EE435: Biometric Signal Processing Spring 2010

Project 1: An Image Processing Function Library

Assigned: Tues 1/19/2010 Due: Tues 1/26/2010

This project introduces you to some of the basics of image processing in MATLAB, including input/output of images.

I. Image Processing Function Library

Note: ***NEVER*** *call a function, program or variable you create by the name of an existing MATLAB function.* If you do, you may no longer have that MATLAB built-in function available. To determine if the function name For example, if you are told to write a function to process the data and plot the results, **DO NOT** call the file "plot.m".

Another example: if you're trying to compute the average of a vector, don't call the result "mean." Consider this sample code (but never use code like it):

```
>>A=[1 2 3];
>>v=mean(A);  % The value of v is 2;
>>mean=v;  % Now the MATLAB function "mean" is no longer available, but the variable mean=2;
```

Write MATLAB functions that perform the following:

1. ***iseven.m and isodd.m:*** These two functions determine if an input number is even or odd. You should include error checking to ensure the input value is an integer, AND that it is scalar (not a vector or matrix). If the input is not an integer, output the null value (which is [ ]), and display a warning message.

   Usage: ***y=iseven(x); y=isodd(x) ;***

   Input: Integer ***x***
   Output: For ***iseven***, outputs a 1 if ***x*** is even, and a 0 if ***x*** is odd. For ***isodd***, outputs a 1 if ***x*** is odd, 0 if it is even.
   Suggestion: Use the MATLAB `mod` function to determine if a value is even or odd.

2. ***stats.m:*** This function computes and displays the global statistics of an input image *or* matrix; dimensions, minimum value, maximum value, range of values (max – min), mean value, and standard deviation.

   Usage: ***stats(x);***

   Input: Image (or matrix) ***x.*** Your function should immediately convert the input to double for computations.
   Output: None; only displays the statistics in the command window, does NOT return any values.
   *Important Notes:
   -The MATLAB `mean2` and `std2` functions should be used to compute the mean and standard deviation.
   -Use the `size` function to find the dimensions, e.g.,

   ```
   [rows,cols,depth]=size(x);
   ```

   -Use the `min` and `max` functions for minimum and maximum value. If the input is a matrix, these MATLAB functions will compute & return the min/max for each column instead of an overall min/max. To compute an overall min/max, you can use the colon (:) operator to perform the computation on the input as if it were a single column, e.g.,

   ```
   xmin = min(x(:));.
   ```

   -Use the MATLAB `sprintf` and `disp` functions to display the values. For example, the following line can be one of the lines within your function:

   ```
   disp(sprintf('Mean value = %f \n',mean2(x)));
   ```

   Here is an example of what your output should look like when variable *x* is a 100x120x7 matrix of floating point values:

   ```
   >> stats(x)
   Dimensions: 100 by 120 by 7
   Min. value = -189.942837
   Max. value = 187.380625
   ```

```
Range of values = 377.323462
Mean value = 0.052872
Standard deviation = 50.056965
```

3. ***extract.m*:** Extracts a rectangular shaped subimage from an input image, given the desired dimensions and the *x*-*y* location of the *center* of this subimage within the original image.

   Usage: ***y=extract(x,rcent,ccent,rows,cols);***

   Input: ***x*** is the input image, ***rows/cols*** = size of the subimage to extract, ***rcent/ccent*** = center indices of subimage.
   Output: ***y*** is the output subimage that is extracted, or *y*=[ ] for an error.

   Error Checking: It should contain some error checking, such as ensuring that the subimage is not larger than the image, or that the input *x*-*y* location values are not zero or negative-valued. If the subimage would include regions off the edges of the image, this is also an error. If an error is detected, it returns a null array and a descriptive error message is displayed in the command window.

   Important Note: Use your *iseven* and/or *isodd* functions to determine if the requested subimage has an even number of rows or columns, then use the directions provided in the "Subimage Extraction Given the Center Coordinates" handout to determine the row/column numbers to extract.

   This will be the most challenging function to write correctly. When you think you've completed it, run the *testextract* program on the course website to check if it works correctly.

4. ***threshold.m:*** Takes an input image and thresholds it to create a binary image.

   Usage: ***y=threshold(x,thresh);***

   Input: ***x*** is the input (typically it is an image), ***thresh*** is the value to use as a threshold
   Output: ***y*** is the output (which is the same size as x). For each pixel in input ***x***, if the value in ***x*** is ≥ ***thresh***, the corresponding value in output ***y*** is a 1; otherwise, the corresponding value in output ***y*** is a 0. Force the output *y* to be of type *logical* in your function, e.g.,

   ```
   y=logical(y);
   ```

   Hint: Don't use loops…use the `find` command instead (it is much, much faster). The `find` command will return the indices in a vector or matrix that meet the argument's criteria.

III. Testing Your Image Processing Function Library

Download the "OsamaBinLaden-bw.jpg" image from the course website and use the `imread` function to read the values into MATLAB. Then, use the functions you have just created to perform the following.

5. Extract the center 151 x 151 subimage from the original Osama image. To determine the center row and column number, divide the rows by 2 and columns by 2. Using your *stats* function, display the statistics of this subimage. Fill in the values in the following table.

Min Value	
Max Value	
Mean	
Std Deviation	
Variance	

6. Use your *extract* function to extract a 111 x 96 subimage from the Osama image, using the location (164,157) as the center. Write out this binary image as a .bmp file (using the `imwrite` function), and printout this picture in your writeup. In Windows, double-click on your image file and view it…if it is not what you expect, let me know. Compute the statistics of this subimage and fill in the table below.

Rows	
Cols	
Min Value	
Max Value	
Mean	
Std Deviation	

7. Using your *iseven* or *isodd* function, determine if the pixel values at the following locations in the original Osama image are even or odd.

Location	**Value**	**Even or Odd? (based on your *iseven* or *isodd* function)**
(211,54)		
(14,13)		
(101,101)		

8. Using your threshold function, threshold the Osama image with a value of 117. Write out this binary image as a .png file and include this picture in your writeup. In Windows, double-click on your image file and view it…if it is not what you expect, let me know. Use the MATLAB `sum` function to count up the number of ones that appear in the binary result.

Total Number of Ones: _____

For a lab report, fill in the blocks on these sheets; answer the questions on these sheets, provide printouts of the images asked for, and provide code for the functions you created (a formal lab report format is not required for this lab).

Subimage Extraction Given the Center Coordinates

The concept of extracting a subimage from a larger image is straightforward, but how it is to be executed depends on the size of the subimage desired, and the reference location (i.e., given the center location or the upper left corner location). There are essentially three cases to be examined, as follows.

Case 1: Subimage is of odd dimensions

When the subimage is to have an odd number of rows and an odd number of columns, the center location sits in the exact center of the subimage. The number of rows in the subimage above the center pixel = the number of rows below the center, and the number of columns to the left of the center pixel = the number of columns to the right of the center pixel. If the desired subimage is *R* x *C*, and both *R* and *C* are odd, then the subimage will have (*R*-1)/2 rows above and below, and (*C*-1)/2 columns to the left and to the right. For example, if it is desired to extract a 3x3 subimage from a 10x10 image, given center coordinates (row center, column center)=(5,4), the situation is as shown in Fig. 1.

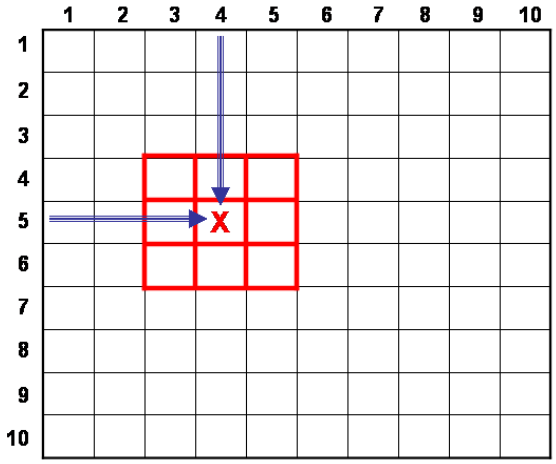

Figure 1: A 3x3 subimage in a 10x10 image centered at (5,4)

In this case, the row numbers to be extracted lie (3-1)/2 = 1 row above and below the center row, and the column numbers to be extracted lie (3-1)/2 = 1 column left and right of the center column. In MATLAB, if the image array is called *A* and the subimage array is called *B*,
>>B=A(4:6,3:5).

If we desired a 7x5 subimage, the row numbers to be extracted lie (7-1)/2 = 3 rows above and below the center row, and the column numbers to be extracted lie (5-1)/2 = 2 columns left and right of the center column.

Case 2: Subimage is of even dimensions

When the subimage is to have an even number of rows and an even number of columns, there is no exact center location of the subimage. If a center location is desired, we must specify how to carry out the extraction. In our class, we will have one more row above the center location than below, and one more column to the left than to the right. If the desired subimage is *R* x *C*, and both *R* and *C* are even, then the subimage will have *R*/2 rows above and *R*/2-1 rows below, and *C*/2 columns to the left and *C*/2-1 columns to the right. For example, if it is desired to extract a 4x4 subimage from a 10x10 image, given center coordinates (row center, column center)=(8,8), the situation is as shown in Fig. 2.

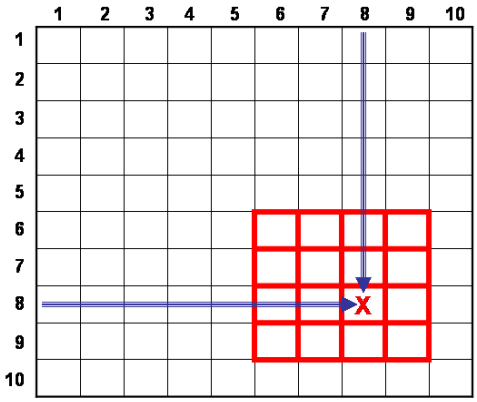

Figure 2: A 4x4 subimage in a 10x10 image centered at (8,8)

In this case, the row numbers to be extracted lie 4/2 = 2 rows above and 4/2-1 = 1 row below the center row, and the column numbers to be extracted lie 4/2 = 2 columns left and 4/2-1 = 1 column to the right of the center column. In MATLAB, if the image array is called *A* and the subimage array is called *B*,
>>B=A(6:9,6:9).

If we desired a 6x10 subimage, the row numbers to be extracted lie 6/2 = 3 rows above and 6/2-1 = 2 rows below the center row, and the column numbers to be extracted lie 10/2 = 5 columns left and 10/2-1 = 4 columns to the right of the center column.

Case 3: Subimage is of a mix of odd and even dimensions

In this case, the methods used above for odd or even dimensions apply individually. For example, if it is desired to extract a 3x4 subimage from a 10x10 image, given center coordinates (row center, column center)=(3,4), the situation is as shown in Fig. 3.

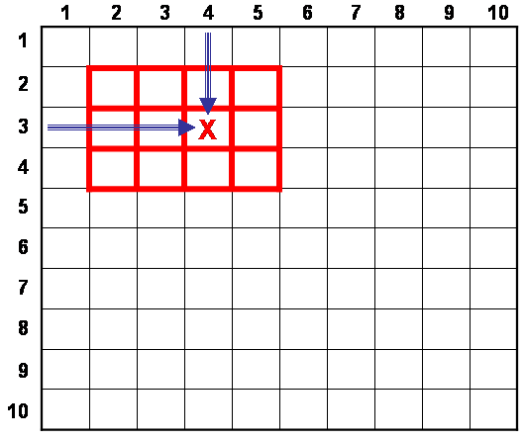

Figure 3: A 3x4 subimage in a 10x10 image centered at (3,4)

In this case, the number of rows of the subimage is odd, so the row numbers to be extracted lie (3-1)/2 = 1 row above and (3-1)/2 = 1 row below the center row. The number of columns is even, so the column numbers to be extracted lie 4/2 = 2 columns left and 4/2-1 = 1 column to the right of the center column. In MATLAB, if the image array is called *A* and the subimage array is called *B*,
>>B=A(2:4,2:5).