CSE 166: Image Processing, Fall 2017 – Assignment 5

Instructor: Ben Ochoa

Due: Wednesday, December 6, 2017, 11:59 PM

Instructions

- Review the academic integrity and collaboration policies on the course website.
- This assignment contains both math and programming problems.
- Programming aspects of this assignment must be completed using MATLAB.
- Unless specified below, you may not use MATLAB functions contained in toolboxes, including the image processing toolbox. Use the MATLAB `which` command to determine which toolbox a function is contained in. If you are unsure about using a specific function, then ask the instructor for clarification.
- You must prepare a report containing your solutions and results.
- Your report will be a pdf file named `CSE_166_hw5_lastname_studentid.pdf`, where `lastname` is your last name and `studentid` is your student ID number.
- All of your MATLAB source code must be included as a listing in the appendix of your report.
- Submit your report on Gradescope.
- Additionally, you must create a zip file named `CSE_166_hw5_lastname_studentid.zip`, where `lastname` and `studentid` is your last name and student ID number, respectively. This zip file will contain the pdf file and a directory named `code` that contains all of your MATLAB source code.
- Submit your completed assignment by email to `rkollipa@eng.ucsd.edu` **and** `asrikant@ucsd.edu`. The subject of the email message must be `CSE 166 Assignment 5`. Attach the zip file to the message.
- It is highly recommended that you begin working on this assignment early to ensure that you have sufficient time to correctly implement the algorithms and prepare a report.

Problems

1. **Textbook problems (15 points)**
   (a) Problem 8.5(a) (1 point)
   (b) Problem 8.9(a) (1 point)
   (c) Problem 9.8 (3 points)
   (d) Problem 9.9 (3 points)

(e) Problem 9.10 (2 points)

(f) Problem 9.11 (2 points)

(g) Problem 9.23 (3 points)

2. **Programming problems (35 points)**

(a) **The discrete cosine transform and lossy block processing (10 points)**

Develop a MATLAB script called `hw5_lossy_dct_block.m` that reads the input image `cameraman.tif` (included with MATLAB), independently processes non-overlapping blocks (i.e., subimages) of size 8×8, and writes the resulting image to file. For each block, compute the discrete cosine transform (DCT), retain the n greatest magnitude DCT coefficients (i.e., set the $64 - n$ remaining DCT coefficients to zero), and compute the inverse DCT. Perform this for $n = 32, 16, 8$ and write the resulting images to `cameraman_dct_32.png`, `cameraman_dct_16.png`, and `cameraman_dct_8.png` respectively.

Include in your report the input image and output images. Additionally, include figures of the error images (with colorbars) and the root-mean-square error associated with each output image. Include a title with each figure. Comment on the resulting images, including any differences between them.

You may use the `blockproc`, `dct2`, `dctmtx`, and `idct2` functions. Use the function `imread` to read the input image in MATLAB. Use `imwrite` to write the output image in MATLAB.

(b) **The discrete wavelet transform and lossy processing (10 points)**

Develop a MATLAB script called `hw5_lossy_dwt.m` that reads the input image `cameraman.tif` (included with MATLAB), computes the discrete wavelet transform (DWT) using the Daubechies db4 filters, sets to zero p percent of the smallest magnitude detail coefficients, and computes the inverse DWT. Perform this for $p = 50\%, 70\%, 90\%$ and write the resulting images to `cameraman_50.png`, `cameraman_70.png`, and `cameraman_90.png` respectively.

Include in your report the input image and output images. Additionally, include figures of the scaled error images (with colorbars) and the root-mean-square error associated with each output image. Include a title with each figure. Comment on the resulting images, including any differences between them.

You may use the `dwt2` and `idwt2` functions. Use the function `imread` to read the input image in MATLAB. Use `imwrite` to write the output image in MATLAB.

(c) **Counting rice grains using morphological image processing (15 points)**

Develop a MATLAB script called `hw5_morphological.m` that reads the input image `rice.png` (included with MATLAB), computes a binary image (called a logical image in MATLAB) where pixel values greater than 115 are set to 1 (and pixel values less than or equal to 115 are set to 0), and performs morphological image processing on the resulting binary image in an attempt to remove the "noise" and isolate each grain of rice, and writes the results to `rice_binary.png`. Your results should be at least as good as those shown in Figure 1. Additionally,

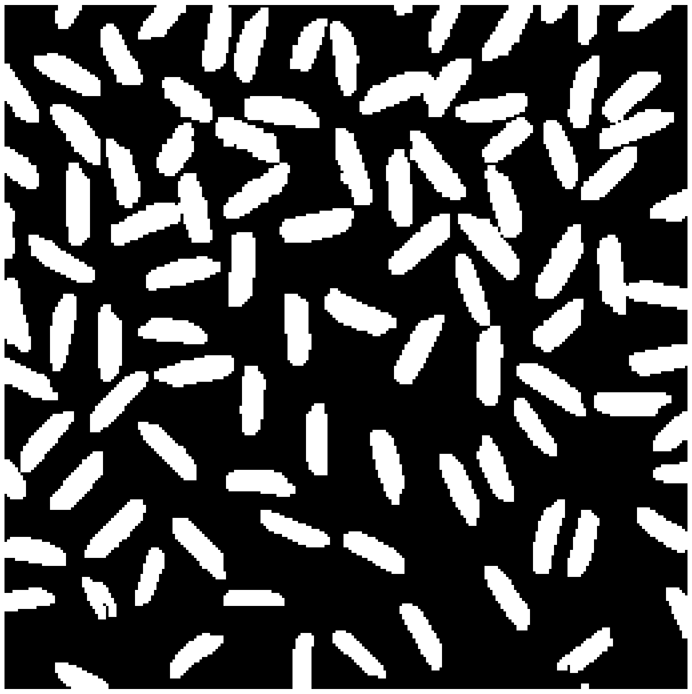

Figure 1: Results from simple morphological image processing.

the developed `hw5_morphological.m` script will *automatically* label each resulting connected component from 1 to N, where N is the number of connected components and write the results to `rice_labels.png`.

Include in your report the input image and output images. Describe the approach and comment on the results, including the number of automatically determined connected components N.

You may use the `imerode`, `imdilate`, `imopen`, `imclose`, and `strel` functions. Use the function `imread` to read the input image in MATLAB. Use `imwrite` to write the output image in MATLAB.