[[1]](#footnote-1)

Lab 3

Chengpin Luo, 11510880, SUSTech

*Abstract*— In this lab, we are going to implement series of basic image operations, including translation, rotation, shear, smoothing, sharpening, gamma correction and histogram enhancement. Those operations would be implemented in the spatial domain. Some of those operations would be implemented using different kinds of techniques so comparisons between them would also be given.

*Index Terms*— image transformation, image smoothing, image sharpening, contrast stretching

# INTRODUCTION

I

N this lab, we are going to perform series of image operation, which is basic, yet significant in image processing, including image translation, rotation, shear(Vertical and Horizontal), smoothing, sharpening and contrast stretching(gamma correction and histogram enhancement). All of the operations in this lab would be implemented in the spatial domain. We will see how the image is relocated, blurred, sharpened and contrast-stretched.

The following sections would be constructed as: (II) Geometric Spatial Transformation (III) Image smoothing and sharpening (IV)Contrast Stretching. (V) Conclusion.

# Geometric spatial transformation

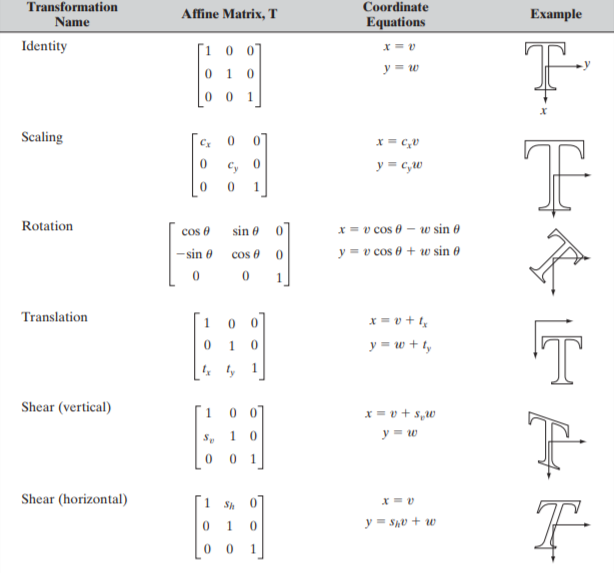
When it comes to geometric spatial transformation, we would think about the translation, rotation, scaling or shear of an image. Basically, a geometric transformation consists of two operations: (1) a spatial transformation of coordinates and (2) intensity interpolation that assigns intensity values to the spatially transformed pixels. Suppose the original pixel is at the location of (v, w) and the transformed coordinate is (x, y), then we have:

(1)

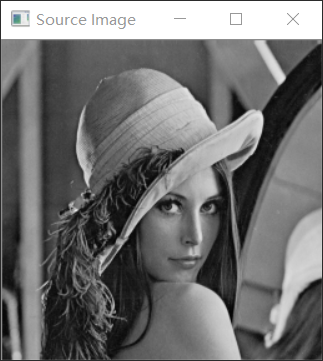
And one of the most popular used transformation is the Affine transform, which has the general form:

(2)

In table 1, series of basic transformations is given in form of affine transformation. Basically, we can compute formula (2) using forward mapping and inverse mapping. Forward mapping refers to compute (x, y) from (v, w) directly while inverse mapping means at each location, (x, y), computes the corresponding location in the input image using (v, w)= T-1(x, y) and then interpolates among the nearest input pixels to determine the intensity of the output pixel value. Inverse mapping is more efficient so it would be used in this lab. And we would use two images, lena.pgm and goldhill pgm for testing in this lab, as shown in figure 1.



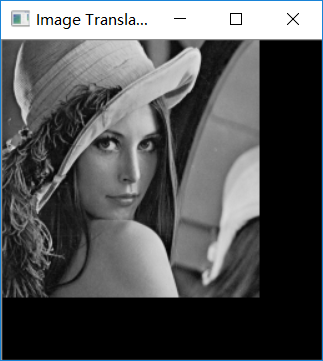
**Table. 1**.Affine Transformation

**Fig. 1.** Original Image (left: Lena, right: Goldhill)

## Translation

As shown in table 1, when we do inverse mapping, we have v=x-tx, w=y-ty. With the output image size unchanged, we could get the results in figure 2 using goldhill.pgm and lena.pgm. In figure 2, we set the offset of x and y as 50 and 50 in goldhill.pgm and -50, -50 in lena.pgm. We can see that the image is translated and some part of them disappeared beyond the boudary.

**Fig. 2.** Image Translation

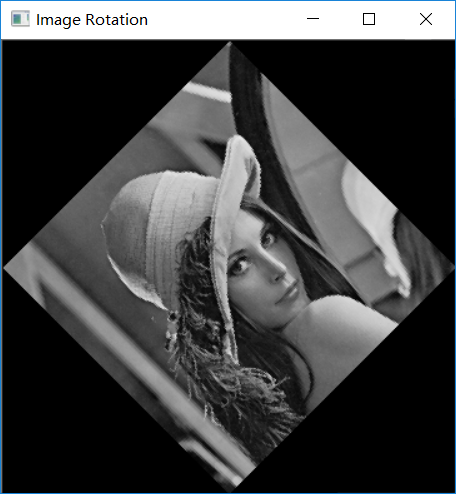
## Image Rotation

The inverse mapping of image rotation is:

(2)

where represents the angle we want to rotate counterclockwise.

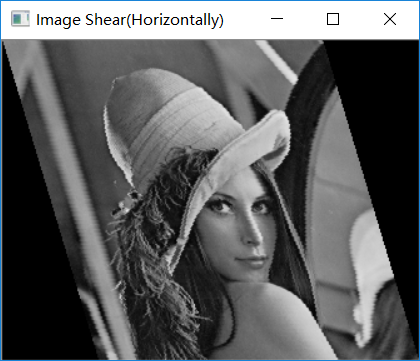
In this part, we would like to show the whole image so the size of the result image has to be changed. We substituted the three corner points (except (0,0)) into (2) and compared them to get the minimum and maximum of the coordinate. And then we could get the size of the output image. One thing need to be noticed is that we rotate the image around its central point rather than the origin, (0,0). So before inverse mapping, we have to do the shifting---add xmin and ymin to the pixels in the result image. After inverse mapping, we could get the results in figure 3, where we rotated lena.pgm for 45 and goldhill.pgm for -45 (clockwise). We could see that the image is rotated perfectly and the boudary problem in image translation has been solved.

**Fig. 3.**Image Rotation

## Image shearing

Image shearing includes shearing vertically and shearing horizontally. According to table 1, shearing is about scale one of the component (x or y) and add it to the other one. Similarly, we resize the output image and did the inverse mapping. The shearing scale is set as 0.3 and the result is shown in figure 4, where the left two is shearing vertically and the right two is shearing horizontally.

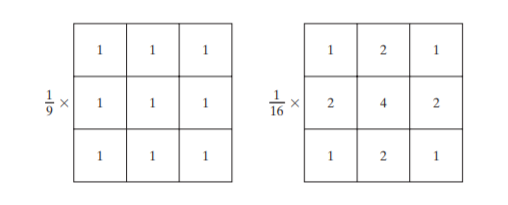
**Fig. 4.**Image Shearing

# Image smoothing and sharpening

## Image smoothing

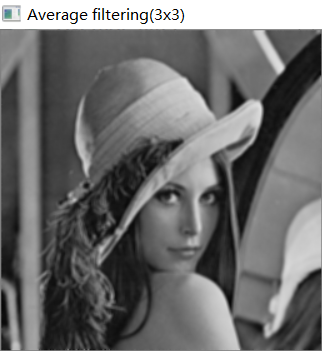
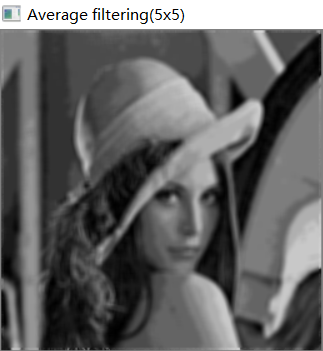
Image smoothing is about blurring the image for noise reduction, which is often implemented in the spatial domain. In this lab, we are going to use an average smoothing filter, a median filter and a binarization filter.

Firstly, we performed the average filtering. Two 3x3 average filters are shown in figure 5. The left one was used to convolve with the original image. Keeping the boundary of the original image unchanged, we could get the results in figure 6. And then we used a 5x5 average filter—averaged 25 pixels in a sub-region and assigned the value to the center pixel, which is also shown in figure 6.



**Fig. 5.**3x3 smoothing Average Filter

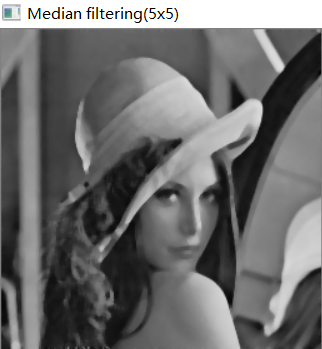
 

**Fig. 6.** Average Filtering(Left:3x3, Right:5x5)

From the results in figure 6, we could see clearly that using a filter with larger size, the result image is much more blurred. This is resonable because more pixel values are average in a trial when the filter is larger.

Now we perform the median filtering. Median filter is also about using a 3x3 or 5x5 kernel. The difference is that, we sort the pixel values under the 3x3 or 5x5 mask and then assign the median value to the center pixel in the sub-region. The result is shown in figure 7, with both 3x3 and 5x5 kernel tested.

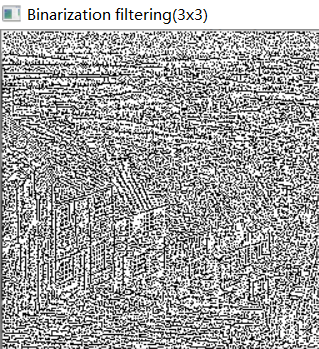
 

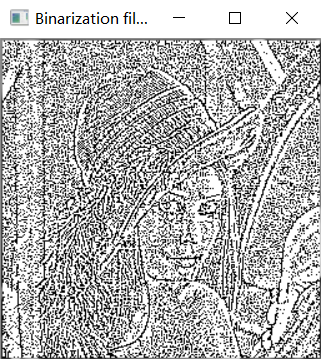
 

**Fig. 7.** Median Filtering

In figure 7, we can see that using median filter could get a less blurred image compared to average filter. Actually, median filter could provide excellent noise-reduction capabilities, with considerably less blurring than linear smoothing filters of similar size, especially salt-and-pepper noise and this is why median filter is more popular and widely-used.

Finally, we did the binarization filtering. The procedure is that we firstly found the median value under a kernel and then set all the values below median to 0 and larger or equal to the median value to 255. The result is shown in figure 8. We can see that when we applied 3x3 filter, too much details were shown that we cannot see the image clearly. With a 5x5 binarization filter, the result image looks better.

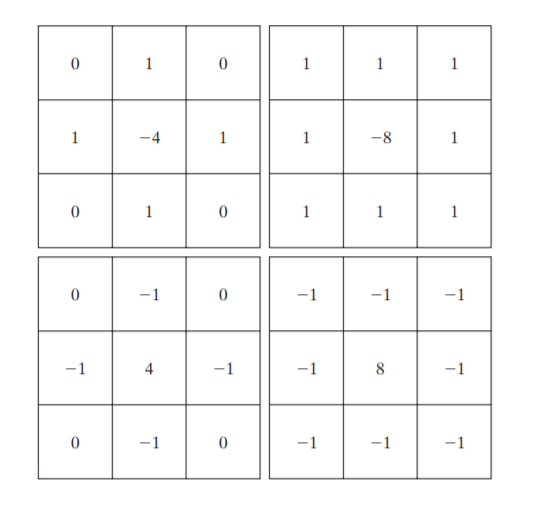


**Fig. 8**.Binarization Filtering

## Image sharpening

The principal objective of image sharpening is to highlight transitions in intensity. Basically, image sharpening could be performed using the first derivative (Sobel Operator) or the second order derivative (Laplacian Operator).

Firstly, we do the image sharpening using Laplacian operator. Four Laplacian operators are shown in figure 9. In this lab, we use the upper left operator, which could be described in formula 3, that is, to sum up the four neighbor pixels and minus the center one.



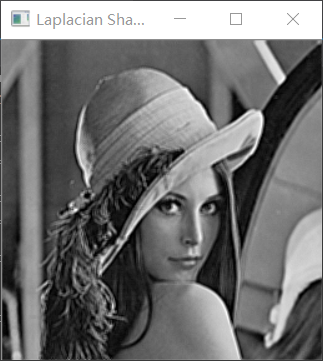
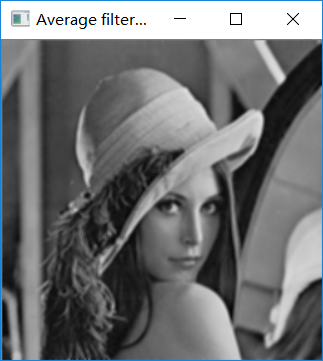
**Fig. 9.** Laplacian operator

(3)

In this way, we could get the edge of the image. And then we add it back to the original image to get a sharper one, as described by formula 4.

(4)

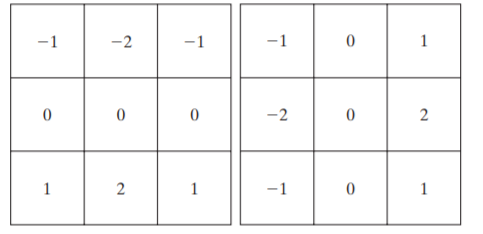
where c is -1 for the upper 2 masks in figure 9 and 1 for the lower 2. After this step, we have to limit the pixel value to the range between 0 and 255 using saturate\_cast in OpenCV. As shown in figure 10, we firstly blurred the image using a 3x3 average filter and then sharpened it using the Laplacian operator. We can see that through the masking process, the image is much sharpened than before.





**Fig. 10.**Laplacian Sharpening (Left: input, Right: output)

Then we do the first-derivative sharpening using Sobel operator. An example of Sobel operator is shown in figure 11.

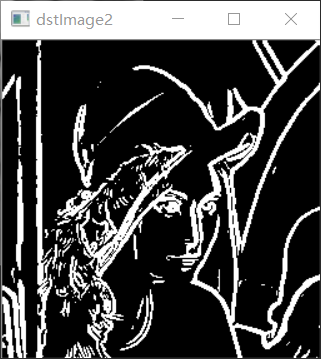


**Fig. 11**. Sobel operator

When we apply these two operators to the input image, we could obtain the gradient along the rows and columns—gx and gy respectively. And then the magnitude of the gradient vector is

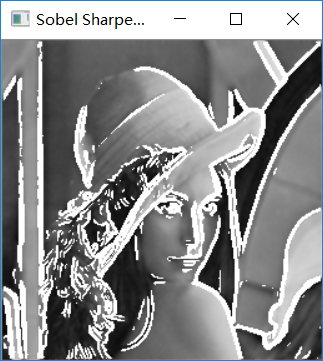
(5)

After we went through the steps discussed above and then applied a threshold to the gradient image, we could see the edge, as shown in figure 12. The threshold here is 100.

**Fig. 12.** Sobel edge detection

We finally add the edge back to the original image, as shown in figure 13. The result looks strange because the edge is highlighted so much.



**Fig. 13.** Sobel Sharpened Image

# Contrast stretching

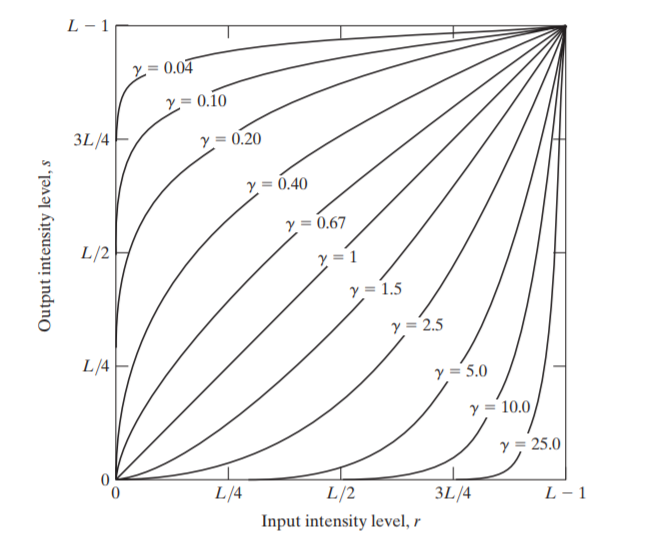
Sometimes, an image is too dark or too bright and then we need to stretch the contrast of the image. Two methods are proposed here—Gamma correction and histogram equalization.

## Gamma correction

Gamma correction is a kind of power-law transformation, which could be described in formula 6.

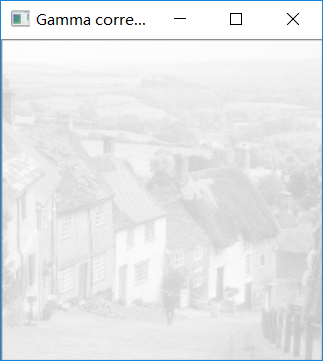
(6)

where r is the input intensity level and s is the output intensity level. A plot of s versus r under different values of gamma is shown in figure 14.



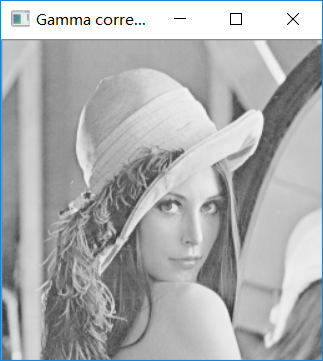
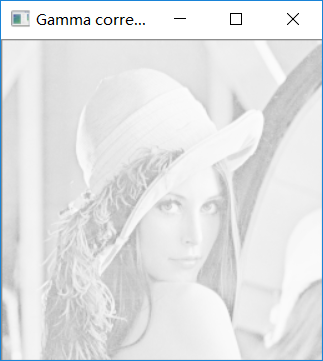
**Fig. 14.** Gamma Correction

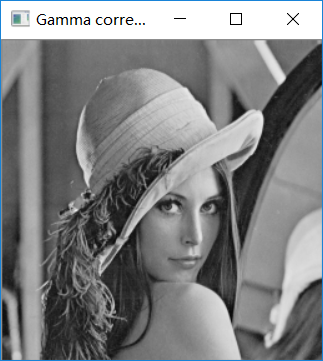
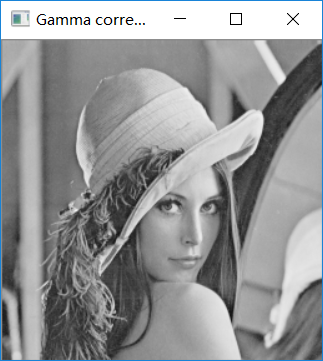
In the real implementation, we have to scale the pixel value to the range of 0-1, that is, to divide the pixel by 255. And then applied formula 6 to the pixels. Finally, we multiply the pixel value with 255 to scale the values back. The results are shown in figure 15 and figure 16. The value of gamma is 0.1, 0.4, 0.6 and 0.8, respectively.





**Fig. 15.** Gamma Transformation (goldhill, gamma=0.1, 0.4, 0.6, 0.8 from left to right, up to down, respectively)





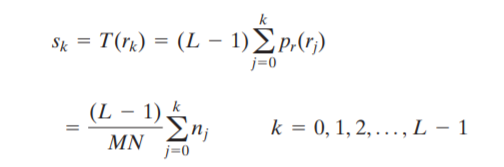
**Fig. 16.** Gamma Transformation (Lena, gamma=0.1, 0.4, 0.6, 0.8 from left to right, up to down, respectively)

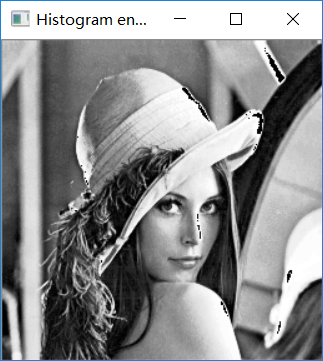
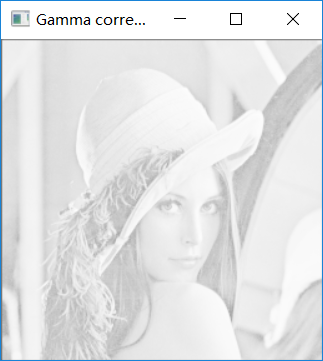
From figure 15 and figure16, we could see that when gamma<1, the pixel value was transformed to higher value when the gamma decreased. As the gamma close to 1, the result image is more close to the original image.

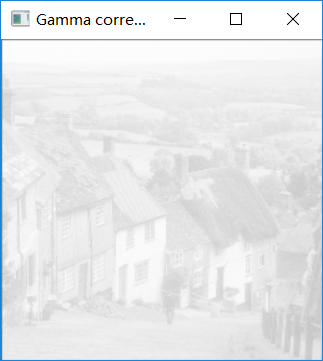
## Histogram equalization

Histogram equalization is about to stretch the range of the pixel value. Basically, we can perform this operation globally or locally.

Firstly, we perform global histogram equalization. As described in formula 7, all the pixel value would be map to a new value so that the contrast is stretched.

 (7) Taking the upper left image in figure 15 and figure 16 as the input image, we could get the output images in figure 17.

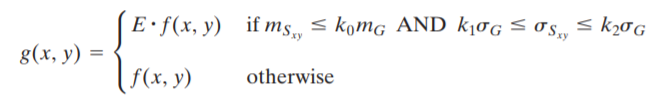




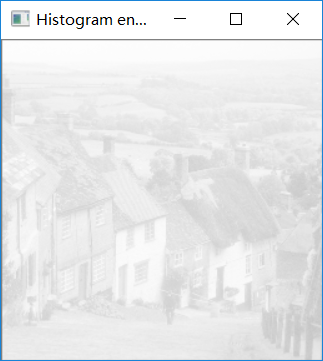
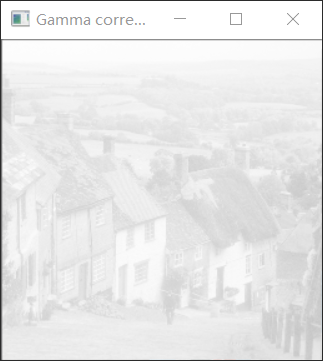
**Fig. 17.** Local Histogram Equalization (L: input, R: Output)

We can see that after equalization, the contrast of the image improved a lot, with some places distorted.

Although this global approach is suitable for overall enhancement, there are cases in which it is necessary to enhance details over small areas in an image. Therefore, we need local enhancement. The algorithm is shown in formula 8:

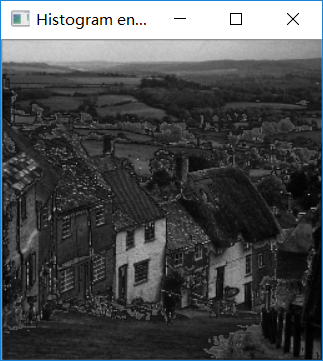
 (8)

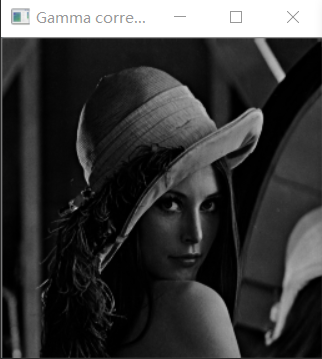
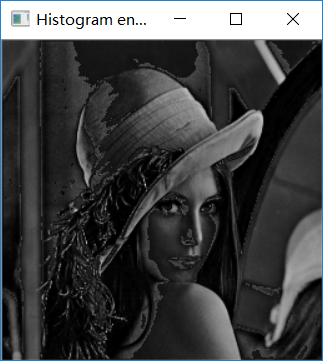
where g (x, y) is the output image and f (x, y) is the input image. mG represents the global mean of pixel value while represents the local mean. and represents the global standard deviation and local standard deviation respectively. This algorithm is aim at enhancing the contrast of the dark regions, where the local mean is smaller than the global mean, and the variance is smaller than the global variance. Besides, the variance should be larger than a threshold so that the region of constant values would not be processed. All the pixel to be processed would increase to E times of the original value. As shown in figure18, this algorithm is inefficient to equalize those dark regions.



**Fig. 18.** Local Histogram Equalization(Gamma<1)

However, when we change the gamma to a value large than 1, the local histogram equalization would perform better, as shown in figure 19. Although some distortions exist, most of the dark region could be seen much clearer.



**Fig. 19.** Local Histogram Equalization(Gamma=2.5)

# Conclusion

In this lab, we have performed several spatial transformations, such as translation and rotation. Then we implemented the spatial filtering, including smoothing and sharpening using different kinds of filter kernel with different pros and cons. Finally, we did the contrast stretching—gamma correction and histogram enhancement.

1. Chengpin Luo is with Electrical and Electronic Engineering Department, Southern University of Science and Technology, 518055, CHINA (Student Number: 11510880, e-mail: 11510880@mail.sustc.edu.cn). [↑](#footnote-ref-1)