**School of Information Science and Engineering**

**2019－2020 school year**

Digital Image Processing Experiment Report

Course Name： Digital Image Processing

Title of Experiment： Image Display and Format Transformation

Major and Class Class 3, Communication Engineering

Student ID 201600121116

Studeng Name 张子超

Date 2019/3/9

# 1． Objectives:

(1). Get familiar with the commonly used format of image files and the transformation of format.

(2). Get familiar with the image matrix display methods.

(3). Get familiar with the image matrix format transformation.

# 2． Experiment Content:

Practicing the command for image reading and writing and conduct transforms among image formats.

Get familiar with the module functions listed below.

Image file I/O:  
imread - Read image file.  
imwrite - Write image file.  
Image display:  
colorbar - Display colorbar.  
getimage - Get image data from axes.  
image - Create and display image object.  
imagesc - Scale data and display as image.  
immovie - Make movie from multiframe indexed image.  
imshow - Display image.  
subimage - Display multiple images in single figure.  
truesize - Adjust display size of image.  
warp - Display image as texture-mapped surface.

zoom - Zoom in and out of image or 2-D plot.

# 3． Experiment Principle:

A digital image can be represented as an matrix with coordinate system for indexing, each of whose element corresponds to the intensity value of this point in the image. Under the premise, we can process the images with tools and methods for data processing. The data classes for image are *uint8, uint16, double, logical* and so on. Thus, we can use a series of gray-level transformation functions to operate on the images like re-mapping the intensity range or stretching contrast.

# 4． Experiment Steps:

## (1). Load image cameraman.tif from hard disk (use function imread).

Typing the below command at the prompt in the Command Window:

y=imread('G:\desktop documents\2018大三下\数字图像处理资料For students\For students\实验\标准图片\camera.tif');

Then the variable y shows up in the Workspace Browser:



We can be told that variable y is a 256×256 matrix in the data class uint8, which shows that this is a intensity image.

## (2). Display the image in the window (use function image or imshow).

Use both the image function and the imshow function:

imshow(y)



image(y)



It’s not hard to find the great difference between them. By researching online, I knew that the function image() displays the data in the input array as graphics, and every element in the array is acquiescently conuted as an index for the current color map. So the image displayed is in color.

## (3). Add a highlight bar to the right of the image (use function colorbar).

Simply type the colorbar command below the precedent command.

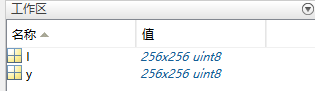


A highlight bar shows up.

## (4). Get the data of the image from the current graphics window (coordinate axis)(use function getimage).

Add a row of code below the commands, **I = getimage**, returns the first image data contained in the graphics object h.

Then a new variable I comes out in the workspace.



## (5). Display the indensity of the image form 64 to 128 (use function imagesc).

The function **imagesc** displays image with scaled colors, each element in the input matrix specifies color for one pixel. **imagesc(\_\_\_,clims)** specifies the data values that map to the first and last elements of the colormap. Specify **clims** as a two-element vector of the form **[cmin cmax]**, where values less than or equal to **cmin** map to the first color in the colormap and values greater than or equal to **cmax** map to the last color in the colormap.

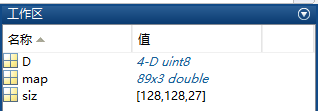
So type the command **imagesc(I,[64,128])**

Here’s what we get:



## (6). Animate a 4-D image (load mri, make animation with immovie, play it with the function movie).

Load mri means loading the MATLAB built-in mMagnetic Resonance Imaging multi-frame image.



After running the command, three new variables appear in the Workspace. *4-D uint8* means that the image is a four-dimension graphic and in data class Unsigned Integer. [128,128,27] means that this multi-frame figure is of 27 frames in total. And map(89x3) is the corresponding value of pixel matrix in colormap.

Then type the below two lines:

**mov = immovie(D,map)**

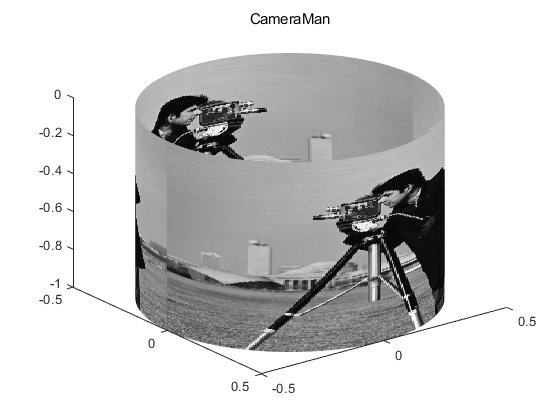
**movie(mov)**

At last a dynamic pic shows on the window.



## (7). Display the image on a main interface (use function warp).

According to the examples provided on the official site of MATLAB, a cylinder should be set up first, then use the **warp** function to display image as texture-mapped surface. The result shows below:



Use the function **cylinder** to build the premise cylinder with the syntax **[X,Y,Z] = cylinder(r,n).** Then use this cylinder as the surface, whose radium should be set to 0.5. In order to achieve the demand display effect, two figures should be put together side by side in one array as a new graph. **z=[y,y];**

# 5． Conclusions and Experiences:

During the experiment, I met a hurdle that took me a bunch of time to get over with. The 7th mission only mentioned a single function for reference, while the actual implementation requires a lot of research. I looked up the manuals in the official website of Matlab for this function, which led me to search the build of the surface. I tried to make a circle and then use an array from 0 to 1 with an internal of 1/256, but then an error occourred. So I found another website telling about how to make a picture into a 3-D sphere. That’s what stimulated me at last. So I tried to generate a cylinder and warp the picture on it and fortunately I finally successed.

# 6． Appendix（Code）:

clc;

clear all;

[y,mapy]=imread('G:\desktop documents\2018大三下\数字图像处理资料For students\For students\实验\标准图片\camera.tif');%导入图像

figure

imshow(y)

colorbar%添加Colorbar

I=getimage%使用getimage从图像中获得数据

figure

imagesc(I,[64,128])

figure

load mri%加载matlab自带的27幅mri图像

mov = immovie(D,map)%制作动画

movie(mov)%播放动图

figure

z=[y,y];%将两幅图拼到一起

[X,Y,Z]=cylinder(0.5,256);%构造圆柱体

B=warp(X,Y,-Z,z)%将这个圆柱体作为表面

title('CameraMan');