Computer Vision CS 543 / ECE 549

Homework 1 Fangwen Wu (fwu11)

1. Lighting

A.
1. Right and above
2. The light captured by the camera had a specular reflection on the surface of that temple tip.
3. In some parts of the wood the albedo is lower compared to the other parts of the wood so less light get reflected so that part looks dark.
4. No. Because the surface is specular so you are not able to see the reflected light from the light source so the table will be dark.

B.
a. It is possible for observed intensities to change. If the camera moves to the area where the specular reflection of the light adds to the reflected lights from points 1, 2 and 3. The observed intensities will be larger than usual.

b. Intensity doesn’t depend on the viewing angle $\mathrm{I}(\mathrm{x}) = \rho N S \cos\theta$

$$\frac{I_1}{I_2} = \cos(180 - \theta_{12}) = \frac{5}{9} \qquad \theta_{12} = 123.8°$$

$$\frac{I_3}{I_2} = cos(180 - \theta_{23}) = \frac{8}{9} \qquad \theta_{23} = 152.7°$$

2. Image Pyramids

1)

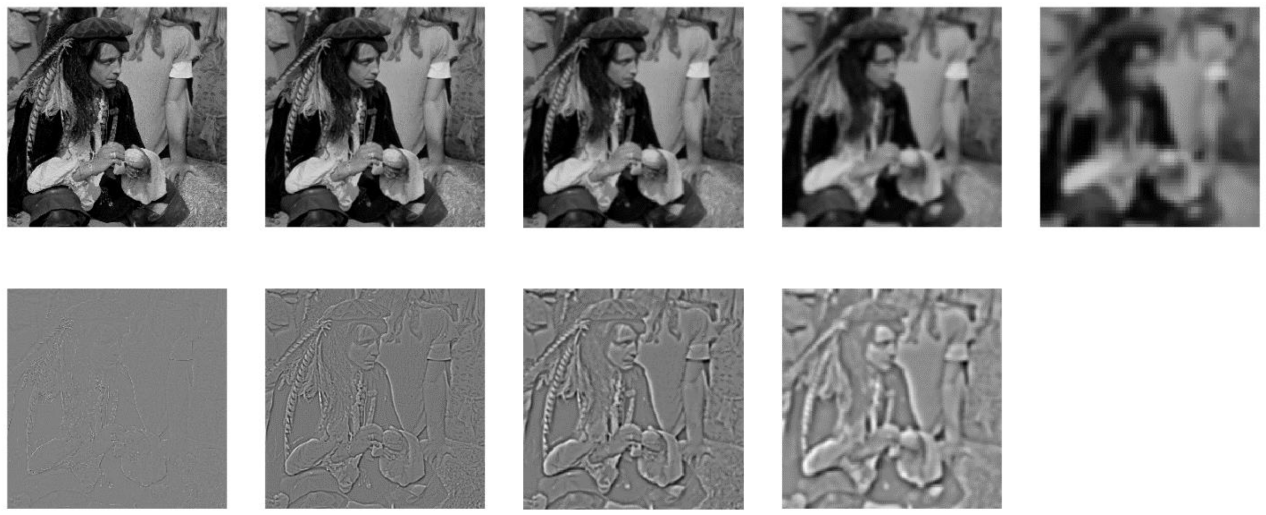


Figure 1 Gaussian pyramid (above) and Laplacian pyramid (below)

The 5th level of Gaussian pyramid and the 5th level of Laplacian pyramid are the same so here we only display the 5th level of Gaussian pyramid.

2)

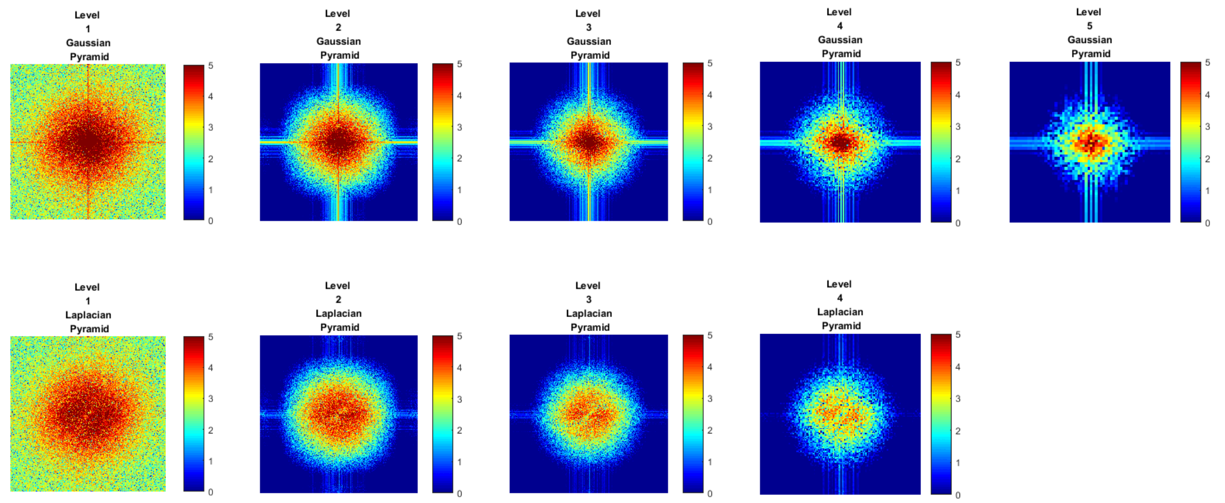


Figure 2 FFT plot of the Gaussian pyramid (above) and Laplacian pyramid (below)

The Gaussian pyramids constructs a sequence of progressively lower resolution images by doing a sequence of low pass filtering. Thus the high-frequency component of the image is removed. The Laplacian pyramids in contrast preserve the higher frequency component that is lost in the Gaussian pyramid.

The code for question 2: imagePyramids.m and plotFFT2.m

3. Edge Detection

Design choices:

Part a: Gaussian filter size of 13 by 13 and the sigma (standard deviation) is 2

Part b: Gaussian derivative filters with 8 orientations of size of 17 by 17 and the sigma is 2

-The filter size is determined by the variance of the Gaussian. Because the Gaussian function approaches 0 after 3 standard deviation from the mean so in order to better capture the shape of the Gaussian filter we choose the filter size to be sigma*(3 or 4)*2+1). Larger sigma detects large scale edges, small sigma detects fine features thus sigma = 2 as a reasonable number.

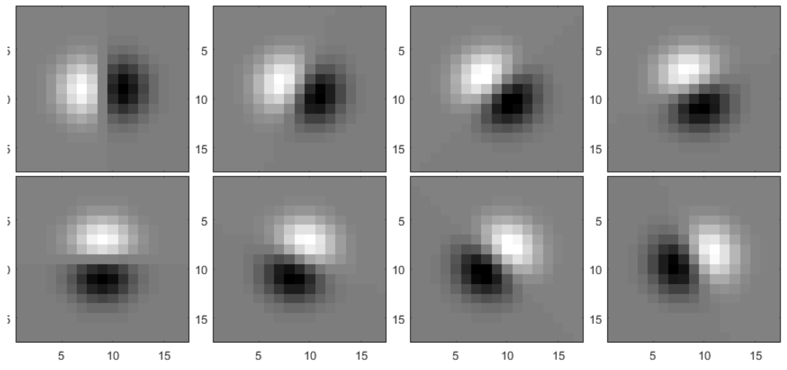
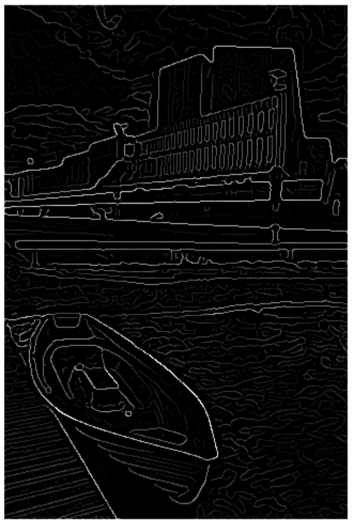
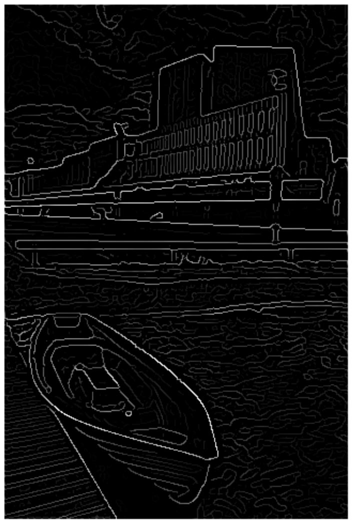
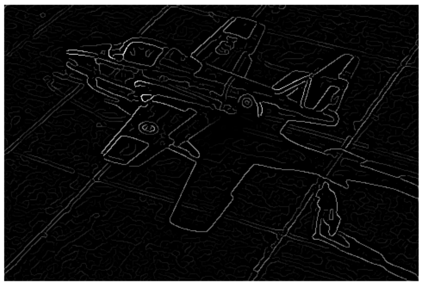
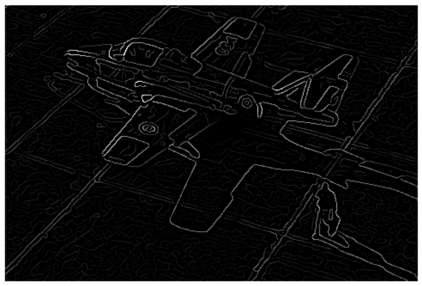


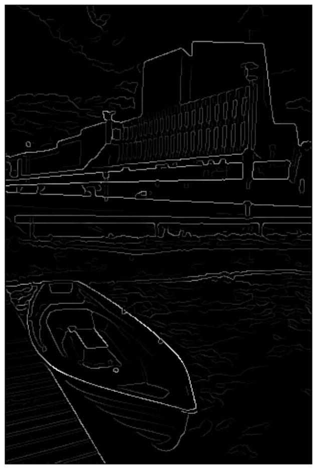
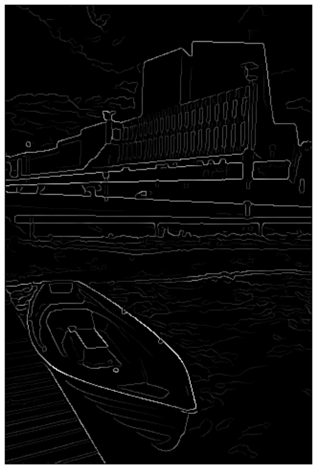
Figure 3 8-orientation Gaussian derivative filter with sigma = 2

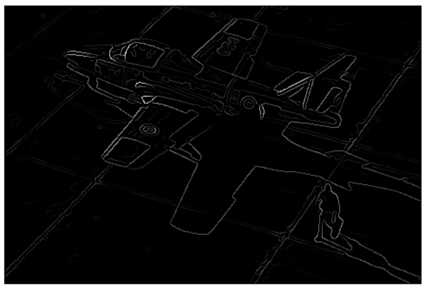
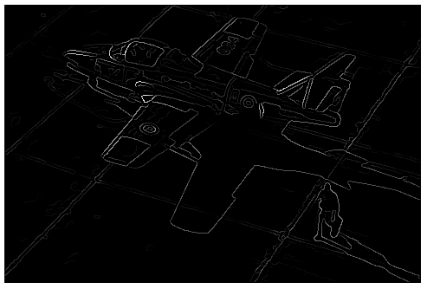
Qualitative results:
Use nonmax.m from left to right origin, gradient magnitude method, oriented filter method

Use canny edge detector
From left to right origin, gradient magnitude method, oriented filter method

Quantitative results:

Method(use nonmax.m)	Overall F-score	Average F-score
Gradient	0.597	0.638
Oriented	0.596	0.637

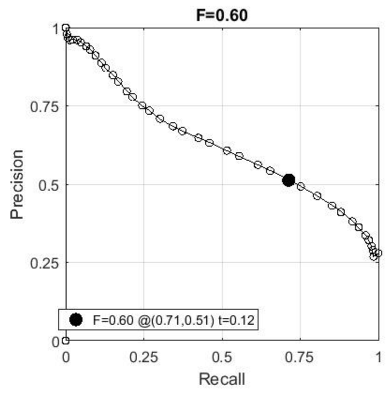


Figure 4 precision-recall plot for gradient method (nonmax)

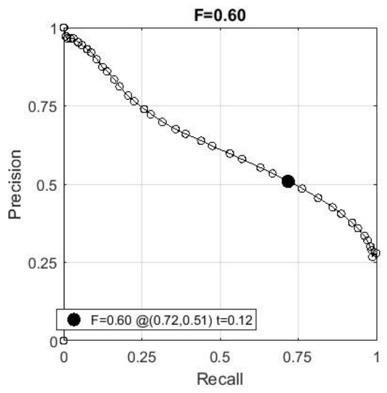


Figure 5 precision-recall plot for oriented filter method (nonmax)

Method(Canny for nonmax)	Overall F-score	Average F-score
Gradient	0.602	0.642
Oriented	0.600	0.640

The f-score from canny edge detector is higher because canny also performs hysteresis thresholding after the non-max suppression.

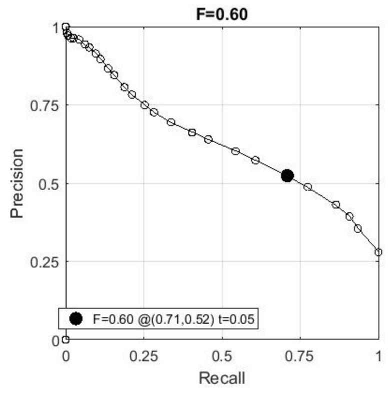


Figure 6 precision-recall plot for gradient method (canny)

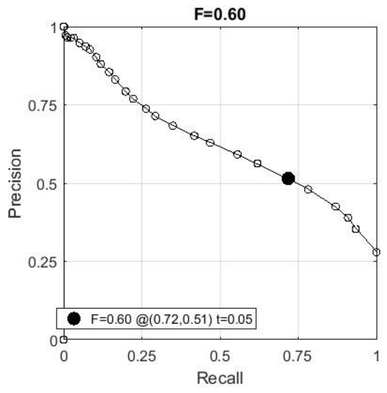
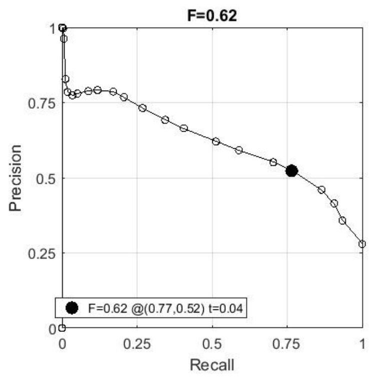


Figure 7 precision-recall plot for oriented filter method (canny)

<u>The code for question 3: gradientMagnitude.m, orientedFilterMagnitude.m, edgeGradient.m and edgeOrientedFilter.m</u>

Ideas for improvement:

1. Use a different color space other than RGB. For example the HSV color space. Convert the RGB image to HSV before any operations with im = rgb2hsv(im). Inspired by [1] The overall f-score reaches 0.621 and the average f-score reaches 0.647 with 8 orientation filter bank + canny edge detector.



2. The other possible improvement not passing grayscale image into canny edge detector. Instead, pass the red, green, blue channel into the function separately and then find the channel gives the max response in each individual pixels as the result. This is likely going to work because the camera sensor has 3 color channels and each of them captures the different wavelength of the light and hence has a different response. If you convert the image to grayscale, some of the information may lost because rgb to grayscale take the weighted average of 3 channel values. (e.g the edge of the object has strong response on the red channel but little response on the green and blue channel, and the grayscale just tells you the averaged response is not strong at all) This may affect the result of doing the edge detection.

3. One method mentioned in [2] is to use more image cues including the Boundary gradient, color gradient and texture gradient. For each cue we are able to obtain the boundary image just like we do in this lab. The last step involves combining these cues, not by simply adding them together, but combine these three cues on each individual pixel by multiplying each cue with a weight vector and compare with annotated pixel label (whether the pixel is in the boundary or not). The question is now transformed into a binary classification problem and it is possible to tune the weights vectors by minimizing the loss function of the logistic regression. After that we can predict new pictures and test how well they perform by plotting the precision-recall plot. This method is likely to improve our result because the edges are caused by a variety of factors, not only the intensity gradient. Hence, getting more image cues to analyze the edge can maximize the likelihood in the edge detection.

[1] Gwanggil Jeon, 2013 http://onlinepresent.org/proceedings/vol22_2013/21.pdf
[2]D. R. Martin, C. C. Fowlkes and J. Malik, "Learning to detect natural image boundaries using local brightness, color, and texture cues," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 5, pp. 530-549, May 2004.