ECE-5554 Computer Vision: Problem Set 0

Murat Ambarkutuk

murata@vt.edu

Mechanical Engineering Department,

Virginia Polytechnic Institute and State University

August 30, 2015

ECE-5554 Computer Vision: Problem Set 0

Matlab Code

Answer Sheet

Short answer problems

1. Completed.

2. (a) Creates a row vector containing random permutations of numbers between 1 and 1000.

   (b) Line 1: Creates a matrix:

$$a = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

   Line 2 assigns the second row of x to the variable b.

$$b = [4, 5, 6]$$

   (c) Creates a matrix:

$$a = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

   Line 2 assigns the all the values the variable b in column vector form.

$$b = [\ 1,\ \ 4,\ \ 7,\ \ 2,\ \ 5,\ \ 8,\ \ 3,\ \ 6,\ \ 9\ ]'$$

   (d) Line 1 creates the column vector f [1×5] with the normally distributed random values.

   Line 2 sets another variable and fills it with the elements of f which are above 0.

(e) Line 1 sets a row vector [1×10] with zeros and adds 0.5 to each element of it.

$$x = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0] + 0.5$$

$$x = [0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5]$$

Line 2 creates another row vector with the same size of vector x [1×10], and multiplies each element of new row vector with 0.5.

$$y = [1, 1, 1, 1, 1, 1, 1, 1, 1, 1] \times 0.5$$

$$y = [0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5]$$

Line 3 sums vector x and y and assigns the result to z.

$$z = x + y$$

$$z = [1, 1, 1, 1, 1, 1, 1, 1, 1, 1]$$

(f) Line 1 creates a row vector [1×100] which contains the sequence starting from 1 to 100 (inclusive).

$$a = [1, 2, 3, 4 \ldots 98, 99, 100]$$

Line 2 flips the vector and sets vector b with these values.

$$a = [100, 99, 98, 97 \ldots 3, 2, 1]$$

3. The code is in PS0_1-3.m file.

(a) The function returning n trials of 6-sided dice roll is given below. (Please find the diceTrials.m file .zip file.) The contents of diceTrials.m:

```
1 function result = diceTrials(n)
2     if(n>0)
3         result = uint8(rand([1,n])*5)+1;
```

```
4       else
5           result = 'You may wanna not to do that operation, the number
6           error(result)
7   end
```

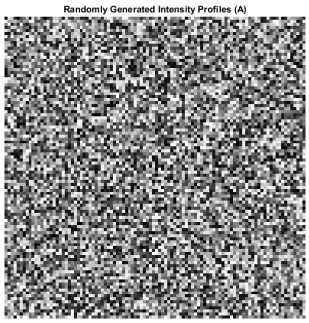
(b) (c) (d) (e)

```
1   %% clear workspace , and command window, close all figures already op
2   clear all, close all, clc;
3   %% PS0-1.3a
4   diceResults = diceTrials(99);
5   %% PS0-1.3b
6   % y = [1, 2, 3, 4, 5, 6]'
7   y = (1:6)';
8   % z = [1, 3, 5; 2, 4, 6]
9   z = reshape(y,[2,3]);
10  %% PS0-1.3c
11  % find the max value of matrice y and of which indice
12  [x, I] = max(y);
13  % convert indice to subscripts (row and column number)
14  [r, c] = ind2sub(size(z),I);
15  %% PS0-1.3d
16  % create vector v = [1, 8, 8, 2, 1, 3, 9, 8]
17  v = [1,8,8,2,1,3,9,8];
18  % alter the value of vector x
19  % the problem can be solved by two different approach
20  % 1 - x = numel(v(v==1))
21  % 2 - x = sum(v==1)
```

```
22  x = numel(v(v==1));
```

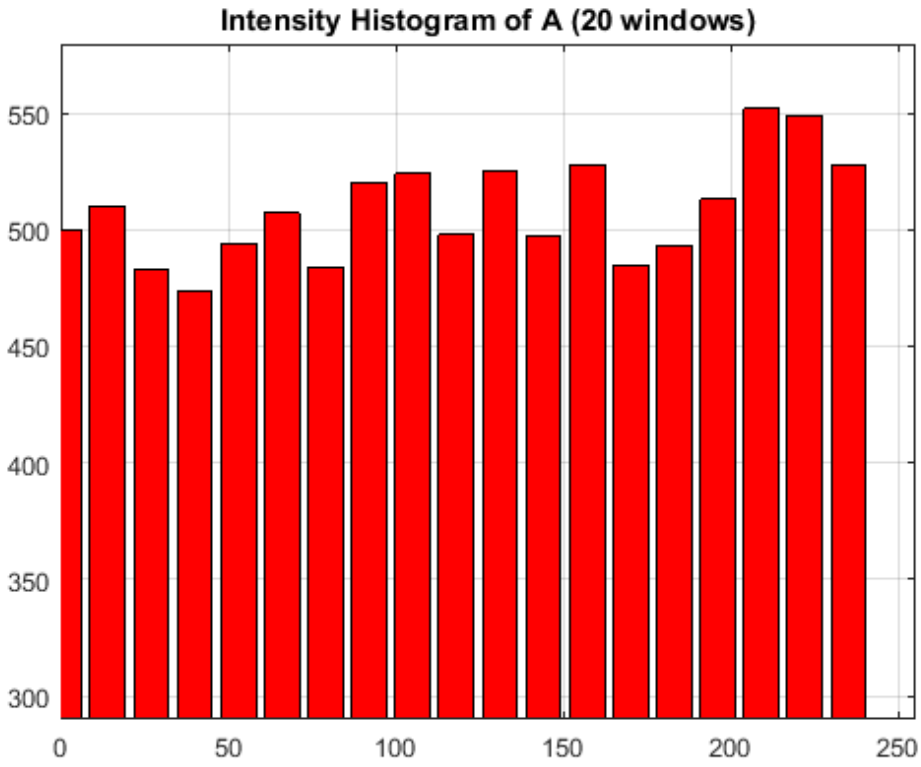
4. The code is in PS0Q1.m file.

   The source matrix created and used as input file is depicted below.

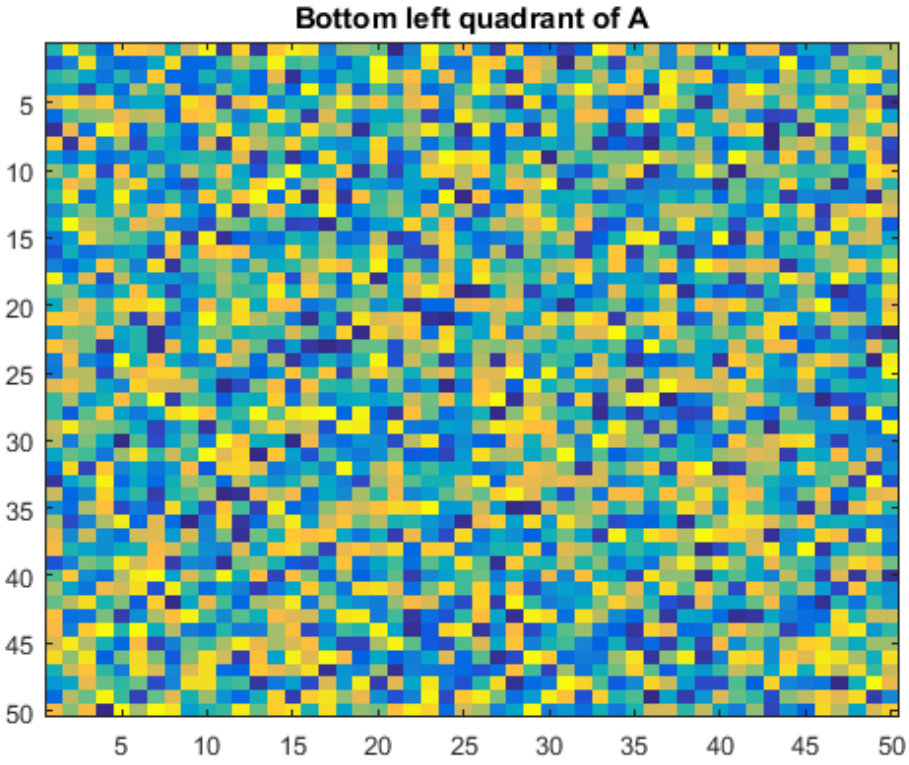


   (a) Result of sort process.

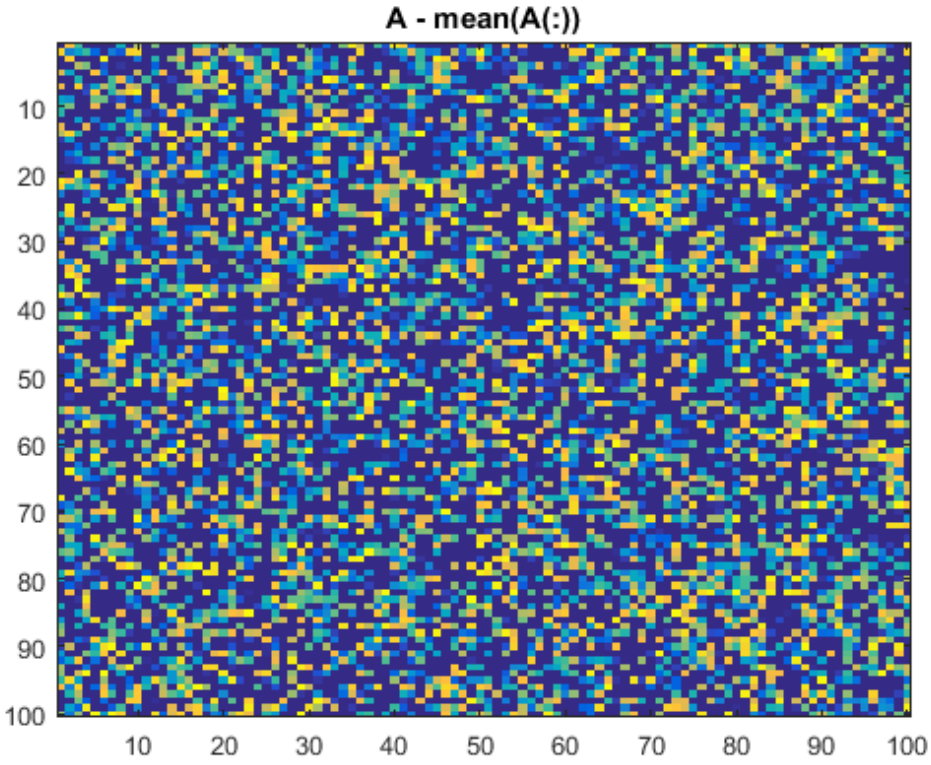
   (b) Intensity histogram of A is given below.



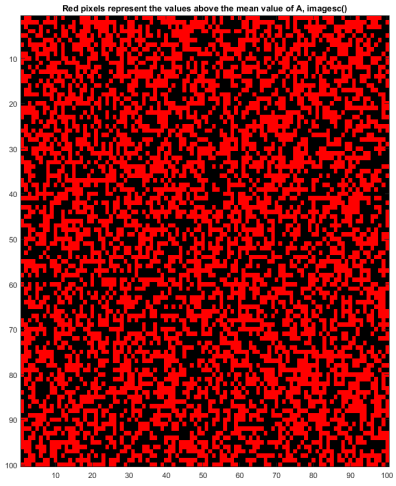
(c) The bottom left quadrant of A is depicted below. Please also find the attached outputXPS0Q1.mat in the zipped folder.

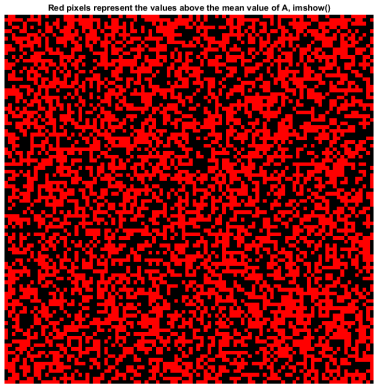


(d) Please find the attached outputYPS0Q1.mat file.



(e) Since the first matrix A is randomly created in $uint8$ data class $[0-255]$, pixels appear the same when the frame is displayed with imagesc() and imshow(). Hence, they both give the same output. Please find the attached outputYPS0Q1.mat file.





The contents of PS0Q1.m:

```
%% clear workspace , and command window, close all figures already op
clear all, close all, clc;
figure(1);
A = uint8(randi(255,[100,100]));
figure(1);
imshow(A);
title('Randomly Generated Intensity Profiles (A)')
save('inputAPS0Q1.mat', 'A');
load('inputAPS0Q1.mat', 'A');
%% PS-0 4a
A_sorted = sort(reshape(A,[numel(A), 1]), 'descend');
A_sorted = reshape(A_sorted, size(A));
figure(2);
imshow(A_sorted);
title('Sorted Intensity Values (A)')
%% PS-0 4b
bins = 20;
maxA = max(A(:));
minA = min(A(:));
range = (maxA-minA)/bins;
hist = zeros(1,bins);
y = zeros(1,bins);
for i=1:20
    hist(i) = numel(A(A>=(minA+(i-1)*range) & A<(minA+(i)*range)));
    y(i) = minA+(i-1)*range;
end
figure(3);
```

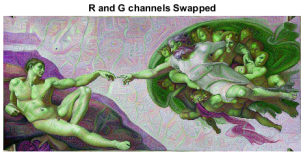
```
28 bar(y,hist, 0.8, 'r');
29 axis([0 255 min(hist) max(hist)*1.05])
30 grid on;
31 title('Intensity Histogram of A (20 windows)');
32 %% PS-0 4c
33 % X = A_sorted(size(A,1)/2:size(A,1), 0:size(A,2)/2);
34 X = A(size(A,1)/2+1:size(A,1), 1:size(A,2)/2);
35 save('outputXPS0Q1.mat','X');
36 figure(4);
37 imagesc(X);
38 title('Bottom left quadrant of A');
39 %% PS-0 4d
40 Y = A - mean(A(:));
41 save('outputYPS0Q1.mat','Y');
42 figure(5);
43 imagesc(Y);
44 title('A - mean(A(:))');
45 %% PS-0 4e
46 Z = uint8(zeros(size(A_sorted,1),size(A_sorted,2),3));
47 ind = find(A>mean(A(:)));
48 [u, v] = ind2sub(size(A),ind);
49 for i=1:numel(ind)
50     Z(u(i),v(i),:) = [255,0,0];
51 end
52 figure(6);
53 subplot(1,2,1), imagesc(Z); title('Red pixels represent the values ab
54 subplot(1,2,2), imshow(Z); title('Red pixels represent the values abo
```

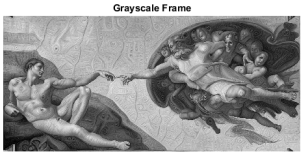
```
55 imwrite(Z, 'outputZPS0Q1.png');
```

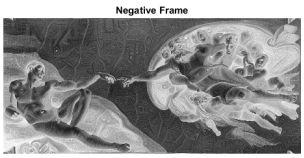
Short Programming Question

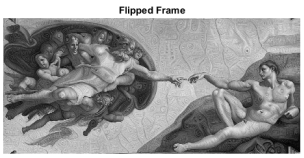
Input image:

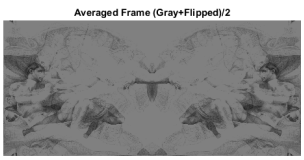
Results:

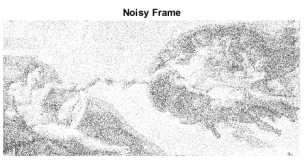












1. 

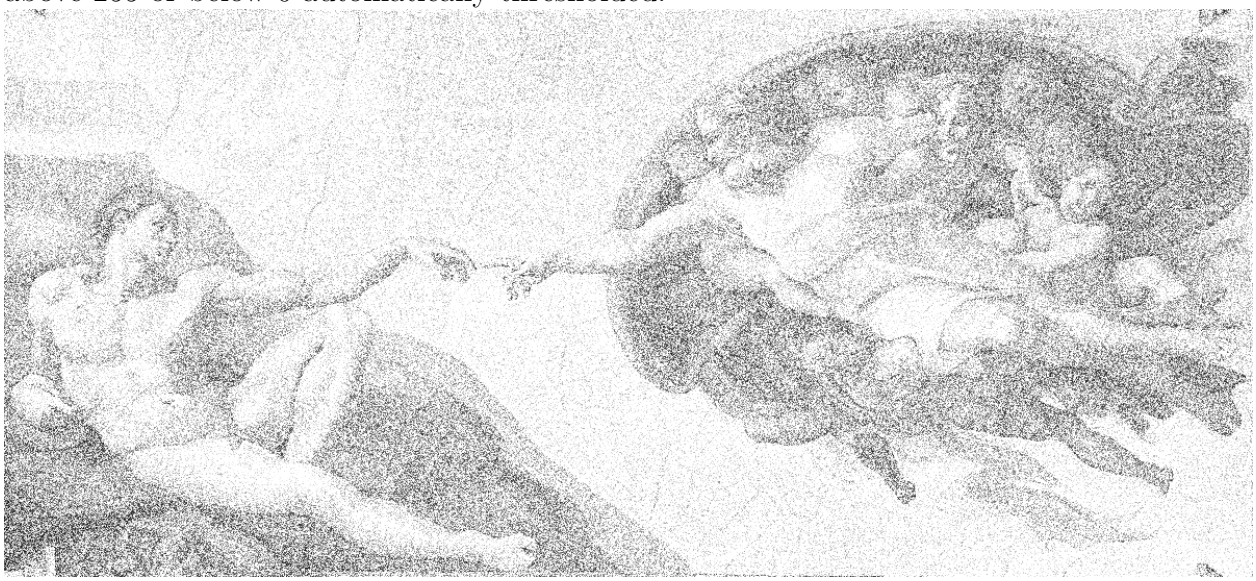
2. 

3. (a)

(b)

(c)

(d) Because all the operations was completed with *uint*8 data class, any pixel going above 255 or below 0 automatically thresholded.



The contents of PS0Q2.m:

```
%% clear workspace , and command window, close all figures already open.
close all, clear all, clc;
%% read the RGB image and create figure
frame = imread('inputPS0Q2.jpg');
figure(1);
```

```
%% swap channels
% the problem can be solved by two different approach
% 1 - built-in permute function
% 2 - traditional channel swapping
R = frame(:,:,1);
G = frame(:,:,2);
B = frame(:,:,3);
frameSwapped(:,:,1) = G;
frameSwapped(:,:,2) = R;
frameSwapped(:,:,3) = B;
imwrite(frameSwapped, 'swapImgPS0Q2.png');
subplot(3,2,1), imshow(frameSwapped);
title('R and G channels Swapped');
%% Convert RGB frame to Grayscale
frameGray =  rgb2gray(frame);
imwrite(frameGray, 'grayImgPS0Q2.png');
subplot(3,2,2), imshow(frameGray);
title('Grayscale Frame');
% the problem can be solved by two different approach
% 1 - built-in imcomplement function
% 2 - exploiting Matlab data class features
frameNegative = 255-frameGray;
imwrite(frameNegative, 'negativeImgPS0Q2.png');
subplot(3,2,3), imshow(frameNegative);
title('Negative Frame');
%% Flip the frame
frameFlipped = fliplr(frameGray);
```

```
33 imwrite(frameFlipped, 'mirrorImgPS0Q2.png');
34 subplot(3,2,4), imshow(frameFlipped);
35 title('Flipped Frame');
36 %% Average Gray Frame with Flipped
37 frameAverage = (frameGray+frameFlipped)/2;
38 imwrite(frameAverage, 'avgImgPS0Q2.png');
39 subplot(3,2,5), imshow(frameAverage);
40 title('Averaged Frame (Gray+Flipped)/2');
41 %% Add Noise
42 N = uint8(randi(255, size(frameGray)));
43 save('noise.mat', 'N');
44 frameClipped = frameGray+N;
45 imwrite(frameClipped, 'addNoiseImgPS0Q2.png');
46 subplot(3,2,6), imshow(frameClipped);
47 title('Noisy Frame');
```