

# ENEE631 Fall 2001 Assignment 1

Due 9/19/2001 11:59 pm EDT

There are two parts in this assignment. Please hand in your solutions to the problems in Part-I as well as your answers, observations, and/or inductions for Part-II as hard copy. This write-up should be legible and be put in the TA's mailbox by the due time. In the mean time, please put the image results and the source codes for Part-II in your own web page. These images and codes must be well annotated and indexed in your homepage and write-up. Please remember to give the URL link in your write-up.

## Part-I

1. Consider the following experiments:

[Experiment-1] A die is rolled once.

[Experiment-2] A die is rolled  $N$  times.

[Experiment-3] A die continually until a "6" appears, at which point the experiment stops.

(a) What is the sample space of Experiment-1?

(b) What is the sample space of Experiment-2?

(c) Let  $E_n$  denote the event that the Experiment-3 has exactly  $n$  rolls of the die. What outcomes are contained in  $E_n$ ?

(d) What is the sample space of Experiment-3? \*\*

2. A random 8 x 8 binary image is generated as follows. Each pixel  $X[i, j]$  is an independent random variable taking the value "1" with probability  $p$  and the value "0" with probability  $1-p$ .

(a) For a given pixel  $X[i_0, j_0]$ , what is the mean and the variance?

(b) Let  $Y = X[i_0, j_0] + X[i_1, j_1]$ . Determine the mean and the variance of  $Y$  without computing its probability mass function.

(c) Let  $Z$  denote the random variable that equals to the average pixel value of this 8x8 image, i.e.,

$$Z = \frac{1}{64} \sum_{i,j=1}^8 X[i, j]. \text{ What is the expected value and the variance of } Z? **$$

[ Hint: try to make use of the results from 2(a) and 2(b). ]

\*\* You can choose to use your answers in Quiz-1 (9/4/01) to some or all the questions marked by "\*\*\*". If your answers there are correct, you will be given bonus points to this assignment.

3. Consider a separable 2-D signal  $s(x,y)$ .

(a) Determine the Fourier transform of  $s(x,y)$ .

(b) Show that the Fourier transform of  $s(x,y)$  is also separable.

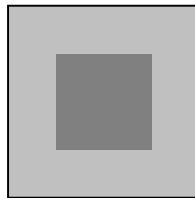
4. Jain's book (pp76) – Problem 3.6.

## Part-II

Students who are not familiar with Matlab please refer to Appendix 1, and students who are not familiar with constructing web page please refer to Appendix 2.

1. Contrast and HVS.

Generate a 100x100 square with uniform luminance  $L$ . Gradually increase the luminance of the central 50x50 square till the point you can observe the difference between it and its surroundings, as illustrated below. Record the luminance difference. Write a program to repeat this experiment (1) by also decreasing the luminance of the central square, (2) by setting  $L$  to 10 different values between 0 and 255. Plot the just-noticeable-difference you observed vs.  $L$  and include this plot in your write-up. Explain what you learned about human visual system from this experiment.



*Note:* The settings of contrast and brightness of monitors will affect the result. In this problem, set these two values of your monitor to 100%.

2. Colors.

- (1) Plot and observe the three components of the RGB image, *Lena.bmp*. Compute the sample variance for the 3 components.
- (2) Write your own Matlab code without using the Matlab built-in function to convert this color image to a grayscale one by keeping the luminance information only. Compute the sample variance of this luminance component. Include the obtained grayscale image in the webpage.
- (3) Use Matlab built-in function to convert the RGB image to HSV and YCbCr color coordinate systems. Observe the three components of each system and include them in the webpage. Compute the variance of the three components and compare the results you obtained for the three systems.

*Note:* Lena image can be obtained from <http://www.ece.umd.edu/class/enee631/am/am1/Lena.bmp>.

### 3. Down Sampling.

- (a) Divide the image *MagicBaboon.bmp* into blocks and each block has the size 4 x 4 pixels. Replace each block by the intensity of the (2,2) pixel within the block. The new image will be 1/4th the size in both dimensions. Display the downsampled image and include in the webpage.
- (b) Repeat (a) but use the (1,1) pixel instead.
- (c) Repeat (a) but replace each block by the average intensity of the whole original block.
- (d) Repeat (c) for block sizes of 2x2, 8x8, 16x16 and 32x32.

Compare results you obtained and discuss what you have learned about sampling from this experiment.

*Note:* The MagicBaboon image can be obtained from

<http://www.ece.umd.edu/class/enee631/am/am1/MagicBaboon.bmp>.

### 4. Quantization

Design 128, 64, 32, 16, 8 and 4-level uniform quantizers and quantize the gray-level image, *Baboon.bmp*. Compare the results by these six different quantizers. Explain the artifacts (e.g., the visibility of undesirable contours) and put the quantized images in the webpage.

*Note:* Baboon can be obtained from <http://www.ece.umd.edu/class/enee631/am/am1/Baboon.bmp>.

### 5. Dithering

There are a few methods to improve the quality of quantized images. One method is called dithering. Dithering suppresses the contouring effects in quantized images by adding pseudorandom noise before image quantization. The details are in Jain's book, page 120.

You are asked to write a function *mydither* (there's already a Matlab function called *dither*) which takes an input image *im*, a noise level *l*, and a number of quantization levels *q*. Create a pseudorandom noise image where every pixel takes on a random value in the range  $[-l, l]$  with a uniform distribution. Add this noise image to *im*, and quantize the result using the *q* level quantizer you wrote above. Finally, subtract the pseudorandom noise image (the same one you used before) from the quantized image to obtain the final dithered image.

Dither *Baboon.bmp* using  $l = 0.0625$  and  $q = 4$  and  $8$ , respectively. Display the results and include it in the webpage. Compare the dithered images with the quantized images without dithering. Discuss how the dithered image quality varies with *l*. Compare the resulting image with and without the final step of subtracting the noise image. Also, assuming the pixel values in the original image vary continuously from 0 to 1, describe in your write-up how you map the pixels with values beyond the original range to the  $[0; 1]$  range (after the addition of noise image) for final display.

6. Halftone

Write a Matlab program to produce a good halftone image for the grayscale image *Baboon.bmp*.

*Note:* copying or modifying codes found in Internet or similar places is not allowed. But you can do literature search and “borrow” some ideas there to design your own halftone image. Try to produce a halftone image with as good visual quality as possible. Put the image result in your web page. Explain in your write-up the method you used. Include the references, if any.

7. Image capturing

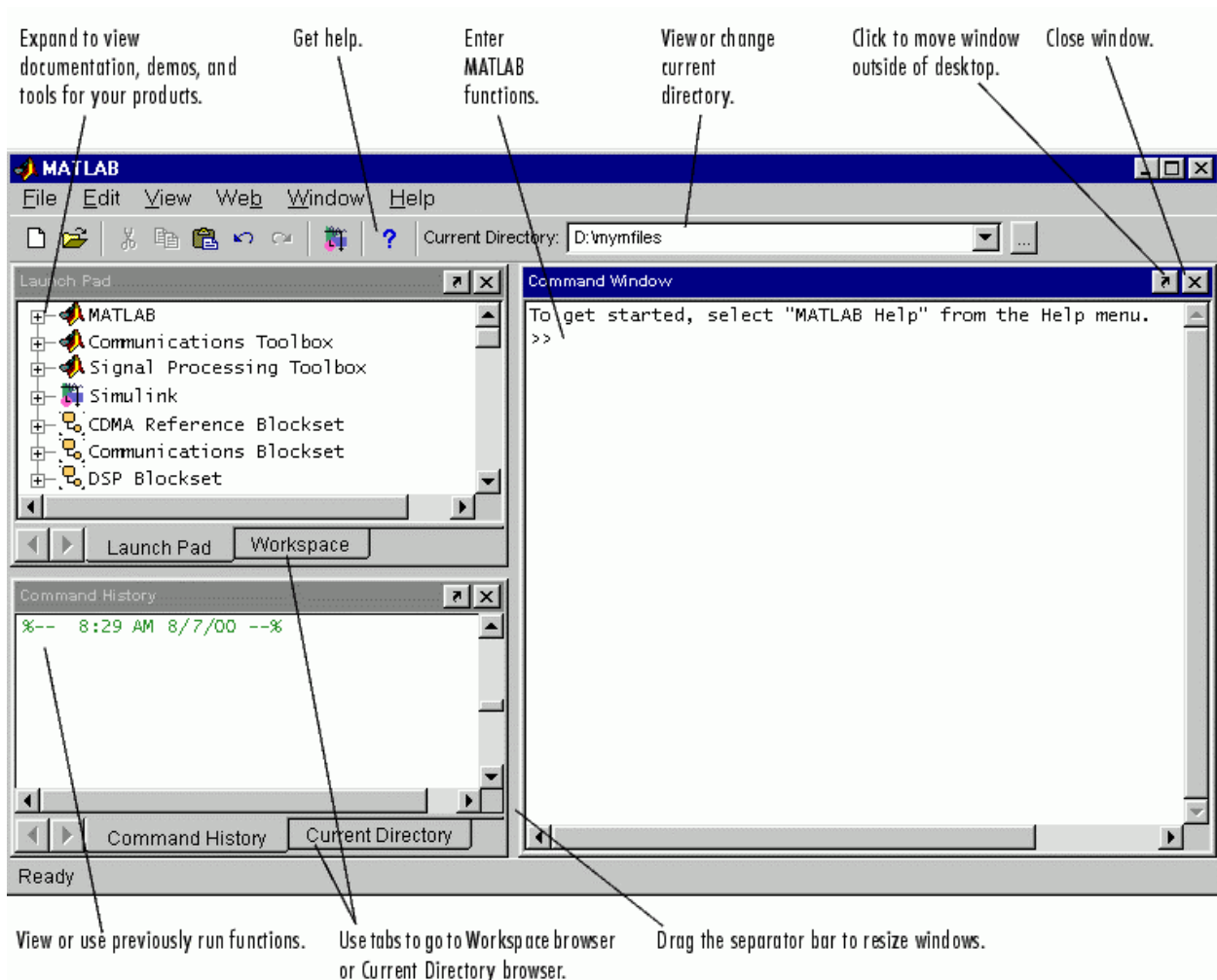
You will be asked to use PC camera to capture a digital image of yourself and include it in the webpage. A PC camera will be provided during the Friday Jasmine Lab hours. The instructions for using PC camera will be handed out before Friday.

## Appendix 1 Getting Started with Matlab Image Processing Toolbox <sup>1</sup>

This section is supposed to give you a general flavor of what can be done using Matlab's Image Processing Toolbox. It is also intended to give you a general flavor of what can be done using Matlab's Image Processing Toolbox.

### 1. Starting Matlab

Begin running Matlab by clicking on its icon under the "Start → Programs" Menu. We will be using the new version known as Matlab 6.1. Upon started, you will find the interface like this:



(From Matlab online documentation.)

<sup>1</sup> This appendix is revised from the Matlab notes of ELE488 Image Processing (Fall 1999 & 2000) at Princeton Univ. and the Matlab documentation. Prepared for ENEE631 @ UMCP by Min Wu and Guan-Ming Su.

## 2. Matlab 6.1 New Features

The new features of Matlab 6.1 include:

- Command Window  
Use the Command Window to enter variables and run functions and M-files
- Command History

Lines you enter in the Command Window are logged in the Command History window. In the Command History, you can view previously used functions, and copy and execute selected lines.

- Launch Pad

MATLAB's Launch Pad provides easy access to tools, demos, and documentation.

- Help Browser

Use the Help browser to search and view documentation for all your MathWorks products. The Help browser is a Web browser integrated into the MATLAB desktop that displays HTML documents.

To open the Help browser, click the help button in the toolbar, or type “help browser” in the Command Window.

- Current Directory Browser

MATLAB file operations use the current directory and the search path as reference points. Any file you want to run must either be in the current directory or on the search path.

To search for, view, open, and make changes to MATLAB-related directories and files, use the MATLAB Current Directory browser. Alternatively, you can use the functions *dir*, *cd*, and *delete*.

After starting Matlab, please put yourself into your working directory rather than keeping yourself in the Matlab directories: doing so could accidentally overwrite some important files!

- Workspace Browser

The MATLAB workspace consists of the set of variables (named arrays) built up during a MATLAB session and stored in memory. You add variables to the workspace by using functions, running M-files, and loading saved workspaces.

To view the workspace and information about each variable, use the workspace browser, or use the functions “who” and “whos”.

- **Editor/Debugger**

Use the Editor/Debugger to create and debug M-files, which are programs you write to run MATLAB functions. The Editor/Debugger provides a graphical user interface for basic text editing, as well as for M-file debugging.

You can use any text editor to create M-files, such as Emacs, and can use references (accessible from the desktop File menu) to specify that editor as the default. If you use another editor, you can still use the MATLAB Editor/Debugger for debugging, or you can use debugging functions, such as “dbstop”, which sets a breakpoint.

If you just need to view the contents of an M-file, you can display it in the Command Window by using the “type” function.

### **3. General Comments**

Since most of you are already familiar with Matlab, this section contains only a brief review on some important points about this software package.

- **General Philosophy**

Matlab is becoming the industry standard for numerical linear algebra, analysis, and visualization. Much of its power lies in its highly optimized operations on vectors and matrices. In many cases, you should be able to eliminate the “for” loops you used to write in C code with Matlab's simple and fast vectorized syntax.

- **Help**

Type "help command\_name" at the Matlab prompt to get help on a specific command. For a nicer interface to the help utility, click the question mark button at the top of the Matlab command window as explained above. Or, from the Help menu item, choose "Help Desk" to get to online HTML help.

From the Help Desk, you can search the online documentation to get help on how to accomplish tasks for which you may not know the specific command names. You can also try using the “lookfor” command from the command prompt.

- **Image Representation**

Many Matlab variables are matrices (arrays) with double precision complex entries. A gray scale image is just an array with real entries ranging from 0 to 1. Practical applications sometimes use 1 byte, i.e., 8 quantization levels, to represent a pixel value. This saves storage space and speeds up computation. From time to time we will also be using this representation in the labs.

- **Saving/Loading Data**

Matlab data can be stored into “.mat” files on your disk. To save your whole workspace in a file called “filename.mat”, use "save filename".

To save one particular variable called “variable\_name” into “filename.mat”, type  
"save filename variable\_name"

Saving will overwrite whatever filename you specify. To append instead of overwrite, use "save filename variable\_name -append".

Note that the saved files are in a special binary format, and hence unreadable by other applications. You can use "save ... -ascii" to save your workspace as a text file. See "help save" for more details.

Loading a “.mat” file is done by typing “load filename”. Again, you can load specific variables using "load filename variable\_name". See "help load" for more details.

- **Image Display**

For gray level images, use "imshow( image\_name, 64)", where 64 specifies the number of gray levels. See “help imshow” for details. If you're displaying a transformed image such as a DCT, you may want to zero out the DC value beforehand to make it easier to see. You could also try viewing the log of the DCT or brightening the color map with “brighten”.

- **Writing Matlab Programs**

Procedures that you call repeatedly can be stored either as functions or as macros. You create both of these by writing a “.m” file in your favorite text editor and storing it in your working directory. Macros operate on existing variables in your workspace (i.e., change them, modify them, create new ones, etc.). They do not have to be passed any inputs. You run them by typing “macro\_name” while running Matlab. Functions operate only on variables passed to them and they do not change the existing workspace. The headers of their “.m” files have to include a statement like:

```
function [out1, ... outN] = function_name( input1, input2, ..., inputN )
```

The final value of variables *out1*, ... , *outN* will be automatically returned, once the function execution is finished. User-created functions are called in the same manner as regular library functions.

- **Some Matlab tips**

In addition to basic commands, e.g., *for*, *end*, *if*, *while*, “;”, “= =”, “=”, etc, the following may be useful in this lab:

- *fft2* – a 2 dimensional Fast Fourier Transform routine. Make sure to use this command for 2-D images, and not “fft”! See the helps for these two commands to understand the difference.
- *Ctrl-C* – stops execution of any command that went awry.
- *max* – For vectors, “max(X)” is the largest element in X. For matrices, it gives a vector containing the maximum element from each column. To get the maximum element of the entire matrix X, try “max(X(:))”. Functions like *min*, *mean*, and *median* can be used in a similar manner.
- *abs* – “abs(X)” gives the absolute value of elements in X.
- Indexing – Use the colon operator and the “end” keyword to get at the entries of matrices easily. For example, to get every 5th element of the vector *a*, use “a(1:5:end)”. See “help colon” and “help end”.

## Appendix 2 How to Construct Your Web Page <sup>2</sup>

### 1. How to generate your web page in HTML language.

There are several ways you can generate your web page.

- (1) In fact, all web pages are text documents with special tags that tell the web browser what to display. An html document can be created in a text editor such as notepad, simpletext, pico, or emacs. You can find these tags at

<http://www.utoronto.ca/webdocs/HTMLdocs/NewHTML/htmlindex.html> .

Also, you can check out the HTML-related books from Web Library

<http://www.netlibrary.com> .

Be sure that your IP is within UMCP domain.

- (2) If you are not very familiar with the different tags used in html, you can use a WYSIWYG (What You See Is What You Get) Editors that resemble a Word Processor in functionality. Some examples of common WYSIWYG editors are Microsoft Frontpage, Frontpage Express and Netscape Composer. You can check out the eBooks from <http://www.netlibrary.com>, too. If you are familiar with Microsoft Word, the most efficient way to construct your web page is that you can use Microsoft Word to transform “.doc” to “.html” file format by using “File→Save As”.

### 2. Where you can put your web page.

Login your UNIX machine in WAM \ GLUE Lab or telnet to it. Type the following instructions.

```
cd /users/$USER/pub
```

Be sure to replace “\$USER” with your username. You can put all of your web pages under this directory. You can use

```
pico index.htm
```

to edit your homepage.

Your personal homepage is [http://www.wam.umd.edu/~your\\_user\\_name](http://www.wam.umd.edu/~your_user_name) , or at [http://www.glue.umd.edu/~your\\_user\\_name](http://www.glue.umd.edu/~your_user_name) .

---

<sup>2</sup> Prepared for ENEE631 @ UMCP by Guan-Ming Su.

### **3. How to transfer your files to your web site**

If you edit your web page off-line or on a PC, you should ftp your files to /users/\$USER/pub.

The method to download and install WS\_FTP can be found at <http://www.helpdesk.umd.edu/pc/apps/wsftp/> . Not sure how to use it? A pictorial explanation is shown in <http://www.helpdesk.umd.edu/pc/apps/wsftp/using/> .