CS270 Homework 2

Deadline 2021/4/30 0:00

Notes:

30 points for question 1, 40 points for question 2, 30 points for question 3, a total of 100 points. The example results provided are just for reference. Please try to achieve the best performance as you can. Discussions are encouraged and plagiarism is strictly prohibited, source code should not be shared in any form. Please submit your homework (report in .pdf format and your codes) to dip_2021@163.com with both subject and file named in this format. (CS270+ID+name+hw1) example, CS270_2019123321_张三_hw1.

Question 1 Image Compression and Watermarking

Tasks:

(1) Please implement the JPEG compression to compress "Lena" image using MATLAB or other platforms. You can use the library functions to perform some operations such as color transform or DCT. But the truncated Huffman coding part must be done by your own codes. Please use quantization table as shown in Fig. 1 and show your results of truncated Huffman codewords and compressed image. You are requested to print your codewords named "code.txt" and show the metrics including Compression ratio and Root mean square error between original image and compressed image for evaluation.

```
Y_Table=[ 16  11  10  16  24  40  51  61 ; ...
          12  12  14  19  26  58  60  55 ; ...
          14  13  16  24  40  57  69  56 ; ...
          14  17  22  29  51  87  80  62 ; ...
          18  22  37  56  68 109 103  77 ; ...
          24  35  55  64  81 104 113  92 ; ...
          49  64  78  87 103 121 120 101 ; ...
          72  92  95  98 112 100 103  99 ];
```

Fig 1: Quantization table

(2) Implement a digital watermarking program based on proper transform. Add 'LOGO_CS270.mat' as a watermark to "Lena" image and implement JPEG compression to this image. Then extract the 'LOGO_CS270.mat' from your compressed image to test the robustness. You are requested to show your watermarking extracted from the compressed image.

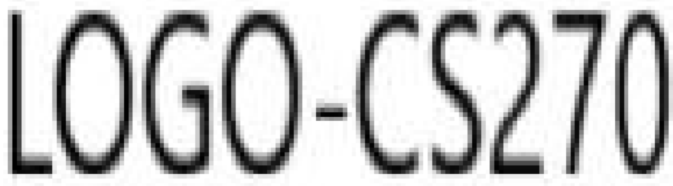


Fig 2: LOGO

Checkpoints

You need to answer these following questions in your report.

1. How do you implement your algorithm? Describe it by flow charts or words shortly. (2 pts)

You need to show these following figures in your report.

1. Results requested in Task (1). (14 pts)
2. Results of watermarking extracted from the compressed image. (14 pts)

Question 2 Image Blending

Part 1 Pyramid-Based Image Blending

In this question, you will implement an image blending technique based on the *Pyramid-Based Blending* algorithm. An example of Pyramid image blending on thet wo input photos is shown in Fig. 3. Image blending can be divided into three steps:

1. Detecting features, 2. image registration, 3. image blending.

Here we only focus on image blending. Feel free to match feature points manually and implement image registration with the help of Matlab build-in function in step 1-2.

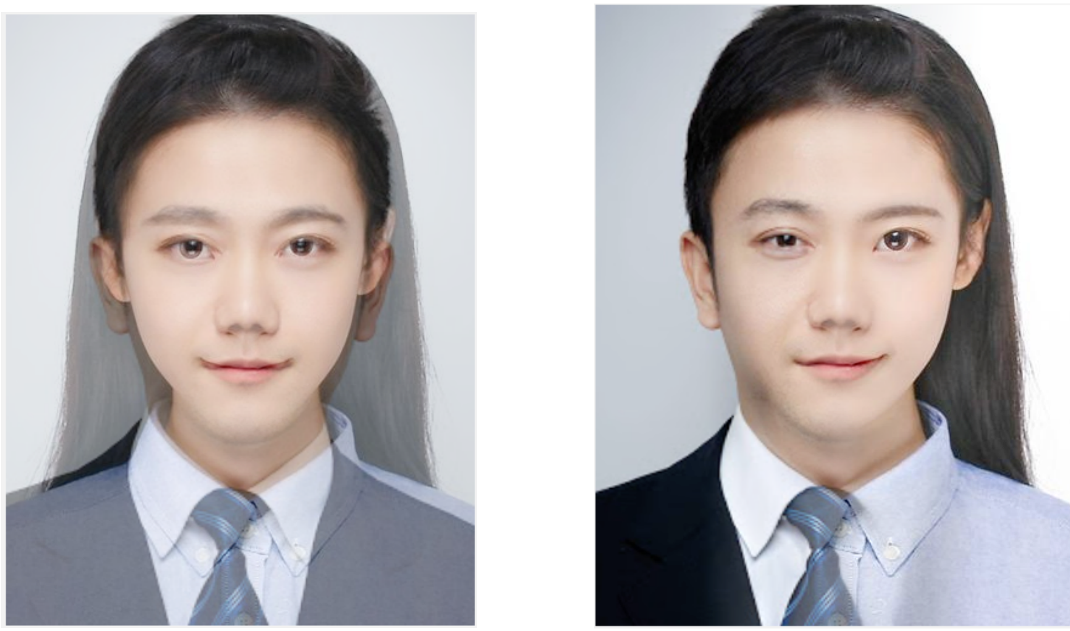
Tasks:

(1) Implement image registration or use build-in function, show your result like Fig. 4(a). Just display the image registration result, it is not necessary to be the same as the reference result.

(2) Implement image blending based on the Pyramid-based blending algorithm, and show your result like Fig. 4(b). Your results should not be worse than the reference results.

(a) Girl (b) Man

Fig 3: Source image



(a) Image registration (b) Pyramid blending

Fig 4: Reference results.

Checkpoints

You need to show figures as the above examples in your report and ensure that it is consistent with the results in your code. It is unnecessary to introduce your implementation detail. (15 pts)

Part 2 Poisson-Based Image Blending

The goal of this assignment is to seamlessly blend an object from a source image into a target image. If we naively tried a simple cut and paste, we would see noticeable seams. However, using the Poisson blending technique, we are able to achieve better results than just cutting and pasting. The technique consists on finding values for the target pixels that maximally preserve the gradient of the source region, without changing any of the background pixels. In other words, we preserve the integrity of the gradient at the seams.

To solve the Poisson blending for two images, we implement the following steps:

1. we select source and target regions (Fig. 5(a) and 5(b)),

2. we get a mask for the source image, and align the two images and the mask (Fig. 6a and 6b),

3. we solve the blending constraints,

4. we copy the solved values into the target image, where for RGB images each channel is processed separately. The cut and paste and the final blending results can be seen in the Fig. 7.

(a) (b)

Fig 5: Source image

(a) (b)

Fig 6

Fig 7

Checkpoints

You need to show figures as the above examples in your report and ensure that it is consistent with the results in your code. It is unnecessary to introduce your implementation detail. (25 pts)

Question 3

In the process of image acquisition, the image blur caused by the relative movement of the acquisition device or target at the moment of exposure is called motion blur. In the spatial domain, the degenerate function model of motion blur can be expressed as

$$g\ (x,y)=h(x,y)*f(x,y)+n(x,y) \qquad (1)$$

Among them, g(x,y) represents the output image, h(x,y) represents the degraded function, f(x,y) represents the input image, and n(x,y) represents the noise.

Equation (1) can be expressed in the frequency domain as

$$G\ (u,v)=H(u,v)F(u,v)+N(u,v) \qquad (2)$$

We can use PSF= fspecial('motion',L,theta) to simulate the convolution kernel h(x,y).

Theta refers to the angle between the direction of motion and the horizontal direction, which is called the direction of motion blur. L refers to the distance the pixel moves in the direction of motion, which is called the motion blur distance.

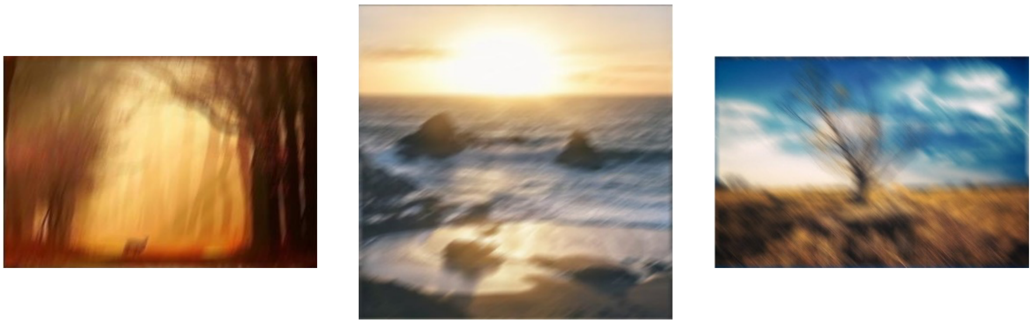


Figure 1 Blurred image

The whole processes are shown as following:

1. Convert to grayscale and implement the FFT to the image. (fft2 in MATLAB)

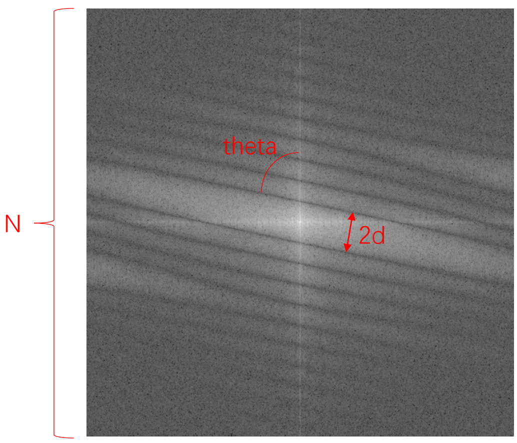


Figure 2 Spectrogram after centering

2. Estimate the paramters L and theta for a N*N grayscale.[1] (You can use other method)

Motion blur will produce periodic bright and dark stripes in the spectrogram. Theta is the angle between the stripe and the vertical direction, d represents the distance between two similar dark stripes (the distance between the two dark stripes near the center is 2d), then L=N/d.(As shown in Figure 2)

You can use the Radon transform to estimate the angle theta, and then rotate it 180-theta degree counterclockwise.

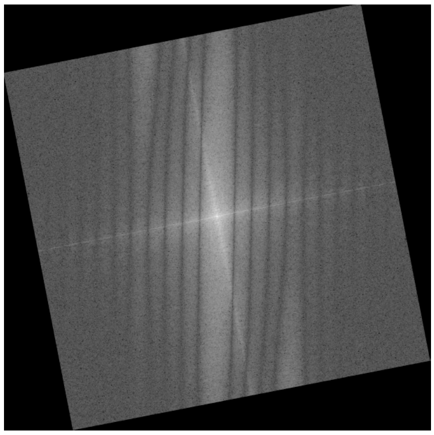


Figure 3 Spectrogram after rotation

Vertical projection of the spectrogram can be used to estimate L, L=N/d;

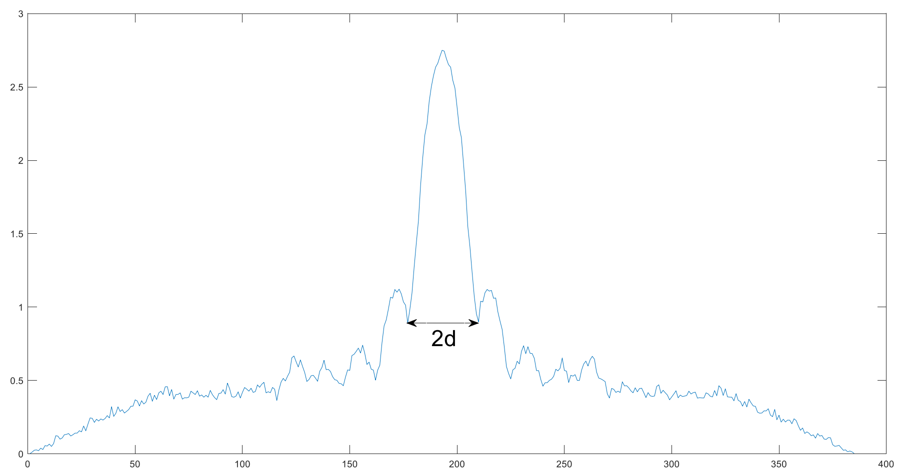


Figure 4 Average pixel of spectrogram column after rotation

3. Use the best filtering method you think to process each channel of the blurred image.

(Such as Wiener Filtering or Constrained Least Squares Filtering. You can get H(u,v) through the psf2otf function, but you need to implement the filtering by yourself)

Results:

Figure 5 Original image(left) and recovered image(right)

Figure 6 Original image(left) and recovered image(right)

Figure 7 Original image(left) and recovered image(right)

Checkpoints

1. How do you implement your algorithm? Describe it by flow charts or words. (6 pts)
2. Describe implement details of your code. (6 pts)

You need to show these following figures in your report.

1. The parameters L and theta that you get from each figure.(3*2pts)
2. Your selected filtering method and the recovered images, as well as the root mean square error of the restored image and the original image . (3*4pts)

$$e_{rms} = \sqrt{\frac{1}{3MN}\sum_{x=1}^{M}\sum_{y=1}^{N}\sum_{c=1}^{3}[\hat{f}(x,y,c) - f(x,y,c)]^2}$$

Reference : [1]廖秋香，卢在盛，彭金虎．运动模糊图像 PSF 参数估计与图像复原研究[J]．高技术通讯，2019, 29(04):40-45.