Exercise for Digital Image Processing

Experiment Preparation

1. **Environment**
   The experimental environment of this course is MATLAB + (MacOS or Windows).
   All the exercises should be written in MATLAB.

2. **About MATLAB**
   MATLAB (matrix laboratory) is a multi-paradigm numerical computing environment and proprietary programming language developed by MathWorks. MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, etc.

   a) .m files

   The program files in MATLAB are with .m extension. Functions should be written in .m files. Functions can accept inputs and return outputs. Internal variables are local to the function. You should name your function files with the same name as the functions. E.g., if your function is *myexperiment*, you should name your .m file as 'myexperiment.m'.

   b) .mat files

   Files with a .mat extension contain MATLAB formatted data, and data can be loaded from or written to these files using the functions *load* and *save*, respectively.

3. **Images**

   a) Image acquirement and save
   Images are usually provided with exercises unless otherwise specified. After processing, you should save processed images in the same path as the exercise.

   b) Image processing
   The main part of this course is about different kinds of processing on images. For images processing, you should write your own code according to the knowledge you learned during the classes. Please do not process the images by simply using the functions in Image Processing Toolbox, unless otherwise specified.

Exercise 1 Introduction & Fundamentals

Complete the following tasks. You could refer to the provided sample code but make sure to write them into .m files.

- **Task 1: Image reading and displaying**

  Download an image from the Internet. Read the downloaded image into MATLAB and observe how the image data looks like in MATLAB. Thereafter display the image in MATLAB.

  Let the downloaded image be 'a.jpg', and it is saved in 'D:¥myimages¥'.
  Then an example program may be like this:

```
im=imread('D:¥myimages¥a.jpg');
imshow(im);
```

If your current folder of MATLAB is also 'D:¥myimages¥', then you can:

```
im=imread('a.jpg');
imshow(im);
```

Optional task: display multiple images via *figure* function. A possible example might be:

```
im_a=imread('a.jpg');
imshow(im_a);
im_b=imread('b.jp')
figure, imshow(im_b);
```

- **Task 2: Image saving**

  Save your image data into disk. A possible example might be:

  ```
  imwrite(im, 'a.jpg');
  ```

  or

  ```
  imwrite(im, 'a', 'png');
  ```

  You can save your data into desired format.

- **Task 3: Convert RGB image to grayscale**

  Generally, a color image that is read into MATLAB is organized in RGB model. This task requires you to convert an RGB image to grayscale. A grayscale image is an image in which the value of each pixel is a single sample representing only an amount of light. RGB value is converted to grayscale by forming a weighted sum of the R, G, and B components: 0.2989 * R + 0.5870 * G + 0.1140 * B.
  A possible example might be:

  ```
  im=imread('a.jpg');
  imshow(im);
  im=double(im);
  im_gray=im(:,:,1)* 0.2989+ im(:,:,2)* 0.5870+ im(:,:,3)* 0.1140;
  im_gray=uint8(im_gray);
  figure, imshow(im_gray);
  ```

✧ **Notice:** This task can also be easily accomplished by the *rgb2gray* function from Image Processing Toolbox of MATLAB:

```
im=imread('a.jpg');
imshow(im);
im_gray= rgb2gray (im);
figure, imshow(im_gray);
```

However, for teaching objectives of this course, when processing images, you should write your own code according to the knowledge you learned during the lectures, rather than simple using such functions.