**Digital Image Processing Laboratory**

Experiment Report

Experiment Title Basic Image Operation & Image Transformation

Student’s Name 蒋云翔

Student’s ID 2022102330

Class CST

Date handed in 2024/10/30

International school, Jinan University

**A. Objectives**

1. To know how to manipulate images
2. To be able to implement basic image transformations in Matlab
3. To be able to use intensity transformations to enhance an image.

**B. Technique**

In this project, the image **tire.tif,** **rice.png,** **kids.tif** and **spine.tif** will be used.

1. Do the basic operation on the tire and rice image.
2. Zooming and shrinking images.
3. Simple intensity transformations on images.

**C. Experiment Content**

(1) **Basic Image Operation**

1. Read the images ***tire.tif****,* ***rice.png*** given in the folder, and show them in one and two figures respectively. (referenced function: imread, imshow, figure, subplot)

|  |
| --- |
| **Code one:**  % Read images  tire = imread('tire.tif');  rice = imread('rice.png');  % Display images in one figure  figure;  subplot(1, 2, 1);  imshow(tire);  title('Tire Image');  subplot(1, 2, 2);  imshow(rice);  title('Rice Image');  % Display images in two separate figures  figure;  imshow(tire);  title('Tire Image');  figure;  imshow(rice);  title('Rice Image'); |

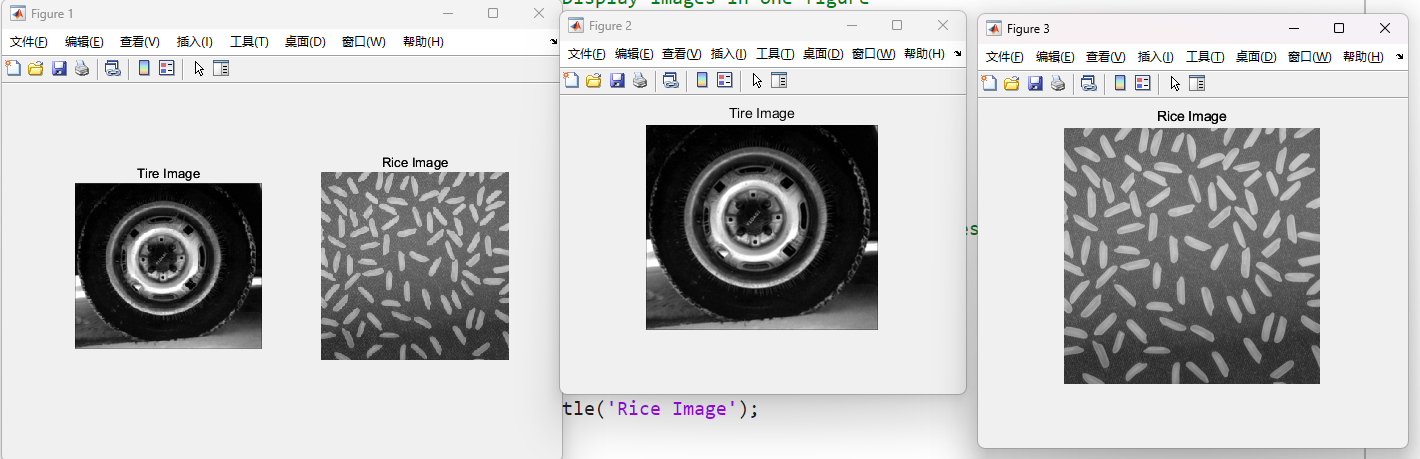


Figure 1: Read and display the images

1. In matlab, observe the images information from the workspace Panel

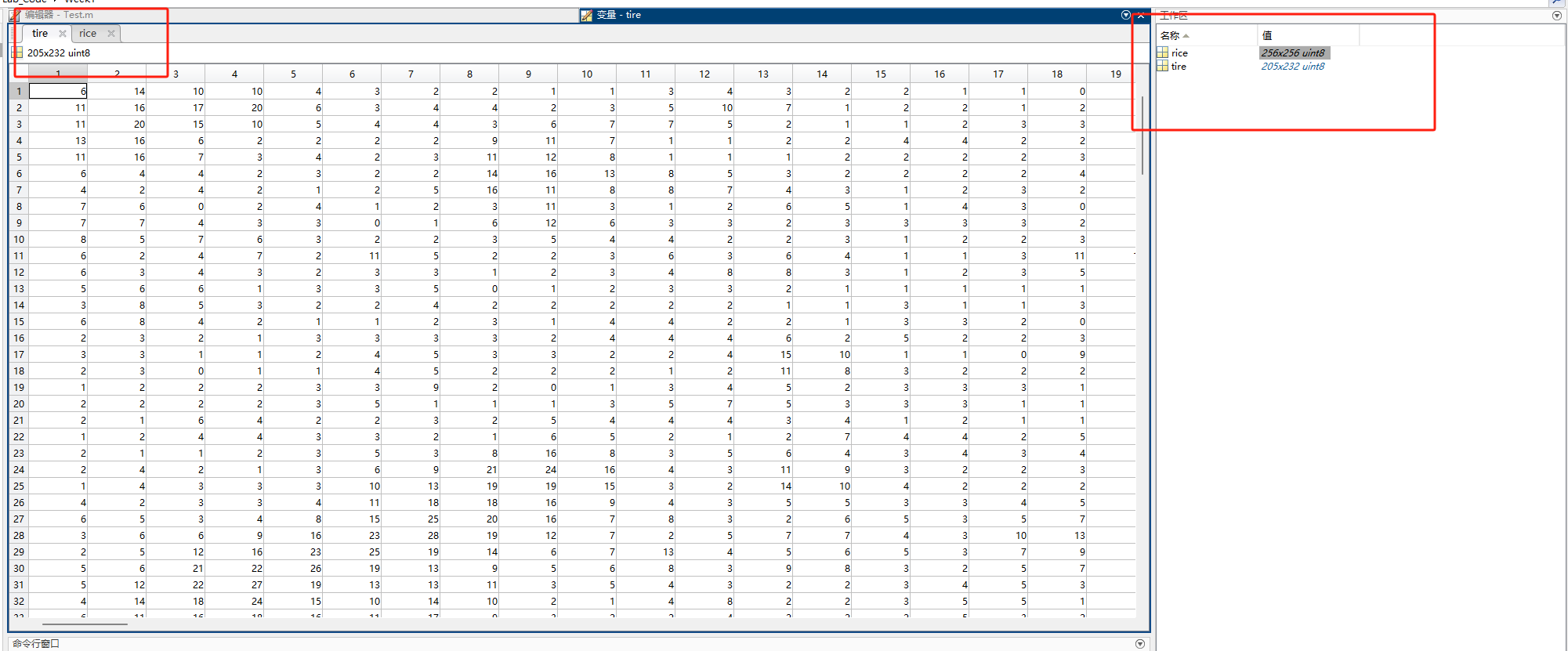


Figure 2: Images information from workspace Panel

1. Use size, imfinfo, whos, etc functions to obtain image information respectively.

|  |
| --- |
| **Code two:**  % Using size  tireSize = size(tire);  riceSize = size(rice);  % Using imfinfo  tireInfo = imfinfo('tire.tif');  riceInfo = imfinfo('rice.png');  % Using whos  tireWhos = whos('tire');  riceWhos = whos('rice');  % Display the information  disp('Tire Image Size:');  disp(tireSize);  disp('Rice Image Size:');  disp(riceSize);  disp('Tire Image Info:');  disp(tireInfo);  disp('Rice Image Info:');  disp(riceInfo);  disp('Tire Image Whos:');  disp(tireWhos);  disp('Rice Image Whos:');  disp(riceWhos); |

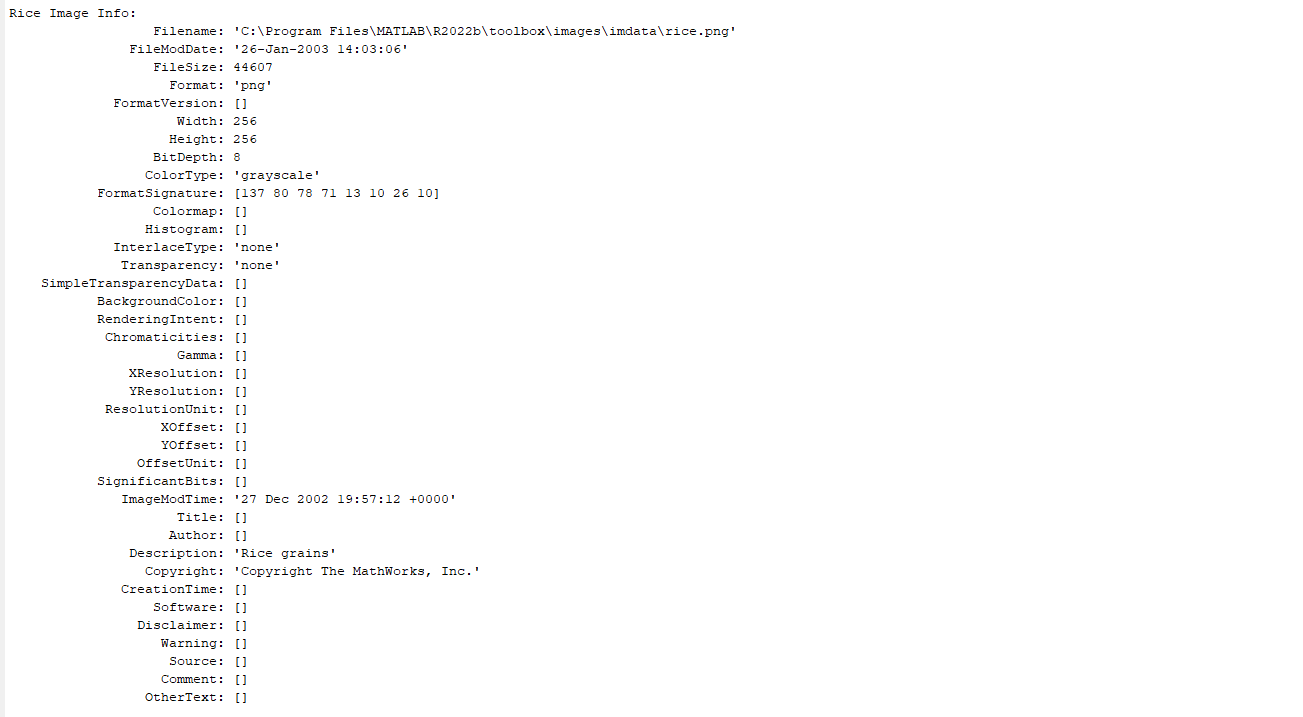
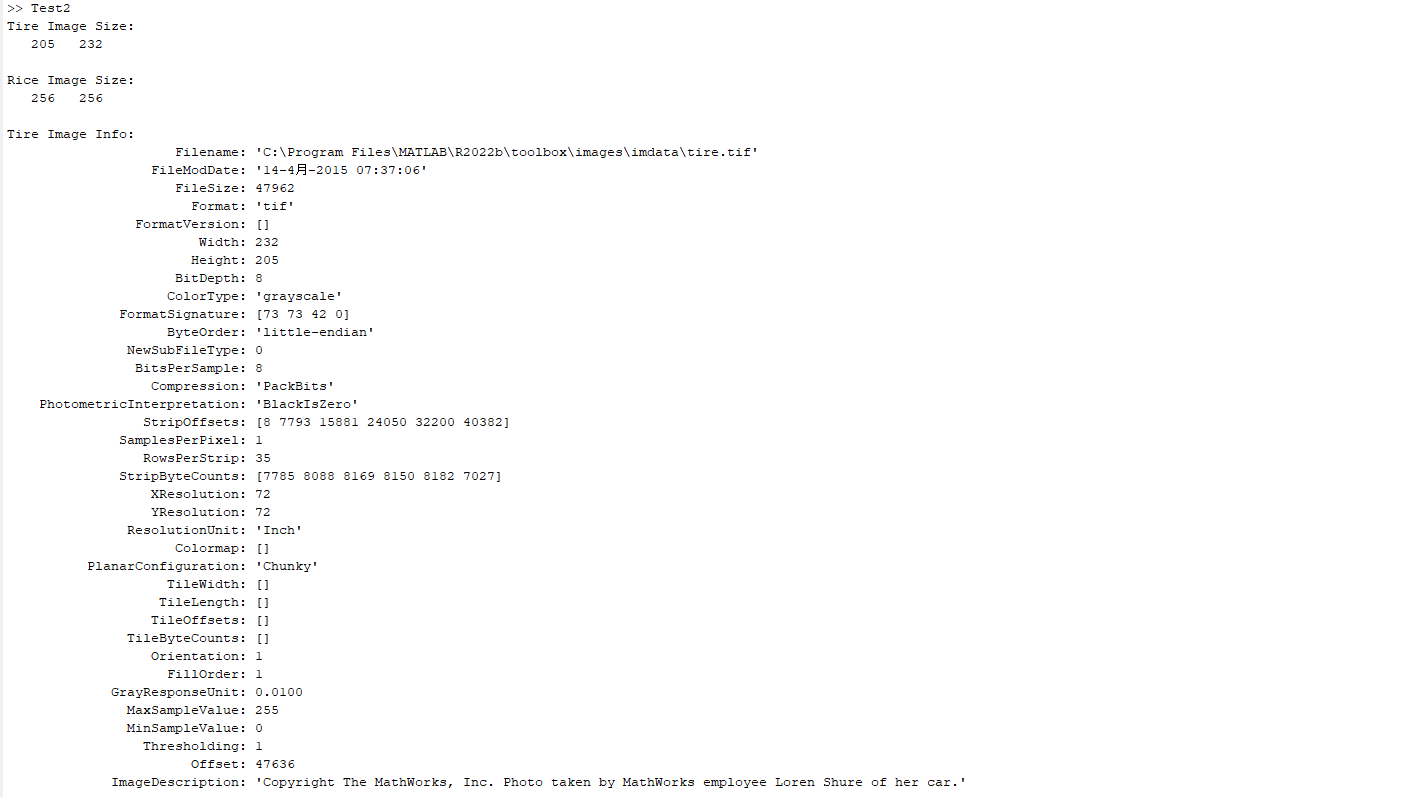


Figure 3: Using special command to show images informatioin

1. Add title to the images. (referenced function: title)

The corresponding code of this part has been referred in Code One above.

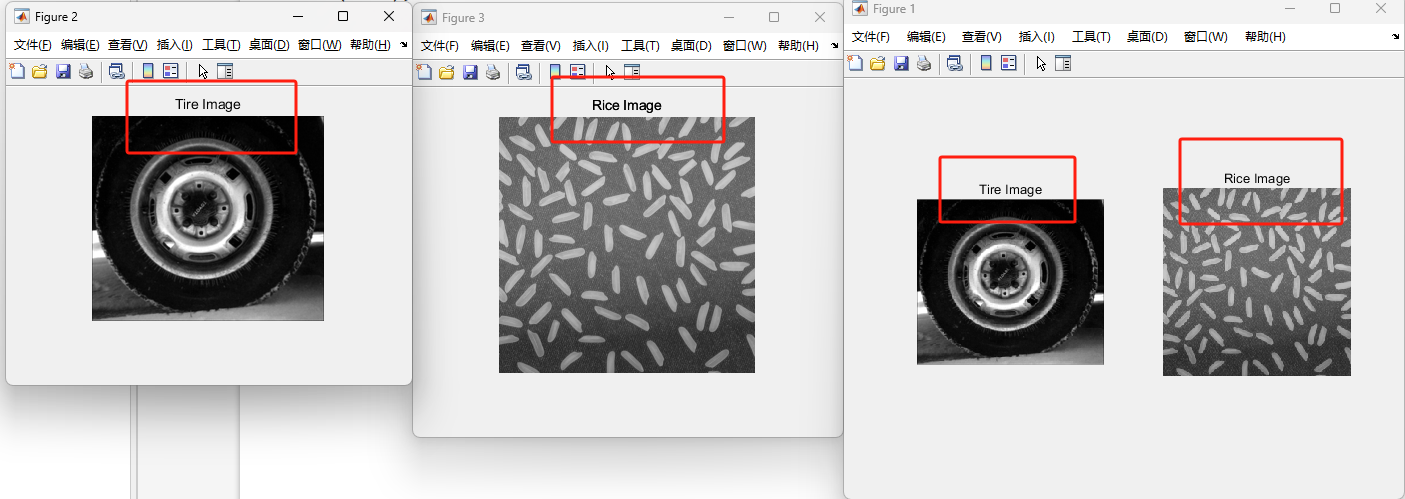


Figure 4: Add title to the images

1. Implement the following codes in the M-file Editor

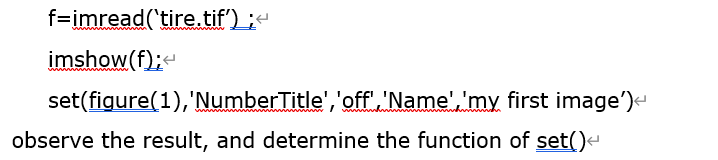


Figure 5: Apply the given code

1. Save ***tire.tif*** to the directory “d:\imagetest”. If the folder does not exist, please create this new folder first. (referenced function: imwrite)

|  |
| --- |
| **Code three:**  % Create a folder and save the image  folder = 'c:\imagetest';  if ~exist(folder, 'dir')  mkdir(folder);  end  % Save the image  imwrite(f, fullfile(folder, 'tire.tif')); |

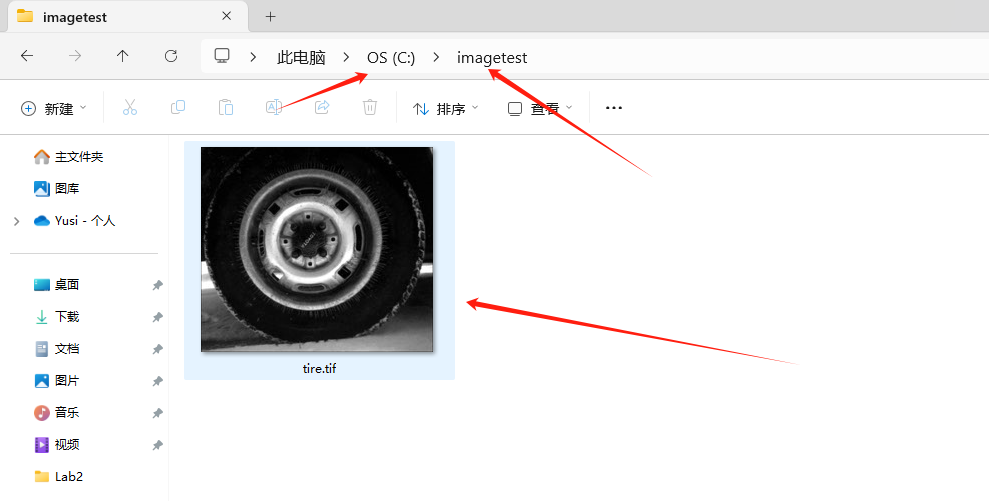


Figure 6: Save image to a directed direction

(2) **Zooming and Shrinking Images**

1. Write a program capable of shrinking(缩放) the image ***kids.tif*** by nearest, bilinear, and bicubic interpolation respectively. (referenced function: imresize)

|  |
| --- |
| Code four  % Read the indexed image with its colormap  [kids, color] = imread('kids.tif');  % Convert indexed image to RGB  kids = ind2rgb(kids, color);  % Shrink the image  kids\_nearest = imresize(kids, 0.5, 'nearest'); % Nearest neighbor interpolation  kids\_bilinear = imresize(kids, 0.5, 'bilinear'); % Bilinear interpolation  kids\_bicubic = imresize(kids, 0.5, 'bicubic'); % Bicubic interpolation  % Display the images  figure;  subplot(1, 3, 1);  imshow(kids\_nearest);  title('Nearest Neighbor Interpolation');  subplot(1, 3, 2);  imshow(kids\_bilinear);  title('Bilinear Interpolation');  subplot(1, 3, 3);  imshow(kids\_bicubic);  title('Bicubic Interpolation'); |

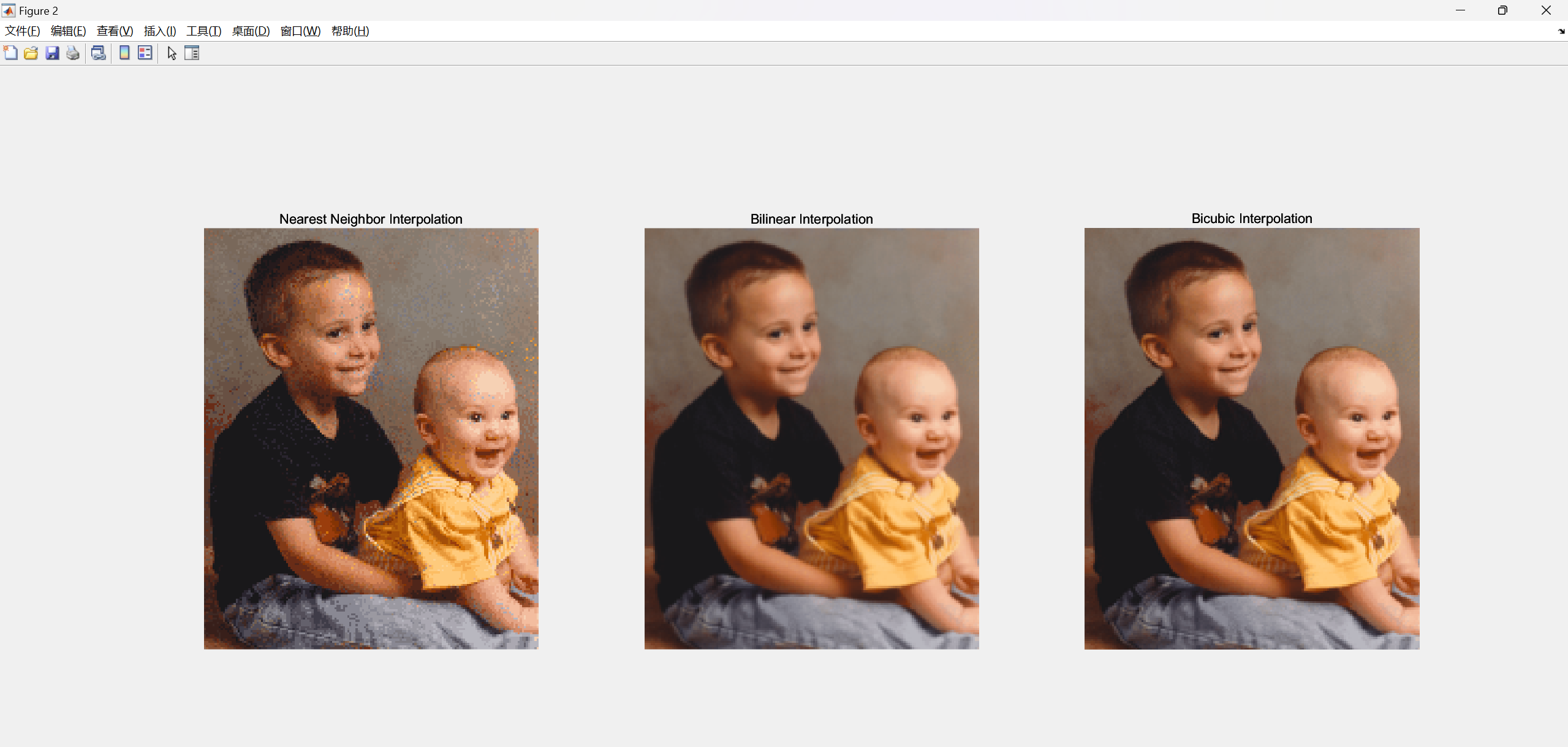


Figure 7: Shrinking the image by 3 different ways

1. Use your program to zoom the images in step 1 back to original size. Explain the reasons for their differences. (referenced function: imresize)

|  |
| --- |
| **Code five:**  % Zoom the images back to original size  kids\_nearest\_zoomed = imresize(kids\_nearest, 2, 'nearest');  kids\_bilinear\_zoomed = imresize(kids\_bilinear, 2, 'bilinear');  kids\_bicubic\_zoomed = imresize(kids\_bicubic, 2, 'bicubic');  % Display the zoomed images  figure;  subplot(1, 3, 1);  imshow(kids\_nearest\_zoomed);  title('Zoomed Nearest Neighbor');  subplot(1, 3, 2);  imshow(kids\_bilinear\_zoomed);  title('Zoomed Bilinear');  subplot(1, 3, 3);  imshow(kids\_bicubic\_zoomed);  title('Zoomed Bicubic'); |

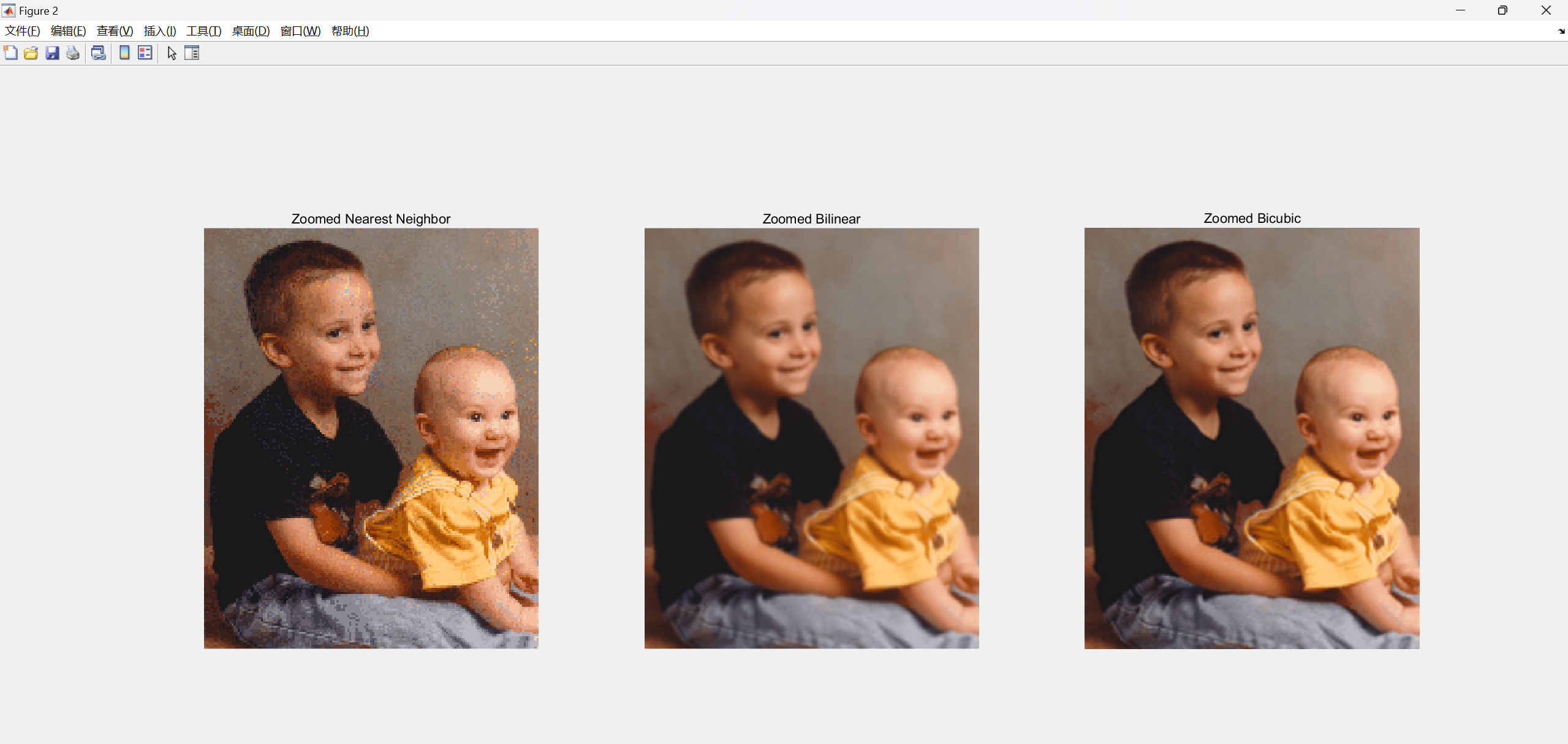


Figure 8: Zoomed images in the last step

**Explanation:** The differences in zoomed images are due to the interpolation methods:

* **Nearest Neighbor:** Simple but can result in blocky and pixelated images.
* **Bilinear:** Uses linear interpolation, resulting in smoother images than nearest neighbor, but might blur some details.
* **Bicubic:** Uses cubic convolution, providing the smoothest and most visually pleasing result, preserving more details.

1. Crop the image ***kids.tif*** to the size specified by you.

|  |
| --- |
| Code six  % Crop the image to specified size [startRow, startCol, numRows, numCols]  crop\_size = [50, 50, 200, 200];  kids\_cropped = imcrop(kids, crop\_size);  % Display the cropped image  figure;  imshow(kids\_cropped);  title('Cropped Image'); |

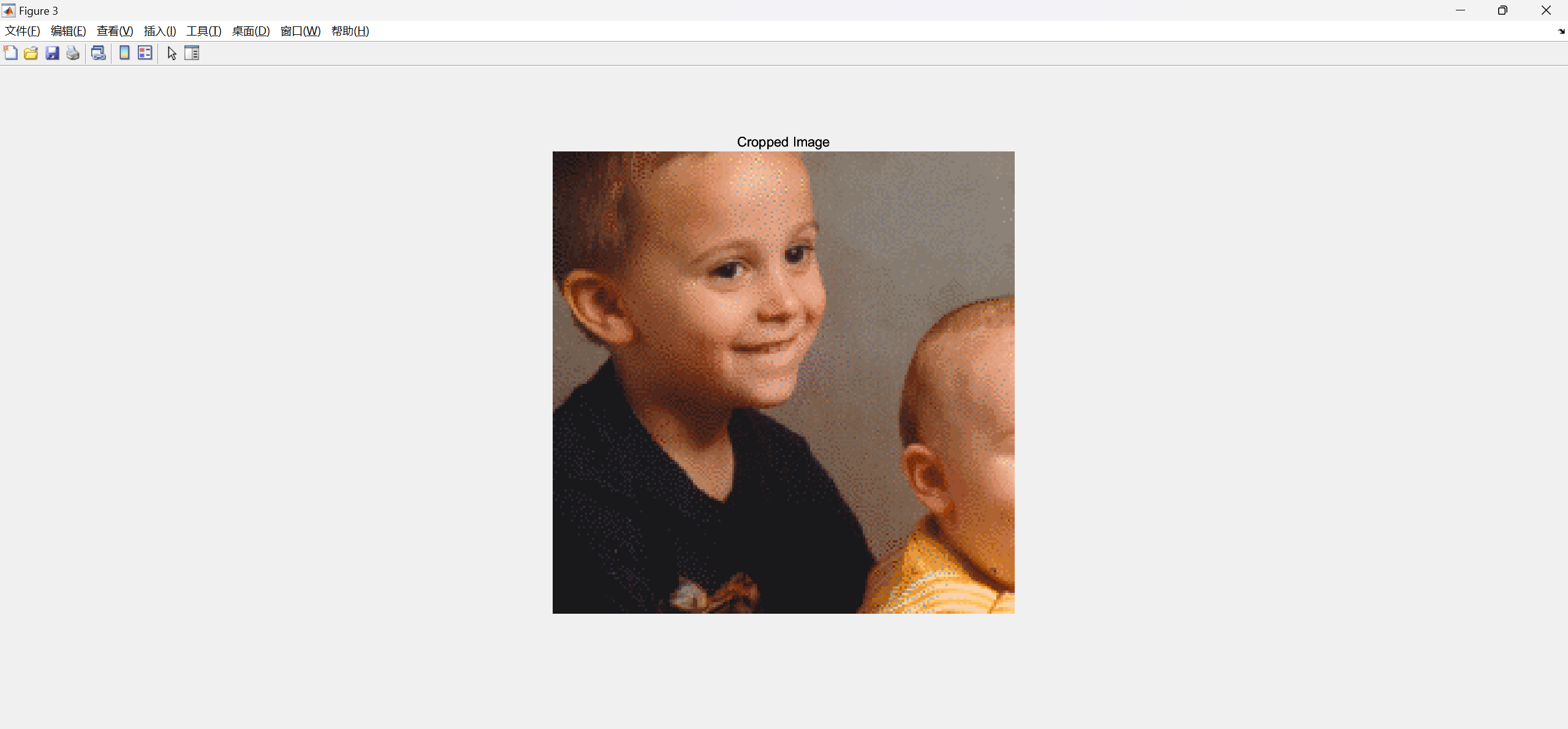


Figure 9: Cropped images

1. Rotate the image ***kids.tif*** by the degree specified by you.

|  |
| --- |
| Code seven  % Rotate the image by a specified degree  angle = 45; % Specify the degree of rotation  kids\_rotated = imrotate(kids, angle);  % Display the rotated image  figure;  imshow(kids\_rotated);  title(['Rotated Image by ', num2str(angle), ' Degrees']); |

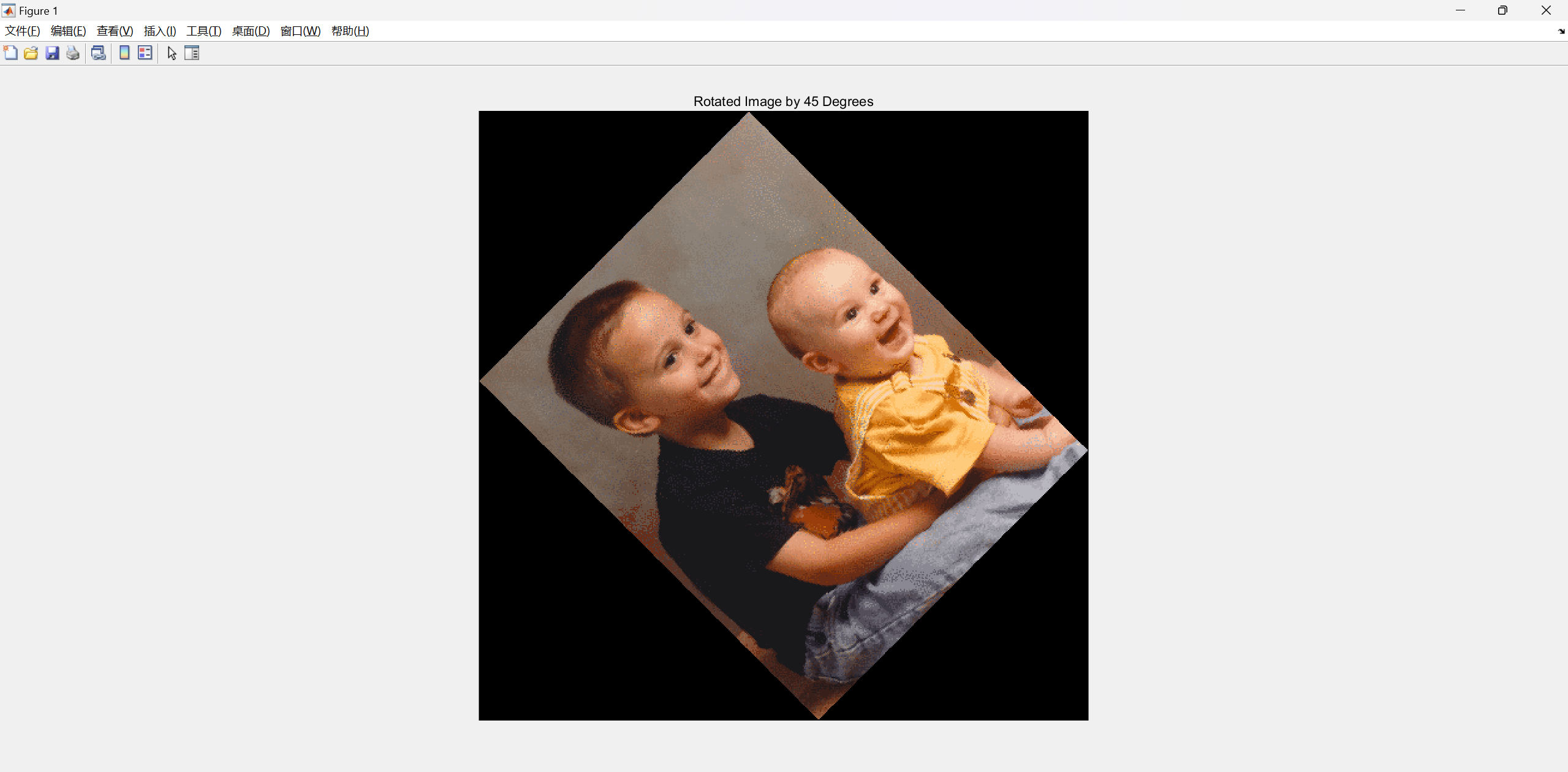


Figure 9: Rotated images by 45 degrees

(3) **Simple Intensity Transformations**

1. Use the log transformation to enhance the image spine.tif(reference function: log):



2. Use a power-law transformation of the form shown as follows to enhance the image spine.tif (reference algorithm operator: ^)



The only free parameter is c in equation (1), but in equation (2) there are two parameters, c and γ for which values have to be selected. As in most enhancement tasks, experimentation is a must.

The objective of this experiment is to obtain the best visual enhancement possible with the methods in 1 and 2. Once (according to your judgment) you have the best 7 visual result for each transformation, explain the reasons for the major differences between them.

3. I observed the impact of parameters on image processing by setting different parameters and uniformly outputting

|  |
| --- |
| Code eight(log transform)  % 读取图像  spine = imread("C:\Users\86136\Desktop\Digital Image processing\Lab\DIP LAB\lab 1\spine.tif");  spine\_double = im2double(spine);  % 参数c的不同值  c\_values = [0.5, 1, 1.5,2,2.5,3,3.5,4];  % 应用对数变换并显示结果  figure;  for i = 1:length(c\_values)  c = c\_values(i);  spine\_log = c \* log(1 + spine\_double);  subplot(1, length(c\_values), i);  imshow(spine\_log);  title(['Log Transform, c = ', num2str(c)]);  end |



Figure 10: All results using log transformation with different parameters

|  |
| --- |
| Code nine(gamma transform)  % 参数c和gamma的不同值  c\_values = [0.5, 1, 2];  gamma\_values = [0.5, 1, 1.5];  % 应用伽玛变换并显示结果  figure;  subplot\_index = 1;  for i = 1:length(c\_values)  for j = 1:length(gamma\_values)  c = c\_values(i);  gamma = gamma\_values(j);  spine\_gamma = c \* (spine\_double .^ gamma);  subplot(length(c\_values), length(gamma\_values), subplot\_index);  imshow(spine\_gamma);  title(['Gamma Transform, c = ', num2str(c), ', \gamma = ', num2str(gamma)]);  subplot\_index = subplot\_index + 1;  end  end |

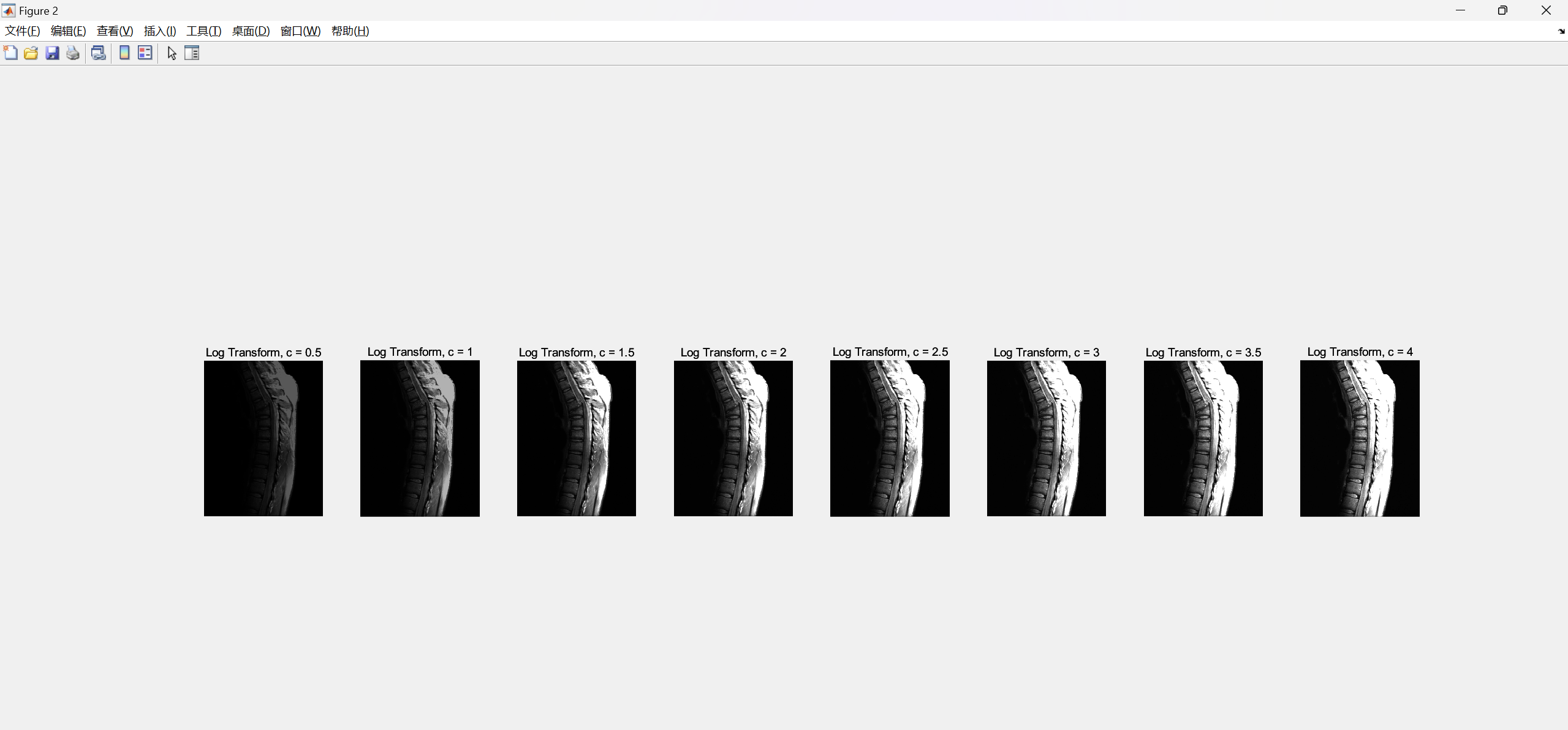


Figure 11: All results using gamma transformation with different parameters

Therefore, I chose the best visual enhancement result with the method 1 and 2, this is a comparison between them and the original image:

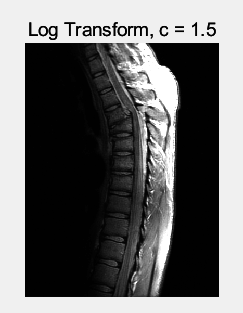
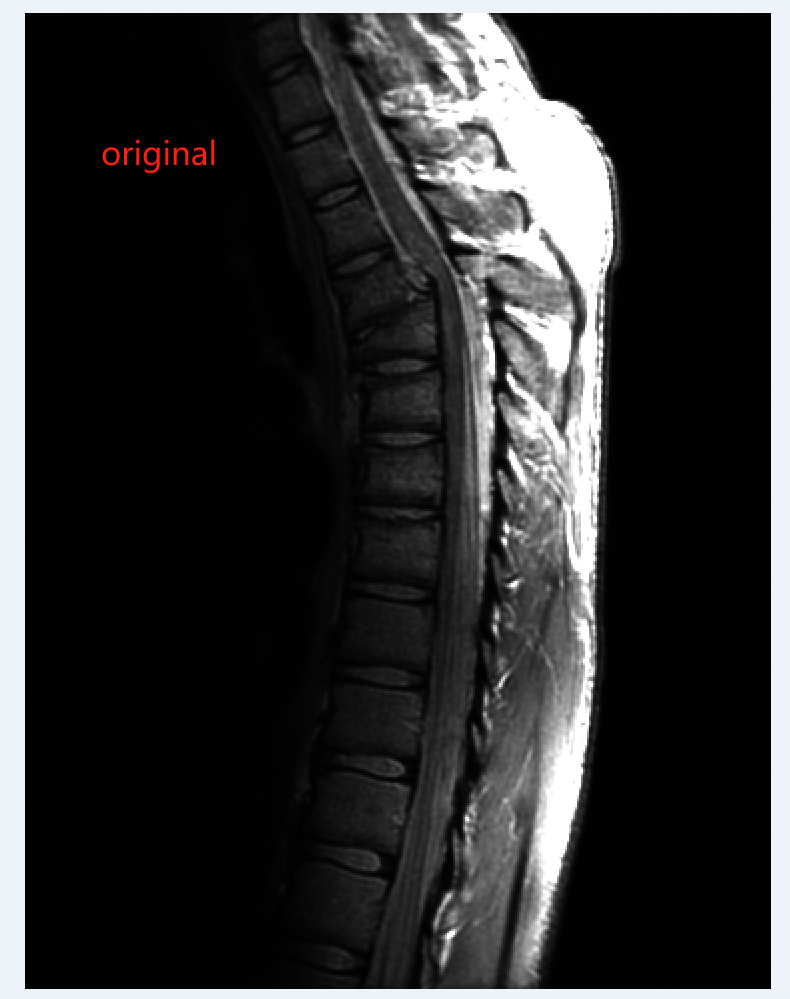


Figure 12: comparison of the original figures

**Explanation of Differences**

* **Log Transformation**: Enhances darker regions more than the brighter ones. It's useful for images with a large range of intensity values. The darker regions become more prominent.
* **Power-Law (Gamma) Transformation**: The effect depends on the value of γ\gamma:
  + If γ<1\gamma < 1, the transformation will map darker input values to higher output values (brightening the image).
  + If γ>1\gamma > 1, it will darken the image.
  + If γ=1\gamma = 1, the image remains unchanged.

Experimentation is crucial for finding the best visual enhancement. Adjust the values of cc and γto see which combination yields the best result according to your judgment.

**D. Conclusions**

Engaging in basic image operations allowed me to master the fundamentals of reading, displaying, and extracting vital information from images. Adding titles and annotations was significant, as it not only made the results more interpretable but also deepened my understanding of the images' characteristics.

The exploration of image zooming and shrinking techniques was particularly fascinating. By experimenting with different interpolation methods, I observed the unique impacts on image quality. Nearest neighbor interpolation might result in a blocky appearance, whereas bilinear and bicubic interpolations provide smoother transitions, preserving the image's integrity even after transformation.

The highlight of this lab was the intensity transformations, where I manipulated image brightness and contrast through log and power-law transformations. It was intriguing to see how different parameter values could enhance the visual appeal of an image. Through trial and error, I developed a keen sense of how to choose optimal parameters to achieve the desired image enhancement.