Problem 1.

1. (10%) Describe the basic principle of CCD and CMOS image sensors and list at least three of their differences

Ans

Part a

(Basic principles CCD): A CCD sensor contains pixels (potential wells) arranged horizontally and vertically. As light falls into CCD the well will attract more and more electrons. The charges are transferred to an amp and then A/D to measure the voltage level for that frame. By nature this is considered an analog device.

(Basic operation CMOS) A CMOS sensor is a light capturing semiconductor tuned for various wavelengths that converts light into a voltage at the pixel location. Since the conversion is at the pixel the individual output is digital in nature. (Effectively the charge conversion is at the site)

The basic principle of a CMOS image sensor

Ans Part b

(differences): keywords: analog, power, expense, quality, speed, noise

The biggest difference is the nature of the two devices. Although both a CCD and a CMOS convert light(photons) into voltage, they do so in very different ways, a CCD is an **analog** device while CMOS is digital. This immediately dictates that a CCD will need to include an external A/D function as part of the overall sensor. There are immediately several consequences of including an external A/D, the CCD sensor will require **more power,** and as a result will be more **expensive.**  However, because of the nature of the devices, analog vs digital and the use of a A/D, the CCD will have in general more dynamic range, sensitivity, and better **quality**. Finally, a CMOS sensor, where CMOS technology is very mature and forms the foundation of modern integrated circuits, will be **faster** due to the **switching speeds** of such digital electronics. Of course with the increase of speed, CMOS sensors have more **noise** than CCDs.

Problem 2.

1. (10%) List at least five types of camera sensor noises. Describe their origin, basic characteristics and how to fix them.

Ans:

1. **Photon Shot Noise**

This noise has its origins in the independence of photons and how they arrive at detectors. The number of photons arriving during a fixed time(exposure), is governed by Poisson’s probability distribution. The basic characteristic of the noise reflected in Poisson’s distribution is that given the average of the distribution you can predict statistically whether you will get k more events.

Fortunately the SNR for this type of noise is proportional to the square root of the average number of photons arriving at the detector. With an annealing effect from the proportional square root function, increasing the detector size, pixel area by a factor of 4x will increase the SNR by a factor of 2.

1. **Dark Current Noise**

Dark current noise is a low-level physics effect due to random behavior of holes and electrons in semiconductor material. As such, it increases linearly with exposure time and exponentially with temperature. Of course, because it is related to the actual material of the device it will vary across sensors that use different semiconductor material that has different doping characteristics

The most direct way to treat this type of noise is to cool the sensor with either water conduction or air flow convection. As noted above, just dealing with the exposure time is not enough as this is a linear relationship, and the temperature is exponential.

1. **Hot Pixels**

Hot pixel noise is primarily an electrical leaking phenomenon due to manufacturing defects. It is the leakage of electrical charge into the microscopic wells of the sensor that in the assignment of electrical charge the voltages are higher than they are (proportionality to photons) that they appear brighter by error.

From the literature it seems that hot pixels appear in all detectors and are hard to deal with as they change over time. Also, they are not only related to increases in exposure time, but to increases in temperature as well, where they are hard to model. Those that are related to temperature can be dealt with in the same way that dark current noise was addressed, with cooling. However, if this does not correct the problem and the hot pixel is repeatable, then perhaps the pixel can be removed in post digital signal processing of the image. Finally, if cooling or post-processing does not work, the exposure time can be experimented with to ameliorate the effects of hot pixels.

1. **Fixed Noise Pattern**

Fixed noise patterns are the result of manufacturing variations pixel columns in CMOS sensors and give some pixels a higher value than the background noise.

Since these types of noise do not change with time, they can be read once and subtracted for all other exposures.

1. **Read Noise**

The read noise mainly affects CMOS sensors and that is because of the nature of how A/D is done with CMOS. As opposed to CCD sensors, CMOS sensors have an A/D at each pixel and as a result affect directly the ability of an individual pixel to determine changes in amplitude for small signal changes. The analogy in digital electronics and digital signal processing would be the quantization level of the system as this determines the performance of a system to provide details to small changes.

Since this type of noise has some relationship to thermal noise then cooling is a possible alleviation.

Problem 3.

1. (20%) Use MATLAB to read image file “cameraman.tif”, add three different types of Gaussian white noises with (1) mean = 0, variance = 0.01; (2) mean = 3, variance = 0.1; and (3) mean = 1, variance = 0.2, and save these three new images as png files.

Ans:

%---------------------------------------------------------------------

% file name : hmwk\_2\_prob\_3.m

% Student: Ray Duran

% Date: 9/29/21

% Class : EECS 590 Professor Liang, Fall Semester

% University of North Dakota

% Descr:

% help imnoise:

% J = imnoise(I,'gaussian',M,V) adds Gaussian white noise of mean M and

% variance V to the image I. When unspecified, M and V default to 0 and

% 0.01 respectively.

%---------------------------------------------------------------------

J = imread('cameraman.tif');

% Add noise

JN1 = imnoise(J,'gaussian',0,0.01);

JN2 = imnoise(J,'gaussian',3,0.1);

JN3 = imnoise(J,'gaussian',1,0.2);

% display

figure

subplot(1,4,1)

imshow(J)

title('Original')

subplot(1,4,2)

imshow(JN1)

title('Noise Mean=0 Var=.01')

subplot(1,4,3)

imshow(JN2)

title('Noise Mean=3 Var=.1')

subplot(1,4,4)

imshow(JN3)

title('Noise Mean=1 Var=.2')

% Save files

imwrite(JN1,'cameraman\_gauss\_m\_eq\_0\_v\_eq\_dot01.tif');

imwrite(JN2,'cameraman\_gauss\_m\_eq\_3\_v\_eq\_dot1.tif');

imwrite(JN3,'cameraman\_gauss\_m\_eq\_1\_v\_eq\_dot02.tif');



Problem 4

1. (30%) Figure 1 is an image taken by an optical microscopy with 40x magnification (meaning that the actual sample is magnified by 60 times on the image sensor). The pixel size of the microscope camera is 6.5 m. Please use image processing method to estimate the size of the cell (indicated with red arrow), including length of the long and short axes, and the area of the cell. Please describe your calculation step-by-step.

Ans:

**From Matlab code:**

**Major axis length = 44.3 mm**

**Minor axis length = 18.5 mm**

**Size of cell = 2.6 mm^2**

%---------------------------------------------------------------------

% file name : hmwk\_2\_prob\_4\_optical\_microscopy.m

% Student: Ray Duran

% Date: 9/29/21

% Class : EECS 590 Professor Liang, Fall Semester

% University of North Dakota

% Descr:

% Estimate the number of pixels in cell.tif

% Code borrowed from Matlab example

% https://www.mathworks.com/help/images/detecting-a-cell-using-image-segmentation.html

%---------------------------------------------------------------------

clf

clear

%% sobel filter - edge detection adn cleanup of image; From MATHWORKS

I = imread('cell.tif');

[~,threshold] = edge(I,'sobel');

fudgeFactor = 0.5;

BWs = edge(I,'sobel',threshold \* fudgeFactor);

figure(1)

imshow(BWs)

title('Binary Gradient Mask')

se90 = strel('line',3,90);

se0 = strel('line',3,0);

BWsdil = imdilate(BWs,[se90 se0]);

figure(2)

imshow(BWsdil)

title('Dilated Gradient Mask')

BWdfill = imfill(BWsdil,'holes');

figure(3)

imshow(BWdfill)

title('Binary Image with Filled Holes')

BWnobord = imclearborder(BWdfill,4);

figure(4)

imshow(BWnobord)

title('Cleared Border Image')

seD = strel('diamond',1);

BWfinal = imerode(BWnobord,seD);

BWfinal = imerode(BWfinal,seD);

figure(5)

imshow(BWfinal)

title('Segmented Image');

%% Convert to uint8 for use with image processing functions

rows = size(BWfinal,1);

cols = size(BWfinal,2);

for i = 1 : rows

for j = 1 : cols

if ( BWfinal(i,j) == 1)

BWfinal\_gray(i,j) = uint8(255);

else

BWfinal\_gray(i,j) = uint8(0);

end

end

end

%% Calculate Area with elliptical axes

% Rotation of the image

degree = 140;

img = BWfinal\_gray;

imgr = imrotate(img,degree,'bilinear','crop');

figure(6)

imshow(imgr);

% Translation of the image

imgt = imtranslate(imgr,[-20,0],'FillValues',0);

figure(7)

imshow(imgt);

% Lets find approx of the minor axis of elliptical which would be the

% height of the fish

[icoor,jcoor,v] = find(imgt);

nnz\_size = size(jcoor,1);

nnz\_count = zeros(1, cols);

for j = 1 : cols

for k = 1 : nnz\_size

if ( jcoor(k) == j)

nnz\_count(j) = nnz\_count(j) + 1;

end

end

end

minor\_axis\_image = max(nnz\_count);

debug = 1;

% Let's find major axis

imgt\_transposed = imgt';

figure(8)

imshow(imgt\_transposed);

debug = 1;

[icoor2,jcoor2,v2] = find(imgt\_transposed);

nnz\_size = size(jcoor2,1);

nnz\_count = zeros(1, cols);

for j = 1 : cols

for k = 1 : nnz\_size

if ( jcoor2(k) == j)

nnz\_count(j) = nnz\_count(j) + 1;

end

end

end

major\_axis\_image = max(nnz\_count);

%

% True pixel size is 6.5e-6\* 40(mag power)

pixel\_size = (6.5/40)\*1e-6; % one aquare

pixel\_dim = sqrt(pixel\_size);

length\_major\_axis = major\_axis\_image\*pixel\_dim;

length\_minor\_axis = minor\_axis\_image\*pixel\_dim;

% Size of the cell

size\_of\_cell = pi\*length\_major\_axis\*length\_minor\_axis;

After rotation and translation and cleanup to estimate see figures 1 & 2



Figure 1. Vertical image: To estimate major axis



Figure 2. Horizontal image: To estimate minor axis

Problem 5.

5. (30%) Figure 2 is an image of rice grains with *Salt and Pepper* noise (See attached file Rice\_noise.png). Please use image processing method to roughly estimate the number of rice grains. Please describe your calculation step-by-step.

**Ans: Estimate from algorithm is approximately 106 grains.**

Step by step in the MATLAB code below and annotated in figures.

%---------------------------------------------------------------------

% file name : hmwk\_2\_prob\_5\_estimate\_rice\_grains\_in\_noise.m

% Student: Ray Duran

% Date: 9/29/21

% Class : EECS 590 Professor Liang, Fall Semester

% University of North Dakota

% Descr:

% Estimate the number of rice grains

%---------------------------------------------------------------------

image = imread('rice\_noise.png');

figure(1)

imshow(image)

% Filter

JI = medfilt2(image);

figure(2)

imshow(JI)

figure(3)

imhist(JI)

% Adjust for darkness at bottom of image

% histogram used for range

rows = size(JI,1);

cols = size(JI,2);

for i = 1 : rows

for j = 1 : cols

if ( (JI(i,j) > 45 ) && (JI(i,j) < 75) )

bw1(i,j) = 1;

else

bw1(i,j) = 0;

end

end

end

figure(4)

imshow(bw1)

[L1,num1] = bwlabel(~bw1,4)

k = 1;

% crop image

% Manually afjust for dark area

for i = 188 : rows

for j = 1 : rows

bw1\_crop(k,j) = bw1(i,j);

end

k = k +1;

end

figure(5)

imshow(bw1\_crop)

[L1,num1] = bwlabel(bw1\_crop,4)

bw2 = JI > 150

figure(6)

imshow(bw2)

[L2,num2] = bwlabel(bw2,4)

% Estimate

rice\_grain\_count = num1 + num2;

X = ['Estimate of grain count is:', num2str(rice\_grain\_count)];

disp(X)



Figure 1. Image original : The original noisy image



Figure 2. Filtered image: Noisy image filtered with Median filter



Figure 3. Histogram : Histogram to adjust for darkness on bottom of picture.



Figure 4.



Figure 5. Bottom of Image: We cropped bottom of image to process separately



Figure 6. Top of image: Isolated top of image to process separately from bottom