EE 569 Homework Guidelines

The following are guidelines to be used when submitting homework for the course.

1. Submission Files

You may have different source code for each part of the homework assignment in your submission, but we would like to have them organized and submitted in a specific way in order to minimize the chance of misunderstanding in grading.

Please do not include any image files in your submission – we will have copies of all input files and compare the outputs that are generated by your source code directly to the results listed in your report. You are required to submit three files with each homework submission:

(1) Homework Report, .pdf format
(2) Source Code, .zip archive containing all source code (include open source code)
(3) Source Code, .txt file containing all of the source code copied into a single file (Don't include open source code here)
Each of these files should be named according to the following format:

EE569_hw<#>_<studentID>_<LastName>.<extension>
EE569_hw1_123456789_Smith.doc

Each source code file must have the following information in the top of the file:

(1) Name
(2) USC ID Number
(3) USC Email
(4) Submission Date

In addition, you must include a README.txt file along with your source code submission, which explains what each source file does and how to compile and/or run your code. These instructions should be clear enough so that the graders may unzip your source code, compile your programs (if necessary), and reproduce the results shown in your report. You will lose points for failure to include this file.

2. Image File Format

Most of the images that you will be provided will be in the RAW format. Each camera manufacturer, and many software manufacturers, have their own RAW format, which frequently include file headers with additional information; the files you will be given will NOT have any header information.

A grayscale file will have sequential bytes that correspond to each pixel in the image, row by row. This means that an image file will basically be an array of bytes that you will have to arrange into a 2 dimensional matrix that forms an image. For instance, given a file of size 24 bytes, the image could be 1x24, 2x12, 3x8, 4x6, 6x4, 8x3, 12x2, or 24x1, depending on how the bytes are arranged. As such, all length and width information will be kept outside of the file, so you'll need to manage it externally.

A colored image file will contain a similar structure, but it will have 3 bytes per pixel (one each for the Red, Green, and Blue pixels). So the first three bytes will correspond to the first pixel, the second three bytes to the second pixel, etc.

All output files, unless specified otherwise, must be in the same RAW format.

3. Programming Language

Your code may be written in MATLAB or C/C++. However, in order to make the work fair to students using both of these languages, we have added some restrictions (primarily to coding with MATLAB). See below for more details on each programming language, but remember that all of the instructions from "Submission Files" still apply.

a. MATLAB

In principle, a MATLAB function is not allowed if there is no corresponding function in C. However, it is ok to use plotting functions such as plot() or imshow() when printing results for your report.

In your README file, you must also describe how to run your code, choose arguments for invocation, and what the output file names/properties will be. It is encouraged that you make your README file an executable M-file so that you can put all of your execution commands and the file can be invoked to run all of your source code sequentially. A sample file is shown below:

```
% EE569 Homework Assignment #1
% Date:      September 1, 2004
% Name:      John Smith
% ID:        123-45-6789
% email:     jsmith@usc.edu
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Problem 1:                    Noise Removing
% Implementation 1:     Linear Transformation
% M-file name:          linear_amp.m
% Usage:                linear_amp
% Output image:         linear_amp_lena.256
% Parameters:           windows size = 3
% Other parameters here
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
disp('Running Code');   % display some useful information
figure(1)               % create a new figure window
linear amp;             % invoke M-file properly
disp('Done, output image is "linear_amp_lena.256"');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Problem 2:
…
```

b. C/C++

In your README file, describe the version of your operating system and compiler, how to choose the arguments for compilation/invocation, and what the file names of the output image are.

It is recommended that you use standard ANSI functions and compile with gcc and/or g++ if possible. This provides maximum interoperability across different platforms. We highly recommend that you prepare the README file in makefile syntax so that you can combine all the compilation and execution commands as well as invoke them using "make –f README". Alternatively, we recommend that you use a simple makefile to make compilation and execution as simple as possible for the graders. A simple readme file would have the following syntax:

```
# EE569 Homework Assignment #1
# Date:      June 1, 2007
# Name:      John Smith
# ID:        1234-5678-90
# email:     jsmith@usc.edu
#
# Compiled on WINDOWS NT with gcc
# Define compiler and linker

CC=gcc
LN=gcc
All : prob1 prob2
prob1:
      @echo "Problem 1"
      @echo "compiling and linking the code"
      $(CC) -c code1.c
      $(CC) -c code2.c
      $(LN) -o solution1 code1.o code2.o
      @echo   "running   the   program,    usage:   solution
      inputImageName outputImageName"
      solution1 foo.ppm goo.ppm
prob2:
…
```

Note that there must be a TAB, rather than white space, at the beginning of each command line.

If you use Visual C++ for your IDE, group your files carefully and include your README file that briefly explains the files and/or subfolders you use. Include only the necessary project files in your submission, but do not include any other files such as .obj files or executables.

4. Image Viewing Software

There are several pieces of software that will allow you to open RAW files for viewing, so that you can verify your code and results. You can view RAW files with Adobe Photoshop, or free RAW file viewer program such as ImageJ.

ImageJ - http://rsb.info.nih.gov/ij/

In MATLAB, you can use imshow() in command. In order to get imshow() to work correctly with color images, the matrix should be of size NxNx3, where (1:N, 1:N, 1) contains the R values, (1:N, 1:N, 2) contains the G values, (1:N, 1:N, 3) contains the B values.

5. Submission Deadlines

Submit you work by 11:59pm PST on the due date, through DEN. Additional instructions will be given for submission specifics, but if something does not function with the submission system, you may send your homework submission files directly to the graders, making sure that you send them before the submission deadline and that you include your name, student ID, and email address so that your work can be accurately recorded.

No paper submissions will be accepted. No late submissions will be accepted.

6. Report Format

The format of the report is not precisely specified, but it should at least include the following topics:

(1) Description of the motivation
(2) Description of the approach and procedures
(3) Results
(4) Discussion of results
(5) Your answers to the non-programming questions (if any)
A sample report has been provided to give an example of what an acceptable report would look like, but you are not required to use that particular format and including all the information does not guarantee correct completion of the report.

The main thing the graders look for is a clear understanding of the material. Testing out a wide variety of methods and then clearly analyzing the merits of each is what is expected of all students who wish to get a good grade.

7. Return of Homework

Homework return will also be handled electronically through DEN. The staff will make every effort to return your graded work to you in a timely fashion, but with larger class sizes we cannot always guarantee that you'll have your homework returned to you before the next one is due.

Your grades will be available in the grade center; we will notify students when grades are available. Graders will have specific office hours during the week after homework is returned so that any issues may be addressed. After the designated week, grades will be considered finalized. Graders have the final say on what the final grade is, so do not contact TAs about grading disputes unless a grader's opinion directly contradicts what the TA or professor taught.

8. References

Below is a list of websites and resources that you can use as reference material for programming:

http://www.cplusplus.com/reference/
http://www.planetpdf.com/developer/article.asp?ContentID=6634

http://www.maths.dundee.ac.uk/~ftp/na-reports/MatlabNotes.pdf