【Experiment name】

Lab 2　image enhancement

【Purpose】

Master the "log transformation image enhancement, exponential transformation image enhancement, threshold image enhancement, histogram, and histogram equalization" methods of digital image processing based on matlab .

【Experimental content】

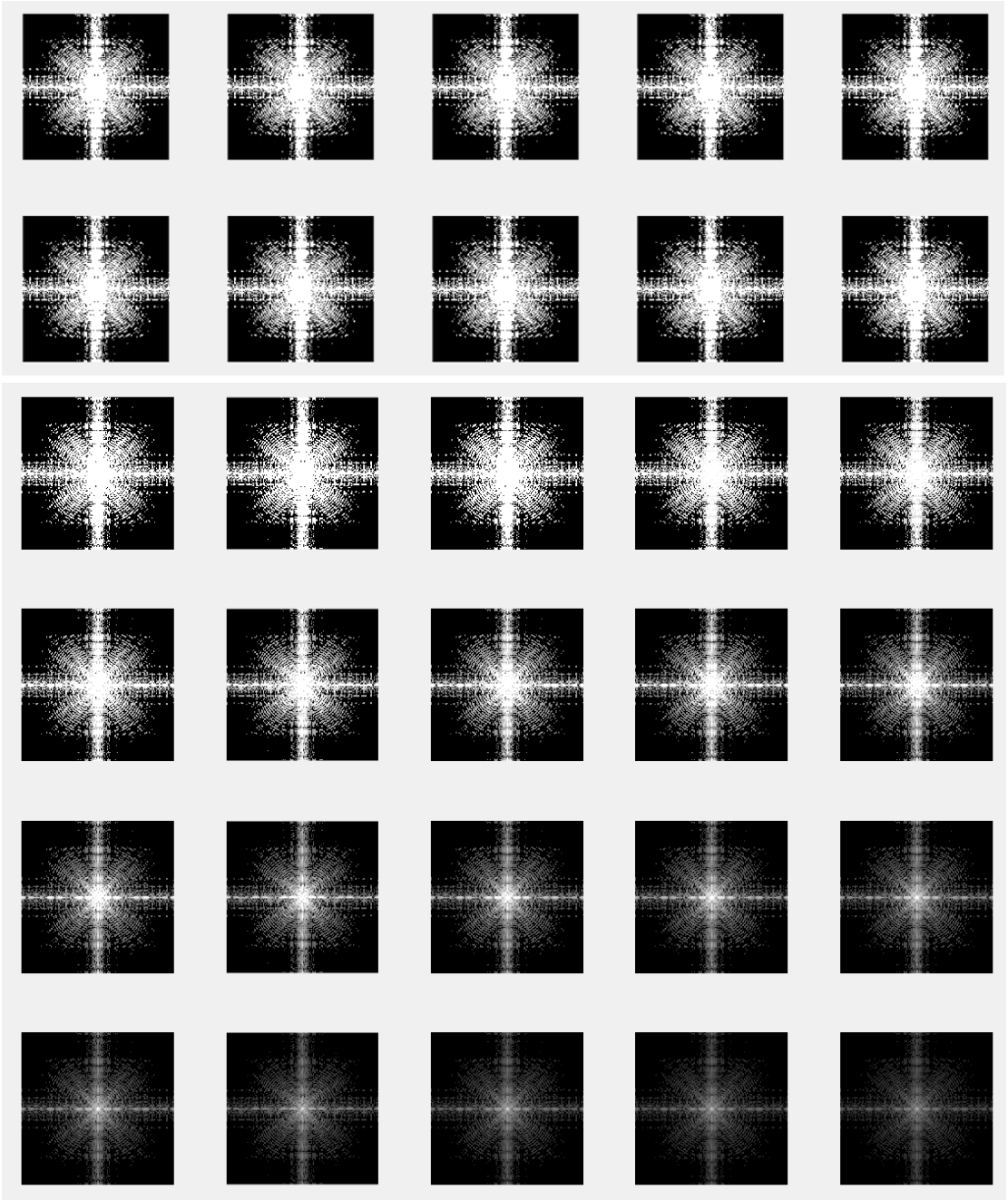
Use the given image to complete log transformation image enhancement, exponential transformation image enhancement, threshold image enhancement, histogram, histogram equalization and other operations.

PROJECT 03-01

task
Image Enhancement Using Log Transformations The focus of this project is to experiment with intensity transformations to enhance an image. Enhance the image "Fig_DFT_no_log.tif" by the log transformation of Eq.: $s = c\log(1 + r)$. Change the only free parameter c until (according to your judgment) you have the best visual result for the transformation
code
%% PROJECT 03-01 close all; clear all; I = imread('Fig_DFT_no_log.tif'); A = double(I); B = ones(size(A)); %% When the parameter is 1.1, 10 possible results are generated % c=1.1; % for i=1:10 % A1 = c^i * log(B+A); % subplot(4,4,i); imshow(A1); % end

```
%% When the parameter is 0.9, 10 possible results are generated
c=0.9;
for i=1:20
A1 = c^(i) * log(B+A);
subplot(4,5,i); imshow(A1);
end
```

Result (as shown in the figure, (4,3) is the clearest, the parameter is c =0.9^8)



PROJECT 03-02

task

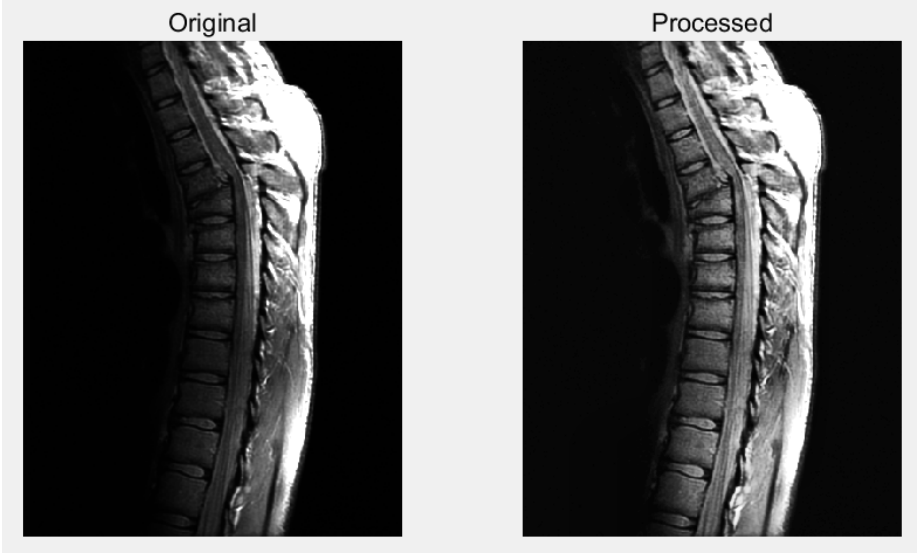
Image Enhancement Using Power-law Transformations

Enhance the image "Fig_fractured_spine.tif" by a power-law transformation of the form shown in Eq.: $s = cr^{\gamma}$. Change the two parameters, c and r until (according to your judgment) you have the best visual result for the transformation.

code

```
%% PROJECT 03-02
close all;
clear all;
I = imread('Fig_fractured_spine.tif');
subplot(2,4,1);imshow(I);
%% parameter setting and standby matrix generation r=0.9
c = 256;
r = 0.9;
[row,col]=size(I);
A = zeros(size(I));
%% generate processing
for i=2:6
for x=1:row
for y=1:col
A(x,y)=c*r^(I(x,y));
end
end
r=r+0.01;
subplot(2,4,i);imshow(uint8(A));
end
subplot(2,4,7);imshow(adaptthisteq(I));
%subplot(1,7,2);imshow(uint8(A));
%subplot(1,3,3);imhist(uint8(A));
```

result



PROJECT 03-03

task

Image Enhancement Using Thresholding

a) Write a MATLAB function **averageIntensity** which calculates the average intensity level of an image. From the command line use this function on the image "Fig_blurry_moon.tif".

(Hint: To calculate the average intensity of the pixels in an image simply iterate through every pixel in the image, summing all of their values and finally divide this sum by the total number of pixels.)

b) Write a MATLAB function **thresholdImage** which thresholds an image based on a threshold level given as a parameter to the function. The function should take two parameters **–** the image to be thresholded and the threshold level. The result of the function should be a new thresholded image. This function would be called as follows:

ThresholdedMoon = thresholdImage(Moon, ave);

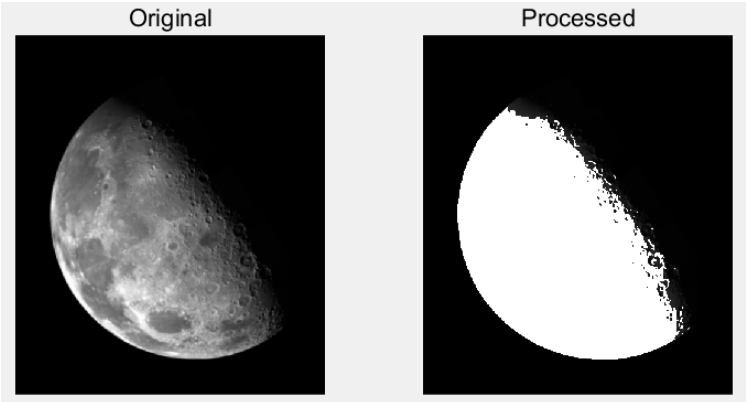
Use this new function from the command line on the image to give images similar to the following:

code

```
%% PROJECT 03-03
close all;
clear all;
I = imread('Fig_blurry_moon.tif');
subplot(1,2,1);imshow(I);title('Original');
%% basic settings
avg=averageIntensity(I);
```

```
fprintf("avg=%f\n",avg);
ThresholdedMoon=thresholdimage(I, avg);
subplot(1,2,2);imshow(ThresholdedMoon);title('Processed')
%% Topic 1 function averageIntensity
function avg=averageIntensity(I)
all=uint32(0);
[row,col]=size(I);
for x=1:row
for y=1:col
all=uint32(I(x,y))+all;
end
end
avg = double(all)/(row*col);
end
%% Topic 2 function thresholdimage
function ThresholdedMoon = thresholdimage(I, level)
[row, col]=size(I);
A=I;
for x=1:row
for y=1:col
if I(x,y)>level
A(x,y)=255;
end
end
end
Thresholded Moon = A;
end
```

result



PROJECT 03-04

task

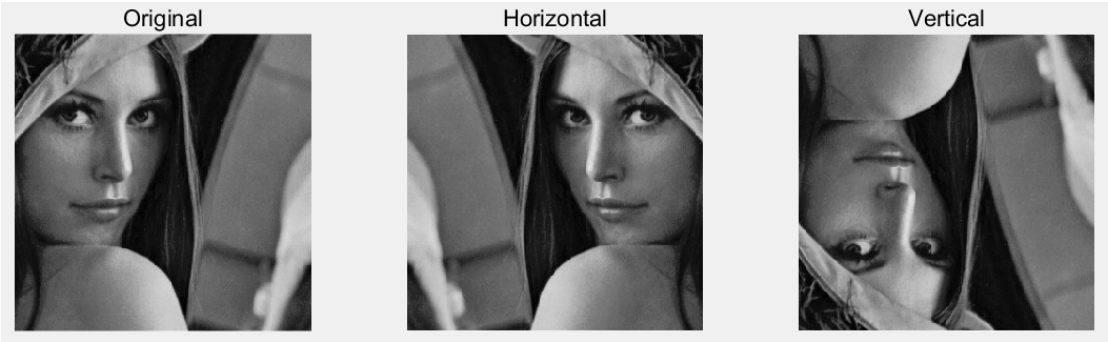
Image Flipping

Write a MATLAB function **flipImage** which flips an image either vertically or horizontally. The function should take two parameters – the matrix storing the image data and a flag to indicate whether the image should be flipped vertically or horizontally. Use this function from the command line to flip the image woman.bmp both vertically and horizontally which should give the following results.

code

```
%% PROJECT 03-04
close all;
clear all;
I = imread('woman.tif');
subplot(1,3,1);imshow(I);title('Original');
A = zeros(size(I)); % A is used to store and process images
A = flipdim(I, 2);
subplot(1,3,2);imshow(A);title('Horizontal'); % horizontal processing
A = flipdim(I, 1);
subplot(1,3,3);imshow(A);title('Vertical'); % vertical processing
```

result



PROJECT 03-05

task

Image Histogram

Write a MATLAB function, **generateHistogram**, which generates the histogram of an image. The function should take an image data array (with pixel values in the range 0 – 255) as its only parameter and return an array containing the histogram of the image. The histogram can be displayed using the built in MATLAB function hist. For example:

```
A = generateHistogram(Image);
hist(A);
```

Use this new function to generate and display histograms for the following images (darkPollen.jpg, lightPollen.jpg, lowContrastPollen.jpg and pollen.jpg).

code

```
%% PROJECT 03-05
close all;
clear all;
I = imread('darkPollen.jpg');
generateHistogram(I);

function A=generateHistogram(Image)
hist(double(Image));
end
```

result