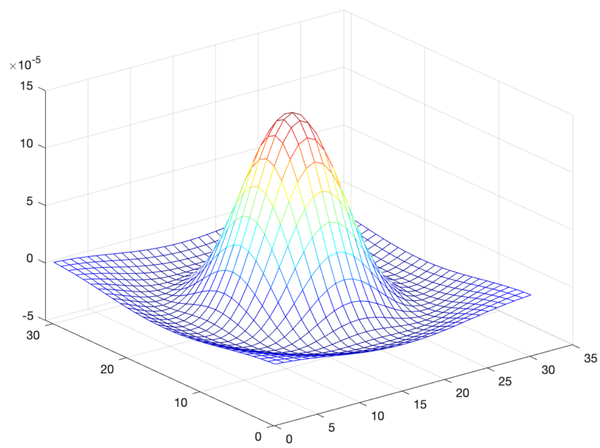

Figure 10: Mesh plot

6. Multi-Scale Representations

6.1. Gaussian Image Pyramid

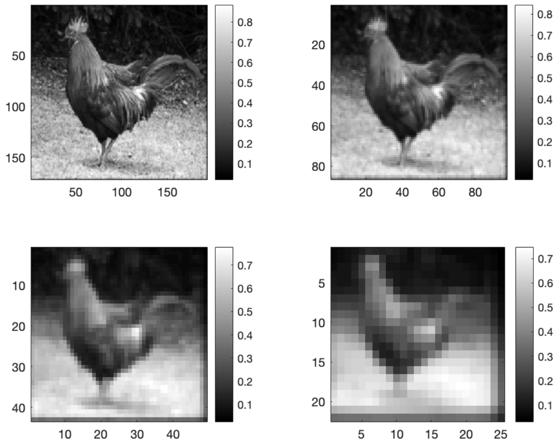

Figure 11: Gaussian image pyramid

6.2. Laplacian Image Pyramid

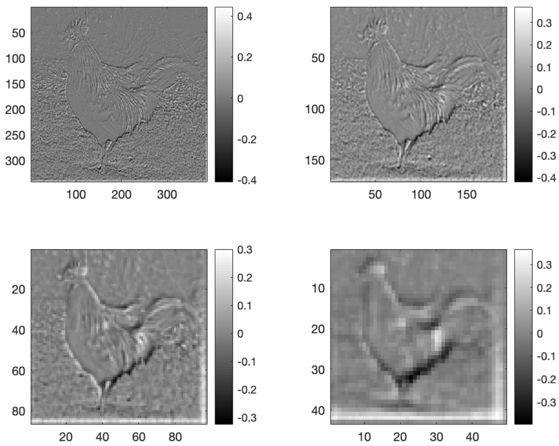

Figure 12: Laplacian image pyramid

A Laplacian image pyramid shows intensity discontinuities at multiple scales. Therefore, Laplacian pyramid can generate levels of band-pass filtered images, because they are a difference of two layers of the Gaussian pyramid, of which are successively sampled with more substantial standard deviation.

```
Ia = imread('rooster.jpg');

g = fspecial('gaussian', 31, 1);
Ia1 = imresize(conv2(Ia,g,'same'), 0.5, 'nearest');
Ia2 = imresize(conv2(Ia1,g,'same'), 0.5, 'nearest');
Ia3 = imresize(conv2(Ia2,g,'same'), 0.5, 'nearest');
Ia4 = imresize(conv2(Ia3,g,'same'), 0.5, 'nearest');
```

```
l1 = Ia - imresize(Ia1, size(Ia));
l2 = Ia1 - imresize(Ia2, size(Ia1));
l3 = Ia2 - imresize(Ia3, size(Ia2));
l4 = Ia3 - imresize(Ia4, size(Ia3));

figure(2);
subplot(2,2,1), imagesc(l1); colorbar
subplot(2,2,2), imagesc(l2); colorbar
subplot(2,2,3), imagesc(l3); colorbar
subplot(2,2,4), imagesc(l4); colorbar
colormap('gray');
```

7. Edge Detection Challenge

```
image = imread('firefighter.jpg');
lower = 0.04347;
higher = 0.34285;
threshold = [lower, higher];
sigma = 0.76304;
edge(image, 'canny', threshold, sigma)
```

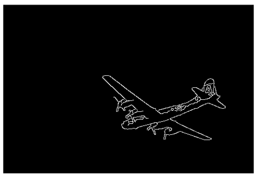

Figure 13: Plane

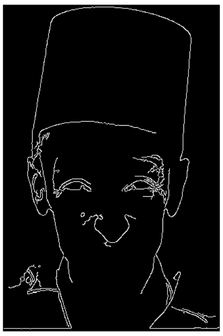

Figure 14: Human

Figure 15: Fire Fighter

This command works well for the plane image, but it also included a logo on its tail. It performs somewhat satisfactory for the human. The person has many wrinkles and a large nose. It was difficult to remove these lines without affecting other edges. The jaw has smaller intensity changes in comparison to other edges and is not detected as an edge.

Finally, the command detected too many edges on firefighter image. The background has many objects with its edges, and the command detected it too well due to intensity differences from these objects. The edge is not complete in the leg area, and it failed to detect shoe and hands.